A fully Bayesian approach to financial forecasting and portfolio selection

Andrew Simpson

NEWCASTLE UNIVERSITY LIBRARY
201 29543 0
Thesis L7349

Submitted for the Degree of Doctor of Philosophy,

University of Newcastle upon Tyne,
September 2002

Abstract

The movements of share prices has long been of interest to both academic researchers as well as market practitioners. The statistical research in this field dates back to the work of Bachelier (1900) and there have been many approaches adopted subsequently.

This thesis considers a Bayesian approach to multivariate forecasting of financial time series based on dynamic linear models. We will also consider the forecasting of the returns distribution using stochastic volatility models. We will then look at combining these two model structures. We will also demonstrate how the posterior forecast distribution can be simulated and how this may be used directly in order to implement a fully Bayesian decision theoretic approach to selection of optimal stock portfolios.

These methods are first illustrated on simulated data and then applied to real data for selected shares from the Standard and Poor 500.

Acknowledgements

I would like firstly to express my sincere gratitude to Darren Wilkinson for his guidance, encouragement and support throughout my time under his supervision. I would like also to thank the University of Newcastle upon Tyne for providing funding for my research and the Department of Statistics for giving me the opportunity to conduct this research.

I would like to thank my mother and the rest of my family for the support and encouragement they have expressed over the years. I thank also my many friends both old and new for their friendship in good times and bad.

Finally I would like to express my eternal thanks to my late partner David Chatt, without whose love and support I would never have commenced this research, let alone completed this thesis.

Contents

List of Figures

Chapter 1

Introduction

1.1 Motivation

The movements of share prices has long been of interest to academic researchers as well as practitioners within the market. Statistical research dates back over a century to the seminal work of Bachelier (1900), who's paper was the first not only to propose a random walk mechanism for share price movements but also to derive formula for the pricing of options. Subsequently there have been many different approaches to looking at share prices and investment decision making.

To borrow the format of Miller (1999) four schools of approaching investment decision making have subsequently arisen. These are examined in turn.

1.1.1 Markowitz and the Theory of Portfolio Selection

The Markowitz (1959) approach to portfolio selection is to consider the mean returns and variance/covariance of a portfolio of shares and select that portfolio which provides the best possible return for a particular investor depending on their view of an acceptable level of risk. Perhaps one of the most important concepts to arise from this paper is that the relevant unit of analysis for an investor is the portfolio, not individual shares, the covariance between shares are key to reducing the risks involved in speculative investment.

Markowitz assumed that the means and variance/covariances could be estimated from the his-

toric data and that not only were these Gaussian but also that they would hold into the future. This is unlikely to be the case and the provision of predictive values for these parameters will be a major part of the thesis which follows.

1.1.2 Capital Asset Pricing Model

The Capital Asset Pricing Model (CAPM) of Sharpe (1970) proposed that all investors would hold the market portfolio, i.e. that portfolio which matched the proportions of the market as a whole. This is now common with many mutual funds and index tracking funds. In this model investors reduce the risk they face by holding proportions of their assets in "non-risky" assets, such as bank deposits or in gilts. There are limitations to CAPM, especially in relation to the level of transaction costs required to maintain the market portfolio.

1.1.3 The Efficient Market Hypothesis

The efficient market hypothesis states in simple terms that you can not buck the market. Essentially no rule can exist based on currently available information that allows an investor to generate above normal rates of return on the investment. This was supported by the random walk view on stock prices as first proposed by Bachelier (1900), which implies that historic patterns and trends, no matter how apparent they appear, have no predictive power for future prices. However from the mid-sixties with the work of Fama (1965) empirical evidence had accumulated to the extent as to be able to say with some conviction that stock prices were not random walks in the strictest sense, at least some predictability could be detected. The efficient market hypothesis does, however remain popular with researchers, especially those from an economics background. It could still carry some validity, if markets work, then even if a researcher finds some method to derive above normal returns the reaction of the market will eventually erode that advantage and return the market to equilibrium. "The apparent situation is that financial markets are in a persistent state of near efficiency, meaning that excess money consistently is possible but very difficult." (Qunintana, Lourdes, Aguilar, and Liu 2003).

1.1.4 Options

The pricing of options is one of the major areas of financial research over the last twenty five years or more. The work of Bachelier (1900) was concerned with the pricing of options, however modern research can be attributed to the paper Black and Scholes (1973). The pricing of options under the "Black-Scholes-Merton" formula depends mainly on observables, namely the current market price of the underlying share, the striking price, the time to maturity of the contract and the risk free interest rate. It also depends crucially on the volatility of the log returns distribution of the underlying share, which is not directly observable. The estimation of this volatility of the log returns has lead to some of the estimation techniques that will be discussed later in this thesis.

1.2 Bayesian Inference

This thesis will consider the problem of making inference on, or providing estimates for the parameters of a variety of models. Frequentist approaches to these models are difficult as the models are complex and predictions of future values are required. Maximum Likelihood estimators are popular with many statisticians and can be appropriate for use with complex models. They, however, fail to provide a mechanism for the incorporation of prior beliefs and they do not provide probabilistic information about the estimates for the model parameters.

Bayesian inference overcomes these two concerns. If we have some data x which we believe can be modelled with a probability density function $f(x|\theta)$, then the likelihood of θ is $L(\theta|x)$. We can express the prior beliefs about the parameter θ in terms of a probability density function $\pi(\theta)$, then the posterior probability density function for θ, $\pi(\theta|x)$ is given by

$$\pi(\theta|x) \propto \pi(\theta) \times \mathrm{L}(\theta|x) \tag{1.1}$$

$$\text{i.e. Posterior } \propto \text{ Prior } \times \text{ Likelihood}$$

The above relationship (1.1) is the underpinning result of Bayesian statistics, it links the prior beliefs about a model's parameters to the data, as expressed through the likelihood to allow us to draw inferences about the model parameters. From this relatively simple expression a whole

range of statistical techniques have arisen to help us make inference about complex models. This thesis will follow the Bayesian paradigm for inference on complex models within the context of financial scenarios.

1.3 Bayes Linear Inference

In Section 1.2 Bayesian Inference has been outlined as the preferred method for making inference about the parameters of complex models. A fully Bayesian approach requires the specification of full joint probability structures, this can be difficult to do in highly complex problems. A Bayes Linear Analysis overcomes this requirement and so can be a preferable approach to such complex problems.

We can define the subjective expectation of a discrete random quantity as

$$\mathrm{E}(X) = \sum_{i=1}^{\infty} x_i P(X = x_i)$$

where $P(X = x_i)$ is the subjective probability of the event $(X = x_i)$ occurring. In Bayes Linear Analysis we assume that assigning subjective probabilities to all possible events is too difficult and so subjective expectation is made the fundamental quantity of interest. The use of subjective expectation allows us to make statements about the location of the parameter of interest, about which we are uncertain. We are often also interested in the spread about that location. Subjective expectation is a powerful tool and allows one to define the subjective variance and covariance as follows:

$$\mathrm{Var}(X) = \mathrm{E}\left(X^2\right) - \mathrm{E}(X)^2$$

$$\mathrm{Cov}(X, Y) = \mathrm{E}(X, Y) - \mathrm{E}(X)\,\mathrm{E}(Y).$$

Bayes Linear Analysis uses subjective expectation as its cornerstone and is, in spirit, similar to a fully Bayesian approach. The prime difference being that it is based on a simpler approach to prior specification and analysis. In a Bayes Linear approach we make prior specification for the collection of means, variances and covariances we are interested in and then update these

via linear fitting. If we have two random vectors B and D whereby observing D we hope to make some inference about B. The adjusted, or Bayes Linear expectation of B_i given D is the linear combination of ${a_i}^T D$ which minimises $\mathrm{E}\left(\left(B_i - {a_i}^T D\right)^2\right)$ over a_i. To undertake this adjustment we must first specify prior mean vectors and the variance matrices for B and D, as well as the covariance matrix between B and D. The *adjusted expectation vector* $\mathrm{E}_D(B)$ for $B|D$ is evaluated as

$$\mathrm{E}_D(B) = \mathrm{E}(B) + \mathrm{Cov}(B, D)\mathrm{Var}\,(D)^{-1}\left[D - \mathrm{E}\,(D)\right] \tag{1.2}$$

This is known as the Bayes Linear Adjusted Expectation. Likewise the Bayes Linear Adjusted Variance is given by

$$\begin{aligned}\mathrm{Var}_D(B) &= \mathrm{Var}\left(B - \mathrm{E}_D\left(B\right)\right) \\ &= \mathrm{Var}(B) - \mathrm{Cov}(B, D)\mathrm{Var}(D)^{-1}\mathrm{Cov}(D, B)\end{aligned} \tag{1.3}$$

One interesting thing to note about both (1.2) and (1.3) is that the Bayes Linear Adjusted expectation and variance can be considered to be made up of the original estimate adjusted slightly in light of the data. For a more detailed treatment of Bayes Linear Inference see Goldstein and Wooff (1995)

Within the Bayesian framework, adjusted expectation offers a tractable approximation to conditional expectation which can be useful in complex problems. Adjusted variance is a strict upper bound to expected posterior variance over all prior specifications consistent with the given moment structure. "These approximations are exact in certain cases, most particularly if the joint probability distribution of B,D is multivariate normal. This leads to formal relationships between the analysis of Gaussian structures and Bayes Linear calculations" Goldstein (1998). This arises in the case of Dynamic Linear Models which will be returned to in Chapter 2.

1.4 Markov Chain Monte Carlo

In Section 1.2 and Section 1.3 it was stated how Bayesian inference provides a suitable framework for the analysis of complex models, allowing for the incorporation of prior beliefs and

providing probabilistic information about the model parameters. Fully Bayesian analytical approaches to complex problems, however, are at best prohibitively complex and can often be intractable. If we abandon the analytical approach and adopt stochastic simulation techniques then the Bayesian analysis of almost all statistical problems is possible. In many cases what are required are integrals of complex high-dimensional probability distributions. Techniques for Monte Carlo (MC) integration using Markov Chains (MC) known as Markov Chain Monte Carlo (MCMC) can be used to find these integrals (see Gamerman (1997) for a general treatment and the references therein).

Simply put, these techniques involve sampling from a posterior distribution of interest, by simulating values from a Markov Chain which has the posterior distribution of interest as its stationary distribution. Key to this, is of course, constructing a Markov Chain that has precisely the distribution of interest as its stationary distribution.

Suppose we wanted to simulate values from a random vector $\underline{\theta} = (\theta_1, \theta_2, \cdots, \theta_k)$, which is possibly high-dimensional. The best strategy depends on what is known about the distribution of $\underline{\theta}$ and especially about the structure of the density for $\underline{\theta}$. One standard method assumes knowledge of each of the distributions

$$\begin{aligned}
&\theta_1 \\
&\theta_2|\theta_1 = x_1 \\
&\theta_3|\theta_1 = x_1, \theta_2 = x_2 \\
&\quad\vdots \\
&\theta_k|\theta_1 = x_1, \theta_2 = x_2, \cdots, \theta_{k-1} = x_{k-1}
\end{aligned}$$

In this case one could simulate on observation from $\underline{\theta}$ by progressing through these distributions, hierarchically, simulating component values from each in turn, conditioning on the simulated values of earlier components. This is a standard algorithm for simulating multivariate normal observations. In reality rarely do we know these distributions and hence we need to turn to MCMC techniques. One such technique which could be used when the above listed distributions are not known, but when their univariate conditions are, is the Gibbs Sampler.

1.4.1 The Gibbs Sampler

The Gibbs sampler is a MCMC technique which allows simulation from the joint distribution when only the full conditionals are known and can be simulated from. It is particularly appropriate when sampling from the marginal distributions is either not convenient or not possible. Using the square bracket notation of Gelfand and Smith (1990), let $[\underline{\theta}]$ denote the joint density of a vector of parameters $\underline{\theta} = (\theta_1, \theta_2, \ldots, \theta_k)$. The situation often arises where the marginal density

$$[\theta_i] = \int \cdots \int \int \cdots \int [\underline{\theta}] d\theta_1 \ldots d\theta_{i-1} d\theta_{i+1} \ldots d\theta_k, \tag{1.4}$$

or some information pertaining to the density, such as its mean or variance is required. The most natural approach and perhaps straightforward approach would be to calculate the density and use it to obtain the desired characteristics. The integrations, however, necessary to compute the marginal density, whether analytically or via numerical methods, might be intractable or at least highly problematic. Gibbs sampling allows us to learn about such densities without having to perform integrals such as (1.4). The Gibbs sampler, instead of determining or approximating $[\theta_i]$ directly, is an algorithm which allows one to generate a sample from the density of $[\underline{\theta}]$. Given the full conditional distribution $\left[\theta_i | \underline{\theta}_{i\backslash}\right]$, for each element θ_i, where $\underline{\theta}_{i\backslash}$ denotes the vector $\underline{\theta}$ with the i^{th} element omitted. The Gibbs sampler follows the following algorithm:

1. Initialise the iteration counter to $j = 1$ and initialise the state of the chain to $\underline{\theta}^{(0)} = \left({\theta_1}^{(0)}, {\theta_2}^{(0)}, \ldots, {\theta_k}^{(0)}\right)$.

2. Obtain a new value $\underline{\theta}^{(j)}$ from $\underline{\theta}^{(j-1)}$

$$\begin{aligned}
{\theta_1}^{(j)} &\sim \left[\theta_1 | {\theta_2}^{(j-1)}, {\theta_3}^{(j-1)}, {\theta_4}^{(j-1)}, \ldots, {\theta_k}^{(j-1)}\right] \\
{\theta_2}^{(j)} &\sim \left[\theta_2 | {\theta_1}^{(j)}, {\theta_3}^{(j-1)}, {\theta_4}^{(j-1)}, \ldots, {\theta_k}^{(j-1)}\right] \\
{\theta_3}^{(j)} &\sim \left[\theta_3 | {\theta_1}^{(j)}, {\theta_2}^{(j)}, {\theta_4}^{(j-1)}, \ldots, {\theta_k}^{(j-1)}\right] \\
&\vdots \\
{\theta_k}^{(j)} &\sim \left[\theta_k | {\theta_1}^{(j)}, {\theta_2}^{(j)}, {\theta_3}^{(j)}, \ldots, {\theta_{k-1}}^{(j)}\right]
\end{aligned}$$

3. Increment the counter from j to $j + 1$ and return to step 2.

In essence what this algorithm does is sample each parameter of interest from its full univariate conditional distribution, conditioning on the current value of each of the other parameters. This clearly defines a homogeneous Markov Chain. Each simulated value depends only on the previous simulated value, not on previous values or the iteration counter. This process is repeated t times until the vector $\underline{\theta}^{(t)}$ is obtain. The motivation behind implementing a Gibbs sampler is that, it can be shown that, the simulated values converge in distribution to realisations of random variables from the required distribution $[\underline{\theta}]$. Hence, for a sufficiently large t the value $\underline{\theta}_i{}^{(t)}$ represents a value sampled from the marginal distribution of θ_i. One key question that arises is whether or not the Markov Chain has reached its stationary distribution? The sampler needs to be run for a "burn-in" period, to ensure the chain has converged and this "burn-in" is discarded before analysis of the parameters can be undertaken. What is required is a sample of size T from the joint posterior distribution, however opinion is divided on how best to obtain this. One method suggests that the sampler is run T times for t iterations and only this iterate is retained as the sample value. Obviously the chain needs to be started at random values so that the sample will consist of random draws from $[\underline{\theta}]$. An alternative is to run the sampler beyond the "burn-in" stage and take each of $\underline{\theta}^{(t)}, \underline{\theta}^{(t+1)}, \underline{\theta}^{(t+2)}, \ldots, \underline{\theta}^{(t+T)}$ as sample values from $[\underline{\theta}]$. This technique relies on the fact that once equilibrium has been achieved subsequent iterations from the sampler will also yield realisations from the required distribution. This method requires less computational time, however, the sample is not a random sample of independent observations, because successive elements of the chain are correlated.

The issue of convergence of the Markov chain is wide ranging and beyond the scope of this thesis. The two review papers by Mengersen, Robert, and Guihenneuc-Jouyaux (1998) and Cowles and Carlin (1996) provide a more detailed treatment for the interested reader. The important fact to recall is that no convergence diagnostic can say with any certainty whether or not a Gibbs sampler has converged to its underlying stationary distribution. One of the simplest methods of detecting convergence is by visual inspection of the trace plots of the MCMC output. A useful tool to further check convergence is CODA Best, Cowles, and Vines (1995). This is a software package which can be used in conjunction with the statistical software package **R** to perform the various diagnostic techniques reviewed in the above papers.

1.4.2 The Metropolis-Hastings Sampler

Metropolis-Hastings schemes are a widely used class of methods for constructing a reversible Markov Chain which has the distribution of interest as its stationary distribution. These methods are useful when the Gibbs sampler outlined in Section 1.4.1 is not applicable, i.e when we can not easily sample from the distribution of interest.

Suppose the density of interest is $\pi\left(\underline{\theta}\right)$, what we require are sample values from this density. Suppose we have a *proposal distribution* $q\left(\underline{\theta}, \underline{\phi}\right)$, which is infact the transition kernel of a Markov Chain. This proposal distribution needs to be easy to simulate from, or at least easier to simulate from than the distribution for $\pi\left(\underline{\theta}\right)$ or there is no advantage to this methodology. The proposal distribution does not necessarily have to have $\pi\left(\underline{\theta}\right)$ as its stationary density.

The Metropolis-Hastings algorithm can be used to simulate a sample from the density of interest $\pi\left(\underline{\theta}\right)$ using a Markov chain as follows.

1. Initialise the iteration counter to $j = 1$ and initialise the state of the chain to $\underline{\theta}^{(0)} = \left(\theta_1{}^{(0)}, \theta_2{}^{(0)}, \ldots, \theta_k{}^{(0)}\right)$.

2. Simulate a proposed value $\underline{\phi}$ using the kernel $q\left(\underline{\theta}^{(j-1)}, \underline{\phi}\right)$.

3. Evaluate the *acceptance probability* $\alpha\left(\underline{\theta}^{(j-1)}, \underline{\phi}\right)$ of the proposed move where

$$\alpha(\theta, \phi) = \min\left\{1, \frac{\pi(\underline{\phi})q\left(\underline{\phi}, \underline{\theta}\right)}{\pi(\underline{\theta})q\left(\underline{\theta}, \underline{\phi}\right)}\right\}. \tag{1.5}$$

4. With probability $\alpha\left(\underline{\theta}^{(j-1)}, \underline{\phi}\right)$, set $\underline{\theta}^{(j)} = \underline{\phi}$, otherwise set $\underline{\theta}^{(j)} = \underline{\theta}^{(j-1)}$.

5. Increment the counter from j to $j + 1$ and return to step 2.

Put simply at each iteration of the sampler a new value is generated from the proposal distribution $q(\underline{\theta}, \underline{\phi})$ which is either accepted or rejected. If accepted the chain moves to a new value, if rejected the chain maintains its current position. Whether the move is accepted or not depends on the acceptance probability, $\alpha(\underline{\theta}^{(j-1)}, \underline{\phi})$, which depends on the relationship between the density of interest and the proposal distribution. The density of interest $\pi\left(\underline{\theta}\right)$ only enters the acceptance probability as a ratio, hence this method can be used when the density of interest

is known only upto a scaling constant. Whether or not the move is accepted depends on the generation of a random uniform $[0, 1]$ value, u at step 3. If $u \leq \alpha(\underline{\theta}^{(j-1)}, \underline{\phi})$ then the move is accepted, if $\alpha(\underline{\theta}^{(j-1)}, \underline{\phi}) > u$ then the move is rejected and the chain maintains its current position.

1.5 Utility Theory

We have already stated we are interested in using the Bayesian paradigm to make inference on complex financial models, however we are also interested in using that modelling to allow us to make decisions on the portfolio of shares to hold. Utility theory gives us a methodology to aid in that decision process.

If one has a choice between two rewards, R_1 and R_2, one could easily rank those as to one's preference for one above the other. If R_2 is preferred to R_1 this can be written as $R_1 \prec^* R_2$. If one is indifferent between the two rewards R_1 and R_2, this can be written as $R_1 \sim^* R_2$. If one definitely does not prefer R_2 to R_1, i.e. if either $R_2 \prec^* R_1$ or $R_1 \sim^* R_2$ holds then this can be written as $R_2 \preceq^* R_1$. With a collection of n rewards one can create a preference ordering, such that

$$R_1 \preceq^* R_2 \preceq^* \cdots \preceq^* R_n$$

Where R_1 in this case is the least preferable reward and R_n is the most preferred, in the opinion of a particular individual.

A gamble G can be considered as a random reward. Even though gambles are random they can still be ordered by preference. The following gamble,

$$G = p_1 R_1 +_g p_2 R_2 +_g \cdots +_g p_n R_n$$

can be considered as returning reward R_1 with probability p_1, reward R_2 with probability p_2 and so on upto reward R_n with probability p_n. If you had set the preference $R_2 \preceq^* R_1$ and you are then then presented with a gamble, $G = \frac{1}{2} R_1 +_g \frac{1}{2} R_2$, then including this gamble into your preference order would yield $R_2 \preceq^* G \preceq^* R_1$. This is obvious as if you prefer R_1 to R_2

then you would naturally prefer the gamble, G to R_2 as this gives you at least a half chance of obtaining your preferred reward R_1, likewise for preferring R_1 to the gamble G.
Such preference ordering is interesting, however it does not tell you how much you prefer R_1 to R_2. What is required is some form of quantitative measure of preference , utility is that measure. A utility function $u(\cdot)$ maps rewards and gambles to the real line, preserving the preference ordering. Hence if your preference ordering was, $R_2 \preceq^* R_1$, then $u(R_2) \leq u(R_1)$. The utility of a gamble is defined as follows

$$\begin{aligned} u(G) &= p_1 u(R_1) + p_2 u(R_2) + \cdots + p_n u(R_n) \\ \Rightarrow u(G) &= \mathrm{E}\left(u\left(G\right)\right) \end{aligned} \tag{1.6}$$

The above provides a basis for utility based decision making. Preference ordering is preserved and hence we should always choose the reward which yields the largest utility. The result given in (1.6) tells us that given a collection of gambles over a set of rewards you should always choose the gamble with the highest expected utility. This result is important and we will come back to it in Chapter 6 when we consider issues of portfolio selection.
If the rewards are monetary one could just consider the Expected Monetary Value (EMV) of a gamble, i.e.

$$EMV(G) = p_1 £R_1 + p_2 £R_2 + \cdots + p_n £R_n$$

as opposed to the utility

$$u(G) = p_1 u(R_1) + p_2 u(R_2) + \cdots + p_n u(R_n)$$

To compare these two we require the following two definitions. The *certainty equivalent* of G is the amount of money $£R$ such that one is indifferent between $£R$ and G, i.e. $£R \sim^* G$, or $u(£R) = u(G)$. The *risk premium* is defined as the amount Π_G where

$$\Pi_G = EMV(G) - £R$$

If $\Pi_G > 0, \forall G$ for a particular decision maker they can be said to be risk averse. If for a decision maker $\Pi_G < 0, \forall G$, they are said to be risk prone, and if $\Pi_G = 0, \forall G$ then they can be said to be risk neutral. If a decision maker is risk neutral then maximising EMV is a reasonable decision making procedure, this can be the case, especially for large organisations when considering relatively small amounts of money. However as the amounts considered become larger then decision makers tend to become more risk averse, and it is risk averse situations we will consider in more detail in Chapter 6. Risk averse utility functions are always concave in form and in later chapters we will discuss the possible forms of these in more detail.

1.6 Programming aspects

The nature of the research within this thesis is highly computational and we have used two sets of software to enact the models and algorithms discussed. Much initial work was done in the statistical software package **R**, however this was too slow for more serious computation. It is however useful for the analysis and graphical representation of results from other programmes. The serious programming was done in the object orientated programming language **sather** . The advantage of object orientated languages are that they allow for the construction of modular libraries and rich data structures which can be called from a variety of programmes. **sather** was chosen specifically because it was developed with scientific computing in mind and hence deals efficiently with matrices and other mathematical objects. For more details on **sather** see Omohundro and Lim (1992).

1.7 Structure

Having outlined the motivation for this research and the general paradigm under which the research will be conducted, we can go on to look at the problem in more detail. In Chapter 2 we will consider the basic model structure that will be used throughout this thesis and ways in which we can make inference about the unobservable state of this model. In Chapter 3 we will look at how we can either estimate or make inference about the parameters of this model and we will demonstrate this with simulated examples. In Chapter 4 we will consider a related

model structure which allows for comparison with the continuous time models for share price movement. In Chapter 5 we will look at attempts to combine these two models. In Chapter 6 we will look at how the predictions that stem form these models can be used to aid in the selection of portfolios. Finally in Chapter 7 we will demonstrate the techniques on real market data.

Chapter 2

The Dynamic Linear Model

2.1 Introduction

This thesis will be concerned with the modelling of financial time series, both univariate and multivariate. Before continuing it is useful to consider what is meant by this. The Oxford English Dictionary defines a model as "a simplified description of a system . . . that assists calculation and prediction". Alternatively "The reason behind modelling is to provide efficient learning processes that will increase understanding and enable wise decisions" West and Harrison (1997). These two definitions of models are essentially identical; what they say is that models do not necessarily represent truth, that is they do not capture all the complexity of the real world, rather they are tools which allow greater understanding of the systems of interest and so aid the decision process of those trying to use the systems in practical applications.

Models can incorporate factual information or data but they can also include personal experience. In fact all models are by definition subjective, that subjectivity in part coming from experience of the model builder. They are also based on the past with the hope that this provides a guide to the future. The ability of a modelling system to be able to learn and update is fundamental to a good model. Considering this need to be able to incorporate prior beliefs into a model and the necessity for updating leads us to consider modelling under a Bayesian paradigm, which as discussed in Section 1.2 will be the central paradigm of this thesis.

The other simple point to clarify is what are time series. "A time series is defined as a series of observations taken sequentially over time. It is this order property which is crucial to time

series" Pole, West, and Harrison (1994). The prices of openly traded shares and currencies quite naturally follow such an ordering. Prices are quoted monthly, weekly daily, hourly or even more frequently, providing the sequential ordering required to be defined as time series data, it is such financial time series that will be of particular interest to us in this thesis.

2.1.1 Notation

The models used in this chapter and through out this thesis will draw on some basic notation and it is perhaps wise to introduce this before the models are presented in more detail.

Let Y represent the real value of the quantities observed over time. This can either be a series of individual observations or a series of vectors of observations depending whether we are looking at univariate or multivariate series. The time index t is used as a suffix for the time series, hence Y_t represents the value of Y at the t^{th} time point. Observations begin at time $t = 1$ and develop $Y_1, Y_2, Y_3, \ldots$ or $Y_t(t = 1, 2, 3, \ldots)$. We call the relevant information available to us the information set and denote this as D. Hence D_t is the information available at time t, this could just be the observations of the series but could also include any other information that has become available us.

2.2 The Dynamic Linear Model

Statistical modelling of time series processes can be based on a classes of dynamic models. The term dynamic simply relates to the changes in such processes due to the passing of time being the primary motivating force of the model. The normal dynamic linear model is a widely known subclass of these models and one which will provide the basis of much of which follows. These are often referred to as *dynamic linear models* or DLM's. For a full treatment of dynamic linear models within a Bayesian framework see West and Harrison (1997) which has become the standard reference text in this area. We will adopt the standard notation from West and Harrison (1997) to allow the reader familiar with this work to make easy comparisons.

In a time series context it is natural to consider a sequential model definition and structure, due to the sequential and structured nature of the data to be modelled. In producing the models we are seeking to produce forecast values under the model and as time evolves the information

available to us to allow updating of our forecast is received. This sequential conceptualisation focuses attention on statements about the future development of the time series conditioning on all existing information in our possession.

When we wish to start modelling at the time origin, when $t = 0$, then the only information available to us at that time is the initial information set D_0. In forecasting ahead to anytime point $t > 0$, the objective is to calculate the forecast distribution $(Y_t|D_0)$, that is the value of the series at Y_t given the initial information set, D_0. However, at time t we would usually have more information than just D_0, so at any time t, statements about the future are conditioned on all the available information D_t. Hence when forecasting ahead to time $s > t$ statements made about the the random quantities of interest, Y, at time s involve consideration of the conditional forecast distribution $(Y_s|D_t)$. Note since the information set D_t contains all relevant information at time t it by definition includes D_{t-1} and Y_t. We are interested in the future development of the series via probability distributions for $Y_{t+1}, Y_{t+2}, \ldots$ conditioning on D_t. This distribution obviously depends on the parameters determining the distribution forms and moments. To look at a one step ahead forecast our beliefs will be structured in terms of the parametric model

$$(Y_t|\phi_t, D_{t-1}),$$

where ϕ_t is a defining parameter vector at time t. These model parameters ϕ_t are the means by which the information relevant to forecasting is summarised and used in forming forecast distributions. The learning process needs to sequentially update the state of our knowledge about these parameters.

In Section 1.2 we outlined that the probabilistic representation of all uncertain knowledge is the essential element of the Bayesian approach to modelling and forecasting and hence this approach is appropriate for these models. At time t, historic information D_t is summarised through a prior distribution for future model parameters. The prior density $(\phi_t|D_{t-1})$ and the posterior density $(\phi_t|D_t)$ provide a simple and effective transfer of information on the time series process through time, we will address this more fully in Section 2.3.

Having discussed the general properties of dynamic linear models we now need to consider their specific form. We have already stated that at any time, t we can say that the observation of

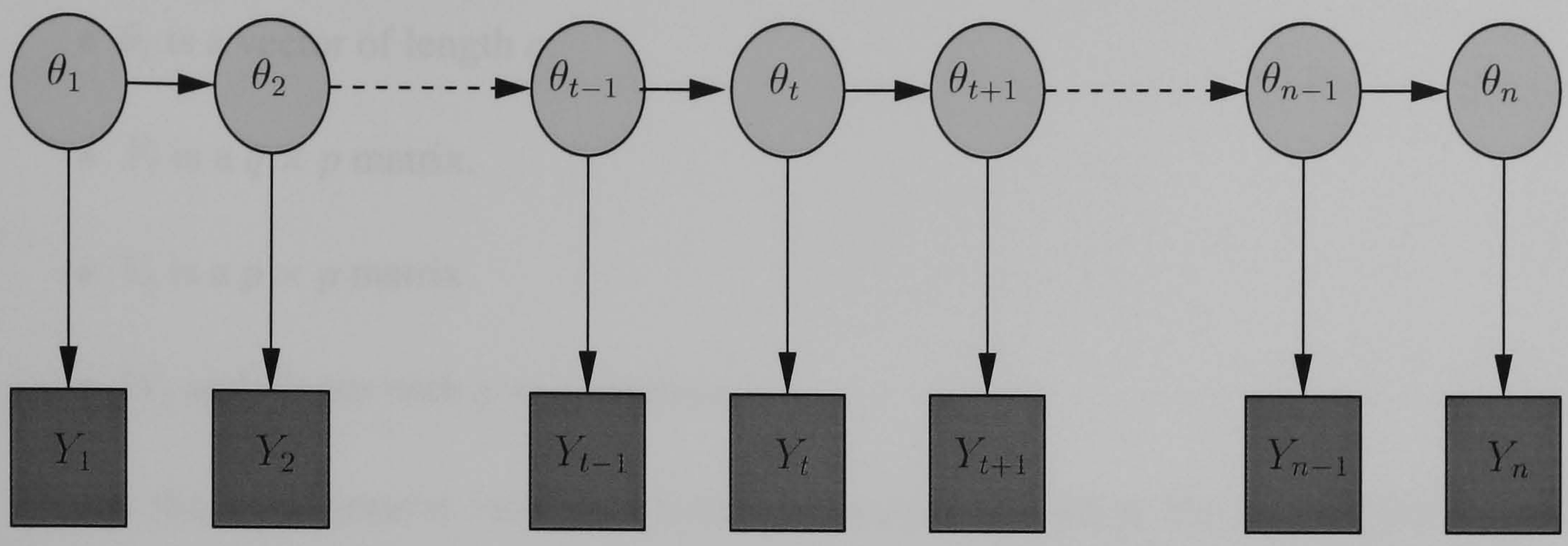


Figure 2.1: The Directed Acyclical Graph of the Dynamic Linear Model

the series can be denoted as Y_t. We let this value depend on some underlying and unknown and unobservable state denoted θ_t. This state then evolves through time according to a Markovian process, hence the state at time $t+1$ is related to the state at t which contains all necessary information about previous values of the series. To visualise this model it is perhaps easiest to consider the Directed Acyclical Graph (DAG) of the basic model structure Figure 2.1. A directed acyclical graph is a clear way of expressing the conditional independence structure of the model.

This DAG tells us a lot about this model. Firstly that the observation at any time t is a reflection of some unknown state, θ_t furthermore that all the information we require to learn about θ_t is contained in the state at the previous time point θ_{t-1}. This model is expressed succinctly in the following set of equations

$$Y_t = F_t'\theta_t + \nu_t \qquad \nu_t \sim N(0, V_t) \tag{2.1}$$

$$\theta_t = G_t\theta_{t-1} + \omega_t \quad \omega_t \sim N(0, W_t) \tag{2.2}$$

with initial information,

$$\theta_0|D_0 \sim N\left(m_0, C_0\right) \tag{2.3}$$

where:

- Y_t is a vector of length p.

- θ_t is a vector of length q.
- F_t is a $q \times p$ matrix.
- V_t is a $p \times p$ matrix.
- W_t and G_t are both $q \times q$ matrices.

This is the most general form of a family of models known as the *Normal Dynamic Linear Model*. The term normal referring to the two Gaussian noise terms. This model will provide the basis for most of the modelling conducted within this thesis. The model is split into two equations, (2.2) is know as the system or state evolution equation and provides the model dynamic. The observation equation (2.1) relates the observation to the unobservable state. Our prior beliefs about the state θ_0 given the initial information set D_0 are summarised in a Normal distribution with mean m_0 and variance C_0.These models can be either univariate or multivariate and the flexibility of model structure is derived from the form of the F_t and G_t matrices. These two terms can be time variable or time invariate depending on the model structure required. The two variance matrices V_t and W_t can also be time variable or fixed depending on the model to be considered.

Let us consider two basic dynamic linear models: the locally constant and the locally linear model which will provide the basis of much of the modelling in Chapter 3.

2.2.1 The Locally Constant Model

In the locally constant model, the state is a single parameter that can be considered to be an underlying mean. The observation Y_t is related to this underlying mean μ_t plus some stochastic noise. It is this mean which evolves through time. The model can be written in the form of (2.1) and (2.2) as

$$Y_t = \mu_t + \nu_t \qquad \nu_t \sim N(0, V)$$
$$\mu_t = \mu_{t-1} + \omega_t \quad \omega_t \sim N(0, W)$$

or as

$$Y_t = F\theta_t + \nu_t \qquad \nu_t \sim N(0, V)$$
$$\theta_t = G\theta_{t-1} + \omega_t \quad \omega_t \sim N(0, W)$$

Where in the univariate case

$$\theta_t = \mu_t \quad \text{and} \quad F_t = G_t = 1$$

and in the multivariate case with p series

$$\theta_t = \begin{pmatrix} \mu_{at} \\ \mu_{bt} \\ \vdots \end{pmatrix} \quad \text{and} \quad F_t = G_t = \mathbf{I} \quad \text{a } p \times p \text{ identity matrix}$$

θ_t is a vector of length p and V and W are both $p \times p$ matrices.

It is important to note that in this case the two variance matrices V and W are both time invariant, that is the variance of the noise components is the same at all time points.

2.2.2 The Locally Linear Model

The locally linear model is an extension of the locally constant model with the addition of a trend component τ. This trend could represent inflation or interest rates in the financial models we will be considering later in this thesis. The variance of this trend component would typically be small. This model is written as

$$Y_t = \mu_t + \nu_t \qquad \nu_t \sim N(0, V)$$
$$\mu_t = \mu_{t-1} + \tau_{t-1} + \omega_t \quad \omega_t \sim N(0, W)$$
$$\tau_t = \tau_{t-1} + \xi_t \qquad \xi_t \sim N(0, Z).$$

We can also write this in a more general DLM format

$$Y_t = F'\theta_t + \nu_t \qquad \nu_t \sim N(0, V)$$
$$\theta_t = G\theta_{t-1} + \omega_t \quad \omega_t \sim N(0, W^*)$$

Where in the univariate case

$$\theta_t = \begin{pmatrix} \mu_t \\ \tau_t \end{pmatrix} F_t = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, G_t = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \quad \text{and} \quad W^* = \begin{pmatrix} W & 0 \\ 0 & Z \end{pmatrix}$$

Or in the multivariate case for p series

$$\theta_t = \begin{pmatrix} \mu_{at} \\ \mu_{bt} \\ \vdots \\ \tau_{at} \\ \tau_{bt} \\ \vdots \end{pmatrix}, F_t = \begin{pmatrix} \mathbf{I} \\ \mathbf{0} \end{pmatrix}, G_t = \begin{pmatrix} \mathbf{I} & \mathbf{I} \\ \mathbf{0} & \mathbf{I} \end{pmatrix} \quad \text{and} \quad W^* = \begin{pmatrix} W & 0 \\ 0 & Z \end{pmatrix}$$

θ is a $q = 2p$ vector which contains two stacked vectors of length p representing μ and τ. F is a $q \times p$ matrix, where $\mathbf{I}$ is a $p \times p$ identity matrix and G is a $q \times q$ matrix. V is the $p \times p$ matrix and W^* is a $q \times q$ matrix having diagonal elements which are the $p \times p$ matrices W and Z which are the variance matrices for the underlying mean and trend components respectively. These two models provide the basis of the models used within this thesis. Later in Chapter 5 we will consider models where the variance matrices are allowed to evolve over time.

2.3 Kalman Filtering and Smoothing

Once we have a model the next natural step is to try and make some inferences on this model. The initial method we will consider for making inference on this model are Kalman filtering and smoothing techniques. These techniques will provide the basis for much of the work which is to follow.

In this section we will consider the following general DLM

$$Y_t = F_t'\theta_t + \nu_t \qquad \nu_t \sim N(0, V_t)$$
$$\theta_t = G_t\theta_{t-1} + \omega_t \quad \omega_t \sim N(0, W_t)$$

with initial information

$$\theta_0|D_0 \sim N(m_0, C_0)$$

where $Y_t, \theta_t, \nu_t, \omega_t$ and m_0 are vectors and F_t, G_t, V_t, W_t and C_0 are matrices. The exact dimensions of these would depend on the specific model structure chosen.

In Kalman filtering and smoothing we are looking to obtain values for the states θ_t given the data Y_t and assuming we know V_t and W_t. We set F_t and G_t according to the model structure we have chosen.

2.3.1 The Filter

Let us first consider the Kalman filter. Here we are looking to derive an expression for the state θ_t given all the available information at time t, that is the information set D_t.

In Section 1.2 we discussed that the Bayesian paradigm was appropriate for use on complex models so let us follow this here. The posterior for the state vector θ at time $t-1$ given the information set D_{t-1} can be summarised as a normal distribution with some mean vector, m_{t-1} and variance matrix, C_{t-1}, that is

$$\theta_{t-1}|D_{t-1} \sim N(m_{t-1}, C_{t-1}).$$

Prior information for the state vector θ at time t can be summarised as a normal distribution with mean vector, a_t and variance matrix, R_t

$$\theta_t|D_{t-1} \sim N(a_t, R_t)$$

where

$$a_t = G_t m_{t-1} \qquad \text{and} \qquad R_t = G_t C_{t-1} G_t{}' + W_t$$

These follow directly from equation (2.2) of our model.

The first thing we might wish to do is forecast one time point ahead, that is forecast values for the series Y_t given the information set D_{t-1}. This is simply a matter of taking the expectation of a linear combination of the normally distributed variables, $\theta_t|D_{t-1}$ and ν_t using equation (2.1). Hence

$$\begin{aligned}
\mathrm{E}\left(Y_t|D_{t-1}\right) &= \mathrm{E}\left(F_t{}'\theta_t + \nu_t|D_{t-1}\right) \\
&= \mathrm{E}\left(F_t{}'\theta_t|D_{t-1}\right) + \mathrm{E}\left(\nu_t|D_{t-1}\right) \\
&= F_t{}'\mathrm{E}\left(\theta_t|D_{t-1}\right) + \mathrm{E}\left(\nu_t|D_{t-1}\right) \\
&= F_t{}'a_t \equiv f_t \\
\mathrm{Var}\left(Y_t|D_{t-1}\right) &= \mathrm{Var}\left(F_t{}'\theta_t + \nu_t|D_{t-1}\right) \\
&= \mathrm{Var}\left(F_t{}'\theta_t|D_{t-1}\right) + \mathrm{Var}\left(\nu_t|D_{t-1}\right) \\
&= F_t{}'\mathrm{Var}\left(\theta_t|D_{t-1}\right) F_t + \mathrm{Var}\left(\nu_t|D_{t-1}\right) \\
&= F_t{}'R_t F_t + V_t \equiv Q_t
\end{aligned}$$

Hence the one-step ahead forecast distribution for the series Y_t given the information set D_{t-1} is simply

$$Y_t|D_{t-1} \sim N(f_t, Q_t)$$

Hence if we know the vector m_{t-1} and matrix C_{t-1} we can predict values for the series at Y_t. We have stated that for the moment that we know V_t and W_t the one thing we do not know at time t is the state θ_t. The model likelihood is the conditional forecast distribution evaluated at

the observed time and has normal form.

$$\mathrm{L}\left(\theta_t|Y_t, V_t\right) \sim N(F_t'\theta_t, V_t).$$

To get the posterior for the state, θ_t given the D_t where $D_t = \{Y_t, D_{t-1}\}$ we can combine the prior with the likelihood using Bayes Theorem, (1.1).

$$\pi\left(\theta_t|D_t\right) \propto \pi\left(\theta_t|D_{t-1}\right) \mathrm{L}\left(\theta_t|Y_t, V_t\right)$$

This is simply a product of two normal densities, hence the posterior is simply

$$\pi\left(\theta_t|D_t\right) \propto \exp\left(-\frac{1}{2}\left(\left(\left(Y_t - F_t'\theta_t\right)'{V_t}^{-1}\left(Y_t - F_t'\theta_t\right)\right) + \left(\left(\theta_t - a_t\right)'{R_t}^{-1}\left(\theta_t - a_t\right)\right)\right)\right)$$

taking logs and multiplying by -2

$$-2\log\pi\left(\theta_t|D_t\right) = \left(\left(Y_t - F_t'\theta_t\right)'{V_t}^{-1}\left(Y_t - F_t'\theta_t\right)\right) + \left(\left(\theta_t - a_t\right)'{R_t}^{-1}\left(\theta_t - a_t\right)\right) + k$$

where k is a constant term not involving θ_t. Rearranging we get

$$-2\log\pi\left(\theta_t|D_t\right) = \theta_t'\left(F_t'{V_t}^{-1}F_t + {R_t}^{-1}\right)\theta_t - 2\theta_t'\left(F_t'{V_t}^{-1}Y_t + {R_t}^{-1}a_t\right) + k.$$

Hence completing the square and dividing through by 2 and exponentiating we get

$$\pi\left(\theta_t|D_t\right) \propto \exp\left(-\frac{1}{2}\left(\left(\theta_t - m_t\right)'{C_t}^{-1}\left(\theta_t - m_t\right)\right)\right)$$

where

$$\begin{aligned} {C_t}^{-1} &= \left(F_t'{V_t}^{-1}F_t + {R_t}^{-1}\right) \\ m_t &= C_t\left(F_t'{V_t}^{-1}Y_t + {R_t}^{-1}a_t\right). \end{aligned}$$

Hence the posterior for the state, θ_t given all available information set D_t is

$$\theta_t|D_t \sim N\left(m_t, C_t\right).$$

The above statement is often referred to as the *Kalman Filter*.

The Kalman Filter allows us to say something about the expectation and variance of the state, θ_t given the information set D_t. As we will see in Chapter 3 this is useful for making inferences about parameters of the model which authors commonly currently assume are known.

The Kalman filter equation does not rely upon a Bayesian paradigm and can be derived using standard normal theory as we will show below. Given

$$\theta_t = G_t\theta_{t-1} + \omega_t \quad \omega_t \sim N(0, W_t)$$

the value of the state at $t-1$ given the information set D_{t-1} can in generally said to be as follows

$$\theta_{t-1}|D_{t-1} \sim N\left(m_{t-1}, C_{t-1}\right)$$

Again using the system equation, (2.2)

$$\theta_t|D_{t-1} \sim N\left(a_t, R_t\right)$$

Where

$$a_t = G_t m_{t-1} \qquad \text{and} \qquad R_t = G_t C_{t-1} G_t{}' + W_t$$

From the observation equation, (2.1) as shown above

$$Y_t|D_{t-1} \sim N\left(f_t, Q_t\right)$$

where

$$f_t = F_t' a_t \quad \text{and} \quad Q_t = F_t' R_t F_t + V_t.$$

Now the vector $(Y, \theta)'$ is jointly normal. To write this joint distribution we require the covariances

$$\begin{aligned} \text{Cov}\left(\theta_t, Y_t | D_{t-1}\right) &= \text{Cov}\left(\theta_t, F_t'\theta_t + \nu_t | D_{t-1}\right) \\ &= \text{Var}\left(\theta_t | D_{t-1}\right) F_t + 0 \\ &= R_t F_t. \end{aligned}$$

Hence the conditional joint distribution of $(Y_t, \theta_t | D_{t-1})$ is

$$\left(\begin{pmatrix} Y_t \\ \theta_t \end{pmatrix} \middle| D_{t-1} \right) \sim N\left(\begin{pmatrix} f_t \\ a_t \end{pmatrix}, \begin{pmatrix} Q_t & F_t' R_t \\ R_t F_t & R_t \end{pmatrix} \right).$$

Using standard results for multivariate normal distributions we can find the conditional distribution for θ_t given Y_t, D_{t-1}

$$\theta_t | D_t \sim N(m_t, C_t)$$

where

$$\begin{aligned} m_t &= a_t + R_t F_t {Q_t}^{-1} \left(Y_t - f_t\right) \\ C_t &= R_t - R_t F_t {Q_t}^{-1} F_t' R_t. \end{aligned}$$

This derivation shows the strength of this model format, allowing us to make inference about the state relying only on the conditional independence structure as laid out in the DAG Figure 2.1. In Section 1.3 we introduced Bayes Linear inferences and discussed that this has advantages when considering complex models as it overcomes the requirement for specifying the full joint probability structure. In this case we can write this relatively easily however we can still apply Bayes Linear methodology to this model and derive the Kalman filter equations.

Using our general DLM as defined by (2.2) and (2.1) then as above, then we can write the expectation and variance for the state at time $t-1$ given the information set D_{t-1}

$$\mathrm{E}_{D_{t-1}}(\theta_{t-1}) = m_{t-1}$$
$$\mathrm{Var}_{D_{t-1}}(\theta_{t-1}) = C_{t-1}.$$

Then at time t these are

$$\mathrm{E}_{D_t}(\theta_t) = m_t$$
$$\mathrm{Var}_{D_t}(\theta_t) = C_t.$$

Using the state equation (2.2) then the adjusted expectation and variance for θ_t given the information set D_{t-1} are

$$\mathrm{E}_{D_{t-1}}(\theta_t) = \mathrm{E}_{D_{t-1}}(G_t\theta_{t-1} + \omega_t) = G_t\mathrm{E}_{D_{t-1}}(\theta_{t-1}) = G_t m_{t-1} = a_t$$
$$\mathrm{Var}_{D_{t-1}}(\theta_t) = \mathrm{Var}_{D_{t-1}}(G_t\theta_{t-1} + \omega_t) = G_t\mathrm{Var}_{D_{t-1}}(\theta_{t-1})G_t' + W = G_tC_{t-1}G_t' + W = R_t.$$

The expectation and variance for the series at time t given the information set D_{t-1} are

$$\mathrm{E}_{D_{t-1}}(Y_t) = \mathrm{E}_{D_{t-1}}(F_t'\theta_t + \nu_t) = F_t'\mathrm{E}_{D_{t-1}}(\theta_t) = F_t'a_t = f_t$$
$$\mathrm{Var}_{D_{t-1}}(Y_t) = \mathrm{Var}_{D_{t-1}}(F_t'\theta_t + \nu_t) = F_t'\mathrm{Var}_{D_{t-1}}(\theta_t)F_t + V_t = F_t'R_tF_t + V = Q_t$$
$$\mathrm{Cov}_{D_{t-1}}(\theta_t, Y_t) = \mathrm{Cov}_{D_{t-1}}(\theta_t, F_t'\theta_t + \nu_t) = \mathrm{Var}_{D_{t-1}}(\theta_t)F_t = R_tF_t.$$

We require the state, θ_t, given the current information set $D_t = \{Y_t, D_{t-1}\}$, the general form of the expectation and variance of which are given above.

Generally the Bayes Linear Adjusted Expectation as discussed in Chapter 1 is given by (1.2). Adjusting for knowing Y_t gives

$$\mathrm{E}_{Y_t}(\theta_t) = \mathrm{E}_{D_{t-1}}(\theta_t) + \mathrm{Cov}_{D_{t-1}}(\theta_t, Y_t)\mathrm{Var}_{D_{t-1}}(Y_t)^{-1}[Y_t - \mathrm{E}_{D_{t-1}}(Y_t)]$$
$$\mathrm{E}_{Y_t}(\theta_t) = a_t + R_tF_tQ_t^{-1}[Y_t - f_t] = m_t.$$

Generally Bayes Linear Adjusted Variance is given by (1.3).

So adjusting for Y_t gives

$$\mathrm{Var}_{Y_t}(\theta_t) = \mathrm{Var}_{D_{t-1}}(\theta_t) - \mathrm{Cov}_{D_{t-1}}(\theta_t, Y_t)\mathrm{Var}_{D_{t-1}}(Y_t)^{-1}\mathrm{Cov}_{D_{t-1}}(\theta_t, Y)'$$

$$\mathrm{Var}_{Y_t}(\theta_t) = R_t - (R_t F_t)Q_t^{-1}(R_t F_t)' = C_t$$

Once again we have derived the Kalman filter expression.

The above serves to illustrate the flexibility of the dynamic linear model format and Kalman filtering. The Bayes Linear derivation is succinct, however we will require the distributional form when we consider simulation smoothing in Section 2.3.3 and so we will rely upon the normal theory justification of the filter and subsequent smoothers.

To summarise the Kalman filter consider the following algorithm.

1. Set initial values for the expectation and variance of the state at time $t = 0$, m_0 and C_0.

2. Calculate the expectation and variance of the state given the information set D_1, that is m_1 and C_1.

3. sequentially move through the available data, Y_t, for $t = (2, \ldots, n)$ calculating values for the expectation and variance of the state, θ_t.

This illustrates the simplicity of Kalman filtering, which simply sequentially calculates the required values.

2.3.2 The Smoother

In section Section 2.3.1 we have produced filtered values for $\mathrm{E}(\theta_t|D_t)$ and $\mathrm{Var}(\theta_t|D_t)$, however we have more information than this as we know the values of the series Y_t upto its last know point, n, i.e $(Y_{t+1}, Y_{t+2}, \ldots, Y_n)$. It would be sensible and likely to produce more accurate estimates for the expectation and variance of the state if we could include all this available information. That is using the expectation vectors and variance matrices produced by the Kalman filter re-estimate these to produce expectation vectors and variance matrices for the state given all the data and all the θ_t's. This process is know as smoothing.

In the smoother everything is conditional upon D where $D = \{D_t, D_t^*\}$ and $D_t^* = (Y_{t+1}, Y_{t+2}, \ldots, Y_n)$. In general if:

$$\theta_t | D_t \sim N(m_t, C_t)$$

then:

$$\theta_t | D \sim N(m_t{}^*, C_t{}^*).$$

The system equation (2.2) gives us the distribution of $\theta_t | \theta_{t-1}, D_{t-1} \sim N(G_t\theta_{t-1}, W_t)$ and we can therefore write its expectation, variance and covariance

$$\mathrm{E}(\theta_t | D_{t-1}) = \mathrm{E}(G_t\theta_{t-1} + \omega_t | D_{t-1}) = G_t\mathrm{E}(\theta_{t-1} | D_{t-1}) = G_t m_{t-1} = a_t$$

$$\mathrm{Var}(\theta_t | D_{t-1}) = \mathrm{Var}(G_t\theta_{t-1} + \omega_t | D_{t-1}) = G_t\mathrm{Var}(\theta_{t-1} | D_{t-1})G_t{}' + W_t = G_tC_{t-1}G_t{}' + W_t = R_t$$

$$\mathrm{Cov}(\theta_{t-1}, \theta_t | D_{t-1}) = \mathrm{Cov}(\theta_{t-1}, G_t\theta_{t-1} + \omega_t | D_{t-1}) = \mathrm{Var}(\theta_{t-1} | D_{t-1})G_t{}' = C_{t-1}G_t{}'.$$

Hence the conditional joint distribution of $(\theta_t, \theta_{t-1} | D_{t-1})$ is

$$\left(\begin{pmatrix} \theta_t \\ \theta_{t-1} \end{pmatrix} \middle| D_{t-1} \right) \sim N\left(\begin{pmatrix} a_t \\ m_{t-1} \end{pmatrix}, \begin{pmatrix} R_t & C_{t-1}G_t{}' \\ G_t{}'C_{t-1} & C_{t-1} \end{pmatrix} \right).$$

Using standard results for multivariate normal distributions we can find the conditional distribution for θ_{t-1} given θ_t

$$\theta_{t-1} | \theta_t, D_{t-1} \sim N(h_t, H_t) \tag{2.4}$$

where

$$h_t = m_{t-1} + C_{t-1}G_t{}'R_t{}^{-1}(\theta_t - a_t) \tag{2.5}$$

$$H_t = C_{t-1} + G_tC_{t-1}R_t{}^{-1}C_{t-1}G_t{}' \tag{2.6}$$

Now because we have

$$f(\theta_0, \ldots, \theta_n|D) = f(\theta_n|D_n) f(\theta_{n-1}|\theta_n, D_{n-1}) \ldots f(\theta_0|\theta_1, D_0),$$

and because of the Markov structure of the the DLM we have that,

$$f(\theta_{t-1}|\theta_t, D) \propto f(\theta_{t-1}|\theta_t, D_{t-1})$$

which we have given in (2.4). Now,

$$f(\theta_{t-1}|D) = \int f(\theta_{t-1}|\theta_t, D) f(\theta_t|D) d\theta_t$$

i.e. $f(\theta_{t-1}|D)$ is the expectation of $f(\theta_{t-1}|\theta_t, D_{t-1})$ with respect to $f(\theta_t|D)$. Let

$$\theta_{t-1}|D \sim N({m_{t-1}}^*, {C_{t-1}}^*).$$

Then

$$\begin{aligned}
{m_{t-1}}^* &= \mathrm{E}\left(\mathrm{E}\left(\theta_{t-1}|\theta_t, D_{t-1}\right)\right) \\
&= \mathrm{E}(h_t) \\
&= m_{t-1} + C_{t-1} G_t{}' {R_t}^{-1} (m_t - a_t) \qquad (2.7) \\
{C_{t-1}}^* &= \mathrm{E}\left(\mathrm{Var}\left(\theta_{t-1}|\theta_t, D_{t-1}\right)\right) + \mathrm{Var}\left(\mathrm{E}\left(\theta_{t-1}|\theta_t, D_{t-1}\right)\right) \\
&= \mathrm{E}(H_t) + \mathrm{Var}(h_t) \\
&= C_{t-1} + G_t C_{t-1} {R_t}^{-1} C_{t-1} G_t{}' + C_{t-1} G_t{}' {R_t}^{-1} (C_t) {R_t}^{-1} C_{t-1} G_t{}' \\
&= C_{t-1} + G_t C_{t-1} {R_t}^{-1} [R_t - C_t] {R_t}^{-1} C_{t-1} G_t{}'. \qquad (2.8)
\end{aligned}$$

The covariance is simply

$$\mathrm{Cov}(\theta_{t-1}, \theta_t|D) = C_{t-1} G_t{}' {R_t}^{-1} C_t.$$

Again to summarise the smoother let us look at the algorithm for using it.

1. Set initial values for the expectation and variance of the state at time $t = 0$, m_0 and C_0.

2. Run the Kalman filter producing expectation vectors and variance matrices for the states, θ_t conditioned on the information set D_t.

3. At $t = n - 1$ using the smoothing equations (2.7) and (2.8), calculate the expectation vectors and variance matrices conditioning on all states and all of the data. We can also keep track of the covariances as we will require these when we consider some forms of parameter estimation in Chapter 3.

4. Repeat sequentially until $\theta_1|D$ is calculated.

This again illustrates the relative simplicity of Kalman filtering and smoothing techniques as they rely simply on sequential calculation.

2.3.3 The Simulation Smoother

There is a second option open to us when it comes to smoothing, this second option will be important when we consider Monte Carlo Markov Chain methods of parameter estimation in Chapter 3. This second method of smoothing is known as simulation smoothing as we simulate values for θ_t for $t = 1, \ldots, n$.

We will wish to sample the set of state vectors $\Theta_n = \theta_0, \theta_1, \ldots, \theta_n$ from the multivariate normal posterior $p(\Theta_n|D_n)$. We can exploit the Markov properties of the system equation of the DLM to write

$$p(\Theta_n|D_n) = p(\theta_n|D_n)p(\theta_{n-1}|\theta_n, D_{n-1}) \ldots p(\theta_1|\theta_2, D_1)p(\theta_0|\theta_1, D_0)$$

As a result we can sample the whole of Θ_n by sequentially simulating the individual state vectors $\theta_n, \theta_{n-1}, \ldots, \theta_0$. This can be done as follows

1. Sample θ_n from $\theta_n|D_n \sim N(m_n, C_n)$, the values for which we have obtained from the Kalman Filter.

2. Then for each $t = n - 1, n - 2, \ldots, 1, 0$ sample θ_{t-1} from $\theta_{t-1}|\theta_t, D_{t-1}$ where the conditioning value for θ_t is the value just sampled.

In the previous section we have already shown the distribution for $\theta_{t-1}|\theta_t, D_{t-1}$ with equation (2.4)

$$\theta_{t-1}|\theta_t, D_{t-1} \sim N(h_t, H_t)$$

where

$$h_t = m_{t-1} + C_{t-1}G_t{}'R_t{}^{-1}(\theta_t - a_t)$$
$$H_t = C_{t-1} + G_tC_{t-1}R_t{}^{-1}C_{t-1}G_t{}'.$$

We already have θ_t as we sample this in the previous iteration. Once again like the smoother this algorithm consists of simple sequential calculations.

2.4 Programming Aspects

The above algorithms can be incorporated within computer programmes to allow us to make full use of their capabilities. A class was developed in **sather** to perform these functions. This class is called KALMAN. The modular library kalman.sa contains routines to perform the Kalman filter as well as either the standard smoother or the simulation smoother, reference for this can be found in Appendix A.

Let us initially consider the following locally constant dynamic linear model,

$$Y_t = \mu_t + \nu_t \quad \nu_t \sim N(0, 225)$$
$$\mu_t = \mu_{t-1} + \omega_t \quad \omega_t \sim N(0, 49)$$

We can simulate 1000 data points from this model after setting an initial value for the state and also keep the values of the state at each time point.

We can run our Kalman filter on the simulated series and keep the values for the expectation and variance of the state at each time point which this calculates. Figure 2.2 shows a subsection of the true states, $(\theta_{600}, \ldots, \theta_{650})$ as a black line. The expectation of the state is plotted as a red line. The 95% confidence interval for the expectation is calculated using the variance of the states

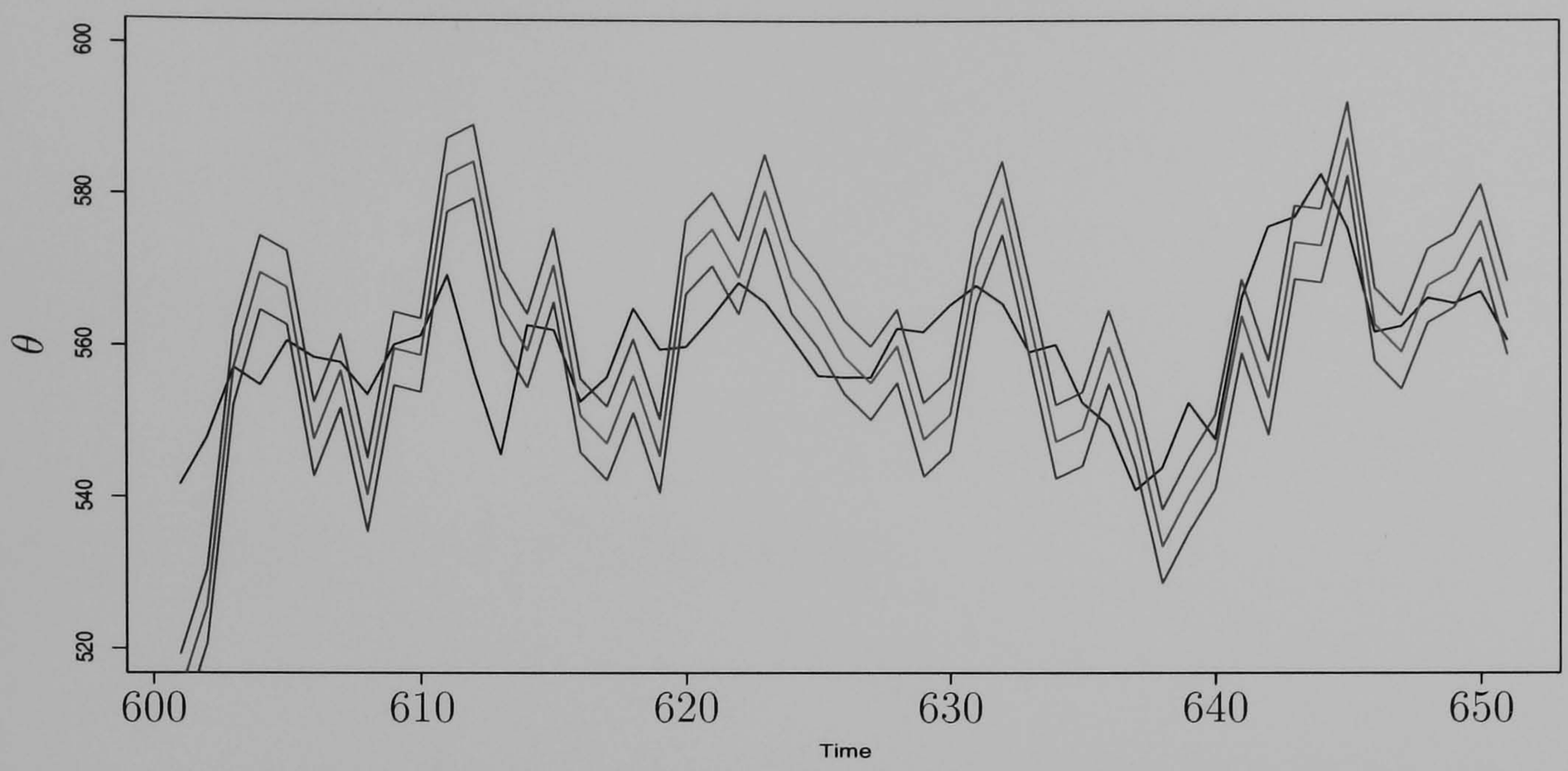


Figure 2.2: Graph of true states and filter estimates.

from the Kalman filter and plotted as blue line on the figure. We can see that the expectation of the states from the Kalman filter is close to the true value. The Kalman filter is sometimes slow to react to changes is the gradient of the slope of the state and perhaps exaggerates variation. Figure 2.3 shows the same results for the smoother. These results are closer to the true values with less exaggeration. This demonstrates the advantage of incorporating all available data into our model.

In Chapter 3 we will go on to look at how we can make use of the Kalman filtering and smoothing techniques we have outlined in this chapter.

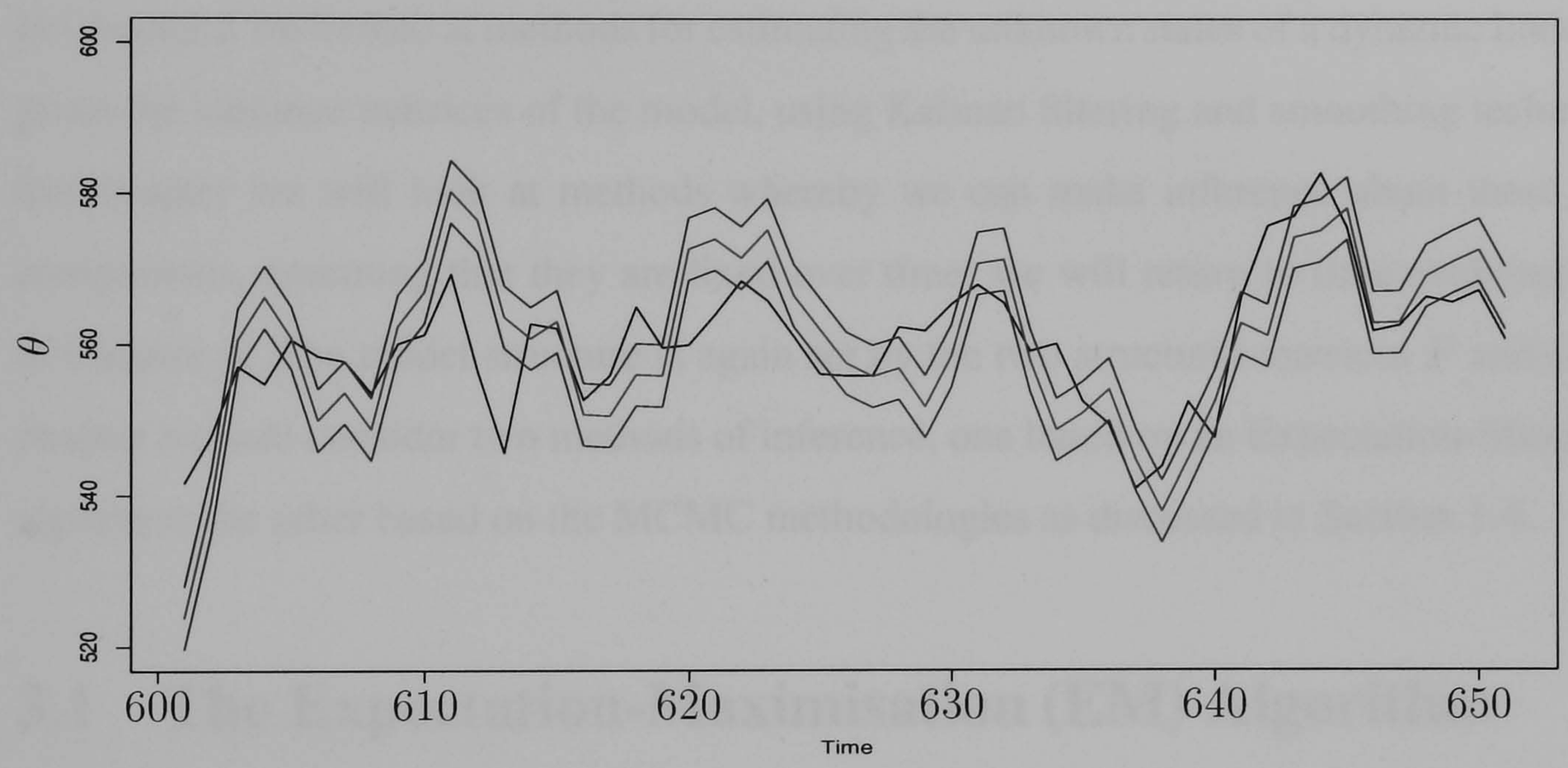


Figure 2.3: Graph of true states and smoother estimates.

Chapter 3

Parameter Estimation

In Chapter 2 we looked at methods for estimating the unknown states of a dynamic linear model, given the variance matrices of the model, using Kalman filtering and smoothing techniques. In this chapter we will look at methods whereby we can make inference about these variance components, assuming that they are fixed over time; we will return to time evolving variance in Chapter 4. The model structure is again set by the two structural matrices F and G. In this chapter we will consider two methods of inference, one based on an Expectation-Maximisation algorithm the other based on the MCMC methodologies as discussed in Section 1.4.

3.1 The Expectation-Maximisation (EM) Algorithm

The EM Algorithm is a two stage algorithm, where we take expectations of the log posterior density (the E-step) and then maximise these (the M-step) in order to generate an estimate of the posterior mode. These steps are repeated until convergence.

Consider the dynamic linear model:

$$Y_t = F'\theta_t + \nu_t \qquad \text{The Observation Equation}$$

$$\theta_t = G\theta_{t-1} + \omega_t \qquad \text{The System Equation}$$

where,

$$\nu_t \sim N(0, V) \qquad \text{and} \qquad \omega_t \sim N(0, W).$$

Then in conjunction with Kalman filtering and smoothing we can enact an EM algorithm to produce posterior modes for the precisions of the two variance parameters V and W in the M-step and posterior expectations ${\nu_t}^2$ and ${\omega_t}^2$ in the E-step. The dimension of the state vector and variance and structural matrices depends on whether we are considering univariate or multivariate series or locally constant or locally linear models as discussed in Section 2.2.1 and Section 2.2.2. For a fuller treatment of the EM algorithm in this context see Koopman (1993).

3.1.1 The Univariate EM Algorithm

In the univariate case the series of interest Y_t is a single series, θ_t is a vector of length 1 or 2 depending whether we are looking at locally constant or locally linear models. Let us initially consider a locally constant model where $\theta_t = \mu_t$.

In this case we are looking to produce estimates for V and W, by finding the posterior modes of the precisions $R = V^{-1}$ and $S = W^{-1}$ and then inverting these to produce estimates for V and W.

Expectation Step (E-step)

Let us first look at the Expectation step. This is a Kalman Filter-Smoother, which is used to generate a filtered and smoothed estimate of the system parameter θ and hence the expectations of ${\nu_t}^2|D$ and ${\omega_t}^2|D$ which will be required in the M-step. In order to be able to run the Kalman filter we require values for the variances V and W. The Kalman filter and smoother are discussed in full detail in Chapter 2. The filter assumes a prior belief of

$$\theta_0 \sim N(m_0, C_0)$$

and requires variance specifications V and W, where:

$$\mathrm{Var}(\nu_t) = V \qquad \text{and} \qquad \mathrm{Var}(\omega_t) = W.$$

The Kalman Filter-Smoother is itself a two stage process, the initial sweep uses these initial estimates and runs through the data sequentially to generate new estimates for m_t and C_t where:

$$\theta_t|D_t \sim N(m_t, C_t) \qquad \text{and} \qquad D_t = (Y_1, Y_2, \ldots, Y_t).$$

The second stage uses these estimates and the data to smooth backwards generating new smoothed estimates for m_t and C_t which incorporate both the original estimates and all the data. Hence,

$$\theta_t|D \sim N({m_t}^*, {C_t}^*) \qquad \text{and} \qquad D = \{D_t, {D_t}^*\} \quad {D_t}^* = (Y_t, Y_{t+2}, \ldots, Y_n)$$

Thus the expectations required by the M-step can be calculated as follows.

$$\mathrm{E}({\nu_t}^2|D) = \mathrm{Var}(\nu_t|D) + \mathrm{E}(\nu_t|D)^2$$

where

$$\mathrm{E}(\nu_t|D) = \mathrm{E}(Y_t - F'\theta_t|D) = Y_t - F'\mathrm{E}(\theta_t|D) = Y_t - F'{m_t}^*$$
$$\mathrm{Var}(\nu_t|D) = \mathrm{Var}(Y_t - F'\theta_t|D) = \mathrm{Var}(F'\theta_t|D) = F'{C_t}^*F$$

and for ω's

$$E(\omega_t^2|D) = \mathrm{Var}(\omega_t|D) + E(\omega_t|D)^2$$

where:

$$
\begin{aligned}
E(\omega_t|D) &= \mathrm{E}(\theta_t - G\theta_{t-1}|D) = \mathrm{E}(\theta_t|D) - \mathrm{E}(G\theta_{t-1}|D) = {m_t}^* - G{m_{t-1}}^* \\
\mathrm{Var}(\omega_t|D) &= \mathrm{Var}(\theta_t - G\theta_{t-1}|D) = \mathrm{Var}(\theta_t|D) + \mathrm{Var}(G\theta_{t-1}|D) - 2\mathrm{Cov}(\theta_t, G\theta_{t-1}|D) \\
&= {C_t}^* + G{C_{t-1}}^*G' - 2G\mathrm{Cov}(\theta_t, \theta_{t-1}|D)G'.
\end{aligned}
$$

All these values are easily calculated from the expectation and variance of the state given all the states and data as generated by the Kalman filter and smoother. The **sather** class KALMAN has been written to include modules which perform these calculations.

The Maximisation Step (M-step)

The maximisation step provides us with the posterior modes for R and S and relies on the expectations generated in the E-step. We place conjugate gamma priors on the precisions of the two variance terms, hence

$$V^{-1} = R \sim \Gamma(a, b) \qquad \text{and} \qquad W^{-1} = S \sim \Gamma(c, d)$$

We can combine this with the Likelihood using Bayes Theorem (1.1) to obtain the posterior for R given $\nu = (\nu_1, \ldots, \nu_n)$ and S given $\omega = (\omega_1, \ldots, \omega_n)$. Since we are using conjugate prior distributions and the fact that R and S are conditionally independent of one another given ν and ω this follows standard results and we write the conditional distributions,

$$R|\nu \sim \Gamma(a + \frac{n}{2}, b + \frac{1}{2}\sum_{t=1}^{n} \nu_t^2)$$

$$S|\omega \sim \Gamma(c + \frac{n}{2}, d + \frac{1}{2}\sum_{t=1}^{n} \omega_t^2).$$

If we look first at the R parameter, in this maximisation step we are looking to find the posterior mode of R and by inversion an estimate of V.

$$\pi(r|\nu) \propto r^{a+\frac{n}{2}-1} \exp\left\{-r(b + \frac{1}{2}\sum_{t=1}^{n} \nu_t^2)\right\}$$

Taking logs:

$$\log \pi(r|\nu) = const + (a + \frac{n}{2} - 1)\log r - r(b + \frac{1}{2}\sum_{t=1}^{n} \nu_t^2)$$

Then taking posterior Expectations of ν with respect to the old estimates which we found in the E-step:

$$= const + (a + \frac{n}{2} - 1)\log r - r(b + \frac{1}{2}\sum_{t=1}^{n} E(\nu_t^2|D))$$

Differentiating with respect to r, equating to zero to find the mode and substituting in the results from the E-step, which are easily calculated in **sather** , gives us,

$$\hat{r} = \frac{(a + \frac{n}{2} - 1)}{(b + \frac{1}{2}(\sum_{t=1}^{n} \mathrm{Var}(\nu_t|D) + \sum_{t=1}^{n} \mathrm{E}(\nu_t|D)^2)}. \tag{3.1}$$

This gives us a new estimate for the posterior mode of R and by inversion an estimate of V.

The results for S and W follow similarly.

$$\pi(s|\omega) \propto s^{c+\frac{n}{2}-1} \exp\left\{-s(d + \frac{1}{2}\sum_{t=1}^{n} \omega_t^2)\right\}$$

Taking logs:

$$\log \pi(s|\omega) = const + (c + \frac{n}{2} - 1)\log s - s(d + \frac{1}{2}\sum_{t=1}^{n} \omega_t^2)$$

Taking posterior expectations of ω with respect to the data and the old estimates which we found in the E-step,

$$= const + (c + \frac{n}{2} - 1)\log s - s(d + \frac{1}{2}\sum_{t=1}^{n} E(\omega_t^2|D))$$

Differentiating with respect to to s , equating to zero to find the mode and substituting in the

results from the E-step which are easily calculated in the **sather** programs,

$$\hat{s} = \frac{(c + \frac{n}{2} - 1)}{(d + \frac{1}{2}(\sum_{t=1}^{n} \mathrm{Var}(\theta_t|D) + \sum_{t=1}^{n} \mathrm{Var}(\theta_{t-1}|D) - 2\sum_{t=1}^{n} \mathrm{Cov}(\theta_t, \theta_{t-1}|D) + \sum_{t=1}^{n} \mathrm{E}(\omega_t|D)^2))} \tag{3.2}$$

We now have estimates for the modes of the parameters R and S which can be inverted to gives estimates for V and W. These can be used in the Kalman Filter to re-estimate the system parameter θ, and hence also provide new estimates for ν and ω. This is the M-step of the EM algorithm.

To clarify this consider the following algorithm

The Algorithm

1. Run the Kalman filter-smoother with some initial values for V and W.

2. Calculate the necessary posterior expectations and variances, this is done online in the **sather** class KALMAN.

3. Calculate the posterior modes from (3.1) and (3.2) and invert to get estimates for V and W to use in the Kalman filter smoother.

4. Re-run the Kalman filter and smoother using these new values for V and W.

5. Repeat steps 2 to 4 until the algorithm converges. In this case convergence is regarded to be when the difference between subsequent values for V and W is sufficiently small. (The value of small is set by the modeller depending on the prior beliefs as to the scale of the two variance components.)

The locally linear model is a simple extension of this algorithm, where $\theta = (\mu_t, \tau_t)'$. For the locally linear model we have a variance matrix W^* which contains two elements W and Z on its diagonal as outlined in Section 2.2.2. In this case we place independent gamma priors on the precisions of both of these terms, i.e.

$$W^{-1} = S \sim \Gamma(c, d) \quad \text{and} \quad Z^{-1} = T \sim \Gamma(e, f)$$

The posterior mode for T follow as before for S calculating the conditional expectations and variances using the appropriate elements of the θ_t vector.

3.1.2 Examples

Let us again as in Section 2.4 consider the following locally constant model,

$$Y_t = \mu_t + \nu_t \quad \nu_t \sim N(0, 225)$$
$$\mu_t = \mu_{t-1} + \omega_t \quad \omega_t \sim N(0, 49)$$

We simulated 1000 points from this model after setting an initial value for μ_0. We wrote a **sather** program to enact the above EM algorithm using the KALMAN class as discussed in Section 2.4.

We set initial values of 10 for both$V$ and W and choose a flat $\Gamma(1, 0.0001)$ prior for both variance components. We can then run the **sather** program *uemlc.sa*, a copy of which can be found in Appendix A. Figure 3.1 shows the results of running this program, the algorithm can be seen to be converging on a value for V of 236.1760 and for W of 59.6314. These values are reasonably close to the true values for these two components of 225 and 49 respectively.

We can also look at a locally linear model, adding a trend term to the above model we get

$$Y_t = \mu_t + \nu_t \quad \nu_t \sim N(0, 225)$$
$$\mu_t = \mu_{t-1} + \tau_{t-1} + \omega_t \quad \omega_t \sim N(0, 49)$$
$$\tau_t = \tau_{t-1} + \xi_t \quad \xi_t \sim N(0, 0.01).$$

Again we simulated 1000 data points from this model. We wrote a **sather** program to enact the above EM algorithm using the KALMAN class as discussed in Section 2.4.

We set initial values of 1000 for V and 10 for both W and Z and choose a flat $\Gamma(1, 0.0001)$ prior for all the variance components. We then ran the **sather** program *uemll.sa*, a copy of which can be found in Appendix A. Figure 3.2 shows the results of running this program, the algorithm can be seen to be converging on a values for V of 224.992, for W of 52.3364 and for Z of 0.000139227. These values are again reasoonably close to the true values for these two

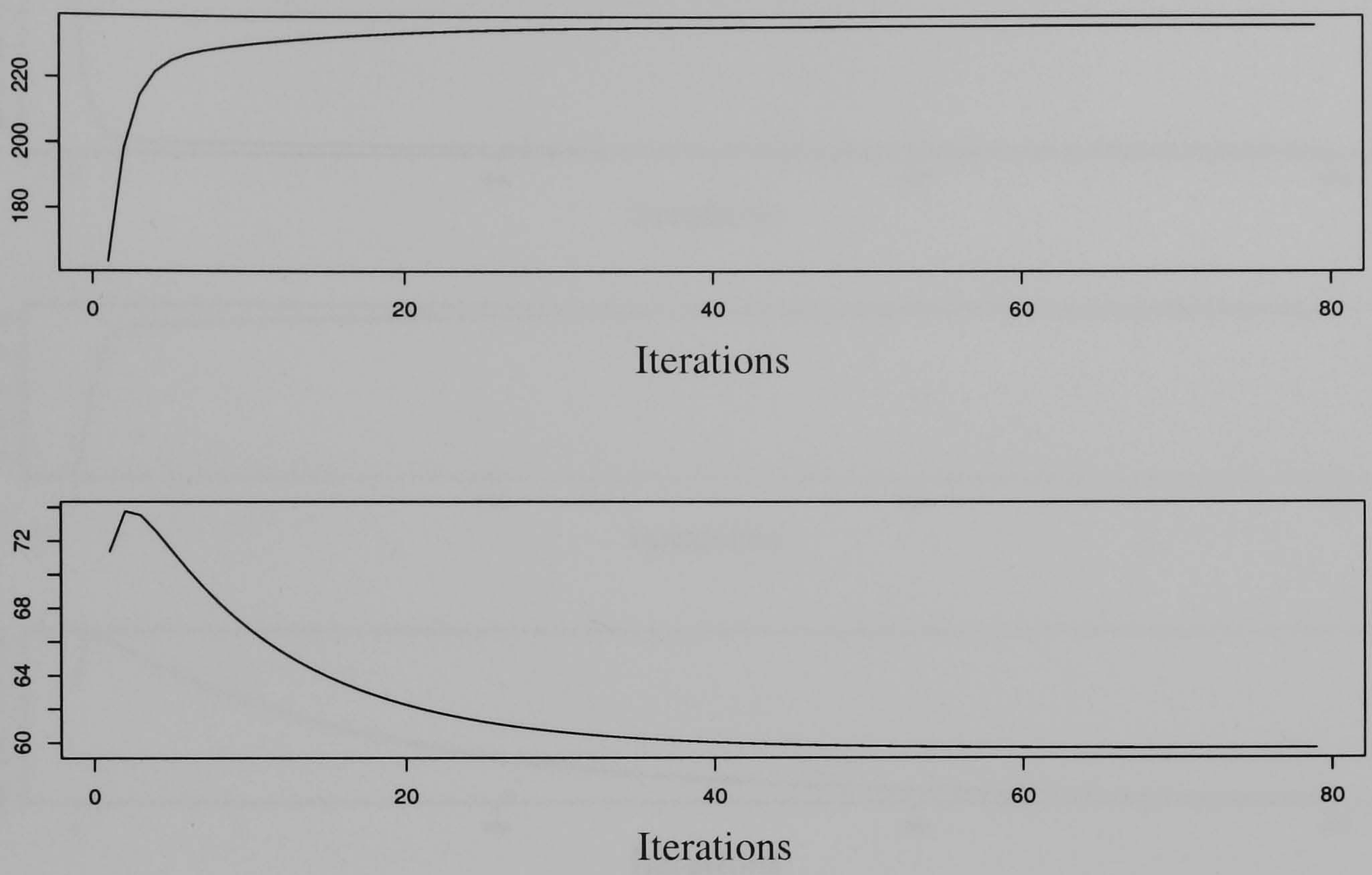


Figure 3.1: Convergence plots for Univariate locally constant EM Algorithm

components of 225, 49 and 0.01 respectively. The value for Z is least well estimated as this component is very small relative to the other two making it difficult for the algorithm to identify its effect. We have no real measure of the accuracy of these estimates although in this simulated case we know the true values from our data.

3.1.3 The Multivariate EM Algorithm

In Section 3.1.1 we have considered univariate series, which although interesting do not reflect the portfolios we will be looking at in later chapters, what is required is an extension of this algorithm for a p dimensional multivariate dynamic linear models. Let us first consider a locally constant model, where $\theta_t = (\mu_{1t}, \mu_{2t}, \ldots, \mu_{pt})$. In this case we will look directly at the posterior modes of V and W.

The Expectation Step

As discussed in Section 2.3.2 the Kalman filter smoother gives us a distribution for $\theta_t|D \sim N(m_t^*, C_t^*)$ and as in the univariate case this can be used to calculate the necessary expecta-

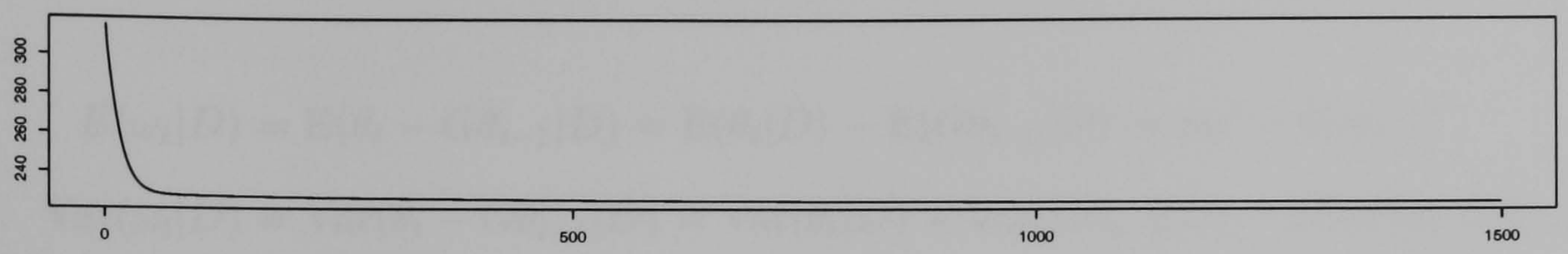


Figure 3.2: Convergence plots for Univariate locally linear EM Algorithm

tions.

For the M step we require posterior expectations for $\nu_t \nu_t'$ and $\omega_t \omega_t'$ for $t = 1, \ldots, n$. These are easily calculated from the following equations

$$\mathrm{E}(\nu_t {\nu_t}' | D) = \mathrm{Var}(\nu_t | D) + \mathrm{E}(\nu_t | D)\mathrm{E}(\nu_t | D)'$$

where

$$\mathrm{E}(\nu_t | D) = \mathrm{E}(Y_t - F'\theta_t | D) = Y_t - F'\mathrm{E}(\theta_t | D) = Y_t - F' {m_t}^*$$

$$\mathrm{Var}(\nu_t | D) = \mathrm{Var}(Y_t - F'\theta_t | D) = \mathrm{Var}(F'\theta_t | D) = F' {C_t}^* F$$

and for ω's

$$E(\omega_t {\omega_t}' | D) = \mathrm{Var}(\omega_t | D) + E(\omega_t | D) E(\omega_t | D)'$$

where:

$$E(\omega_t|D) = \mathrm{E}(\theta_t - G\theta_{t-1}|D) = \mathrm{E}(\theta_t|D) - \mathrm{E}(G\theta_{t-1}|D) = {m_t}^* - G{m_{t-1}}^*$$

$$\begin{aligned}\mathrm{Var}(\omega_t|D) &= \mathrm{Var}(\theta_t - G\theta_{t-1}|D) = \mathrm{Var}(\theta_t|D) + \mathrm{Var}(G\theta_{t-1}|D) - 2\mathrm{Cov}(\theta_t, G\theta_{t-1}|D) \\ &= {C_t}^* + G{C_{t-1}}^*G' - 2G\mathrm{Cov}(\theta_t, \theta_{t-1}|D)G'.\end{aligned}$$

As in the univariate case all these values are easily calculated from the expectation and variance of the state given all the states and data as generated by the Kalman filter and smoother. The **sather** class KALMAN has been written to include modules which perform these calculations.

The Maximisation Step

In the multivariate case, the ν_t's are multivariate normal, with a mean vector length p of zeros and a $p \times p$ variance matrix V, that is

$$\nu_t \sim N(0, V) \quad \text{for} \quad t = 1, 2, \ldots, n.$$

We can place a conjugate Inverse Wishart prior directly to the variance matrix V. This is a natural extension to the univariate example as a Gamma distribution is simply a Wishart with one degree of freedom.

$$V \sim IW(\delta, A).$$

Where δ is the degrees of freedom, which in this case can be regarded as representing the level of prior knowledge and A is a matrix parameter related to the mean of the distribution. The prior for V, therefore has the form

$$\pi(V) \propto |V|^{-\frac{1}{2}(\delta+p+1)} etr\left(-\frac{1}{2}V^{-1}A\right)$$

Where p is the dimension of both A and V, that is they are both $p \times p$ and $etr \equiv \exp(trace(A))$. Now the likelihood of $V|\nu$ is

$$\mathrm{L}(V|\nu) \propto |V|^{-\frac{n}{2}} etr\left(-\frac{1}{2}V^{-1}({\nu_1}{\nu_1}' + \cdots + {\nu_n}{\nu_n}')\right)$$

Using Bayes theorem, (1.1), we can combine the prior and the Likelihood to get the posterior

$$V|\nu \propto |V|^{-\frac{1}{2}(\delta+n+p+1)} etr\left(-\frac{1}{2}V^{-1}(A + {\nu_1}{\nu_1}' + \cdots + {\nu_n}{\nu_n}')\right)$$

That is

$$V|\nu \sim IW(\delta + n, A + {\nu_1}{\nu_1}' + \cdots + {\nu_n}{\nu_n}')$$

We require the posterior mode for $V|\nu$, now in general if

$$B \sim IW(\delta, A)$$
$$mode(B) = \frac{1}{\delta + p + 1}A$$

Hence in this case, taking the posterior expectations from the E-step the value we require can easily be calculated from

$$mode(V|\nu) = \frac{1}{\delta + n + p + 1}\left(A + \mathrm{E}({\nu_1}{\nu_1}'|D) + \cdots + \mathrm{E}({\nu_n}{\nu_n}'|D)\right) \tag{3.3}$$

A similar result can be found for the W matrix by applying an appropriate independent Inverse Wishart prior (which we will still designate as A) where the posterior mode for $W|\omega$ is

$$mode(W|\omega) = \frac{1}{\delta + n + p + 1}\left(A + \mathrm{E}({\omega_1}{\omega_1}'|D) + \cdots + \mathrm{E}({\omega_n}{\omega_n}'|D)\right) \tag{3.4}$$

We now have estimates for the modes of the variance parameters V and W. These can be used in the Kalman Filter to re-estimate the system parameter θ, and hence also provide new estimates for ν and ω. This is the M-step of the EM algorithm.

The Algorithm

1. Run the Kalman filter and smoother with some initial values for V and W, keeping the necessary summations.

2. Calculate the necessary posterior expectations and variances, this is done online in the **sather** class KALMAN.

3. Calculate the posterior modes using equations (3.3) and (3.4). If running with a locally linear model once again independent Inverse Wishart priors are placed on the two separate variance matrices which make up the W^* matrix.

4. Re-run the Kalman filter smoother using the new values for V and W

5. Repeat steps 2 to 4 until the algorithm converges. In this case convergence is regarded to be when the difference between subsequent values for V and W is sufficiently small. (The value of small is set by the modeller depending on the prior beliefs as to the scale of the two variance components.)

3.1.4 Examples

Consider the following locally constant multivariate dynamic linear model.

$$Y_t = F'\theta_t + \nu_t \quad \nu_t \sim N(0, V)$$

$$\theta_t = G\theta_{t-1} + \omega_t \quad \omega_t \sim N(0, W)$$

Where

$$V = \begin{pmatrix} 225 & 100 & 100 & 100 \\ 100 & 300 & 100 & 100 \\ 100 & 100 & 250 & 100 \\ 100 & 100 & 100 & 350 \end{pmatrix} \quad W = \begin{pmatrix} 49 & 10 & 10 & 10 \\ 10 & 35 & 10 & 10 \\ 10 & 10 & 60 & 10 \\ 10 & 10 & 10 & 75 \end{pmatrix}$$

and θ_t is a vector of length 4 and $F = G = \mathbf{I}$ a 4×4 identity matrix.

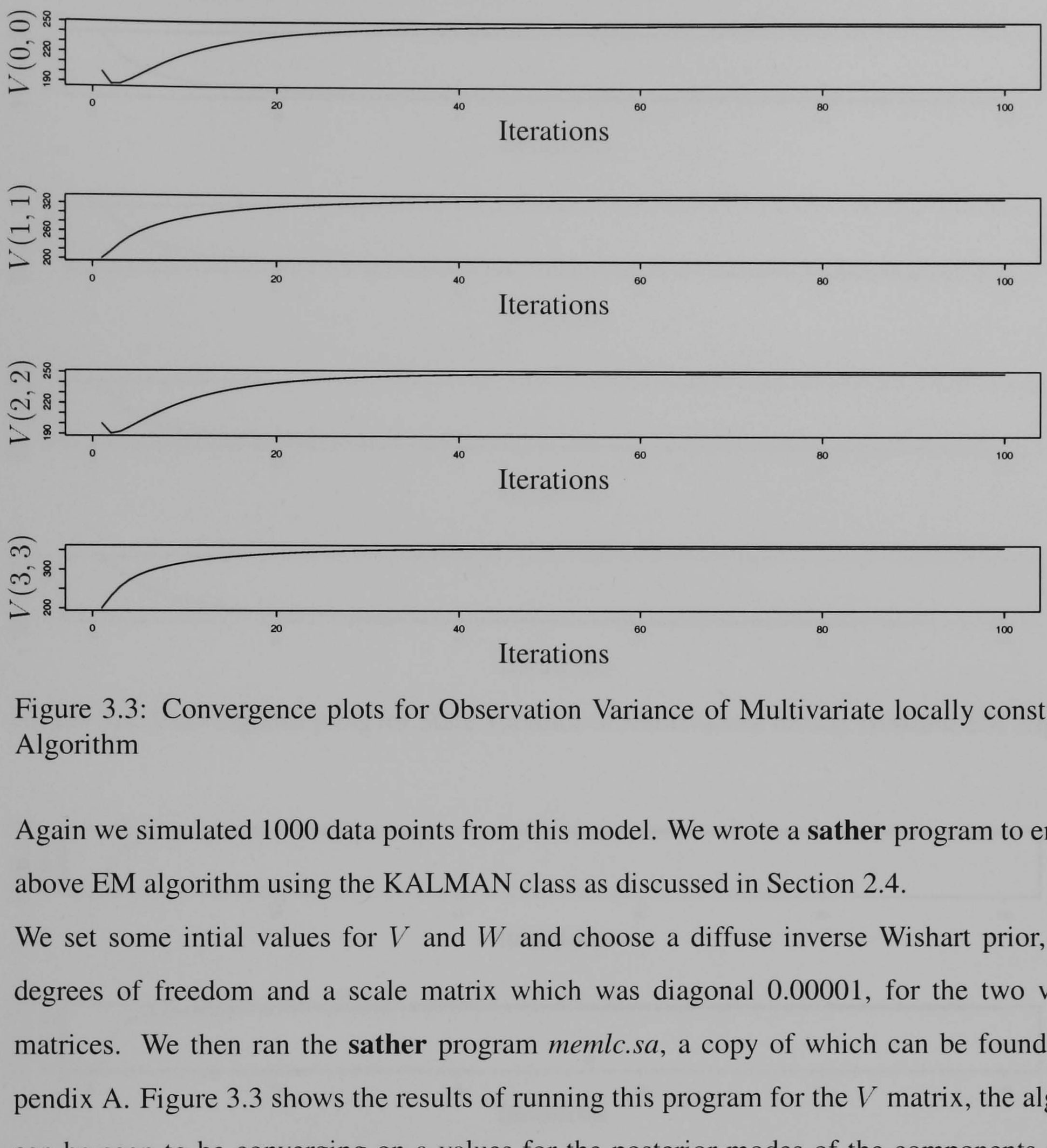


Figure 3.3: Convergence plots for Observation Variance of Multivariate locally constant EM Algorithm

Again we simulated 1000 data points from this model. We wrote a **sather** program to enact the above EM algorithm using the KALMAN class as discussed in Section 2.4.
We set some intial values for V and W and choose a diffuse inverse Wishart prior, with 5 degrees of freedom and a scale matrix which was diagonal 0.00001, for the two variance matrices. We then ran the **sather** program *memlc.sa*, a copy of which can be found in Appendix A. Figure 3.3 shows the results of running this program for the V matrix, the algorithm can be seen to be converging on a values for the posterior modes of the components of V of $214.656, 309.841, 259.549$ and 363.683. Figure 3.4 shows the convergence of the algorithm for the posterior modes of the components of W as $52.3740, 30.9994, 58.2181$ and 71.7171. In both case these are close to the true values as given in our original model. Looking at Figure 3.5 shows convergence towards the true values of these selected covariance terms.

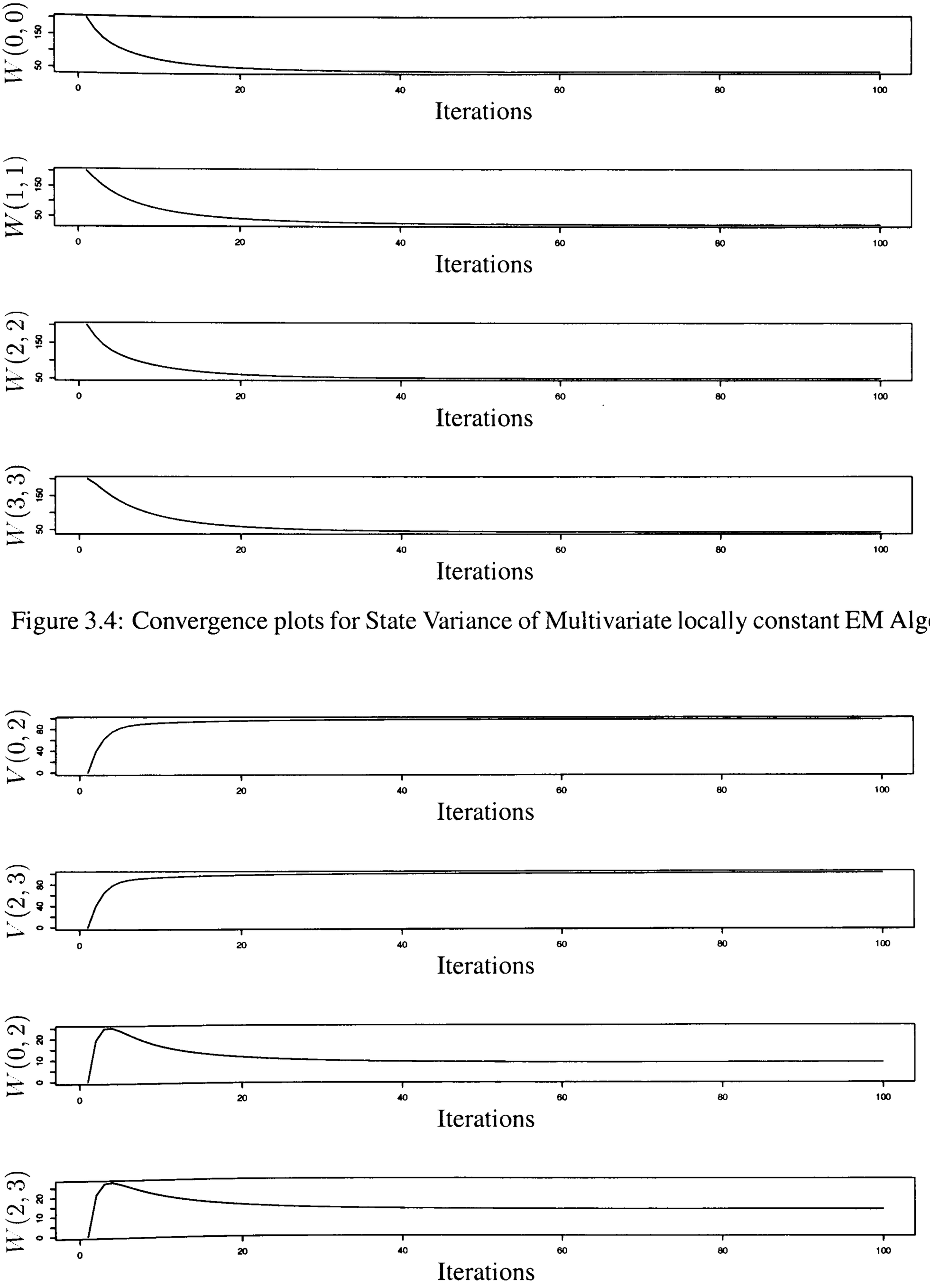


Figure 3.4: Convergence plots for State Variance of Multivariate locally constant EM Algorithm

Figure 3.5: Convergence plots for covariance terms of the Observation and State Variance of Multivariate locally constant EM Algorithm

As with the univariate example we can add in a trend component giving us the following model

$$Y_t = F_t'\theta_t + \nu_t \qquad \nu_t \sim N(0, V)$$
$$\theta_t = G_t\theta_{t-1} + \omega_t \quad \omega_t \sim N(0, W^*)$$

Where

$$\theta_t = \begin{pmatrix} \mu_t \\ \tau_t \end{pmatrix}, F_t = \begin{pmatrix} \mathbf{I} \\ 0 \end{pmatrix}, G_t = \begin{pmatrix} \mathbf{I} & \mathbf{I} \\ 0 & \mathbf{I} \end{pmatrix}, W^* = \begin{pmatrix} W & 0 \\ 0 & Z \end{pmatrix}$$

$$V = \begin{pmatrix} 225 & 100 & 100 & 100 \\ 100 & 300 & 100 & 100 \\ 100 & 100 & 250 & 100 \\ 100 & 100 & 100 & 350 \end{pmatrix} W = \begin{pmatrix} 49 & 10 & 10 & 10 \\ 10 & 35 & 10 & 10 \\ 10 & 10 & 60 & 10 \\ 10 & 10 & 10 & 75 \end{pmatrix} Z = \begin{pmatrix} 0.01 & 0 & 0 & 0 \\ 0 & 0.02 & 0 & 0 \\ 0 & 0 & 0.05 & 0 \\ 0 & 0 & 0 & 0.025 \end{pmatrix}.$$

Again we can simulate 1000 data points from this model. We can write a **sather** program to enact the above EM algorithm using the KALMAN class as discussed in Section 2.4.

We set some initial values for V, W and Z and choose a flat Wishart prior for each of the three variance matrices. We can then run the **sather** program *memll.sa*, a copy of which can be found in Appendix A.

Figure 3.6 shows the results of running this program for the V matrix, the algorithm can be seen to be converging on a values for the posterior modes of the diagonal components of V of $218.743, 313.533, 258.379$ and 328.335. Figure 3.7 shows the convergence of the algorithm for the posterior modes of the diagonal components of W as $52.9007, 39.3974, 66.724$ and 72.5774. Figure 3.8 shows the convergence of the algorithm for the posterior modes of the diagonal components of Z as $0.0032, 0.0002, 0.0002$ and 0.00294. In this case we see the algorithm converging towards values we know to be true, with some under estimation of the trend component variance Z. The values for the simulated data were chosen to reflect what we believe to be likely values in real data sets, a larger value of Z in the simulated data might produce different results.

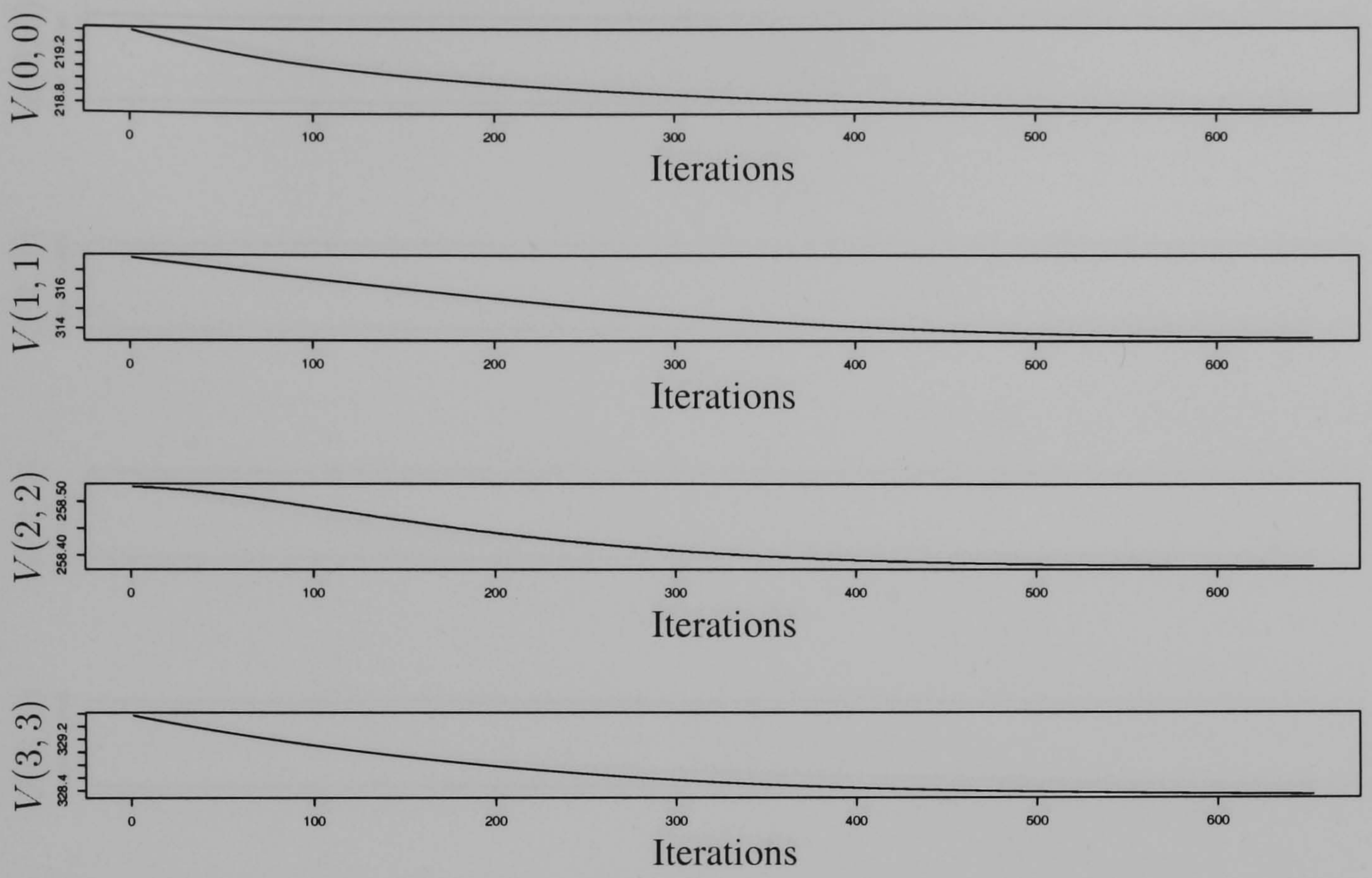


Figure 3.6: Convergence plots for Observation Variance of Multivariate locally linear EM Algorithm

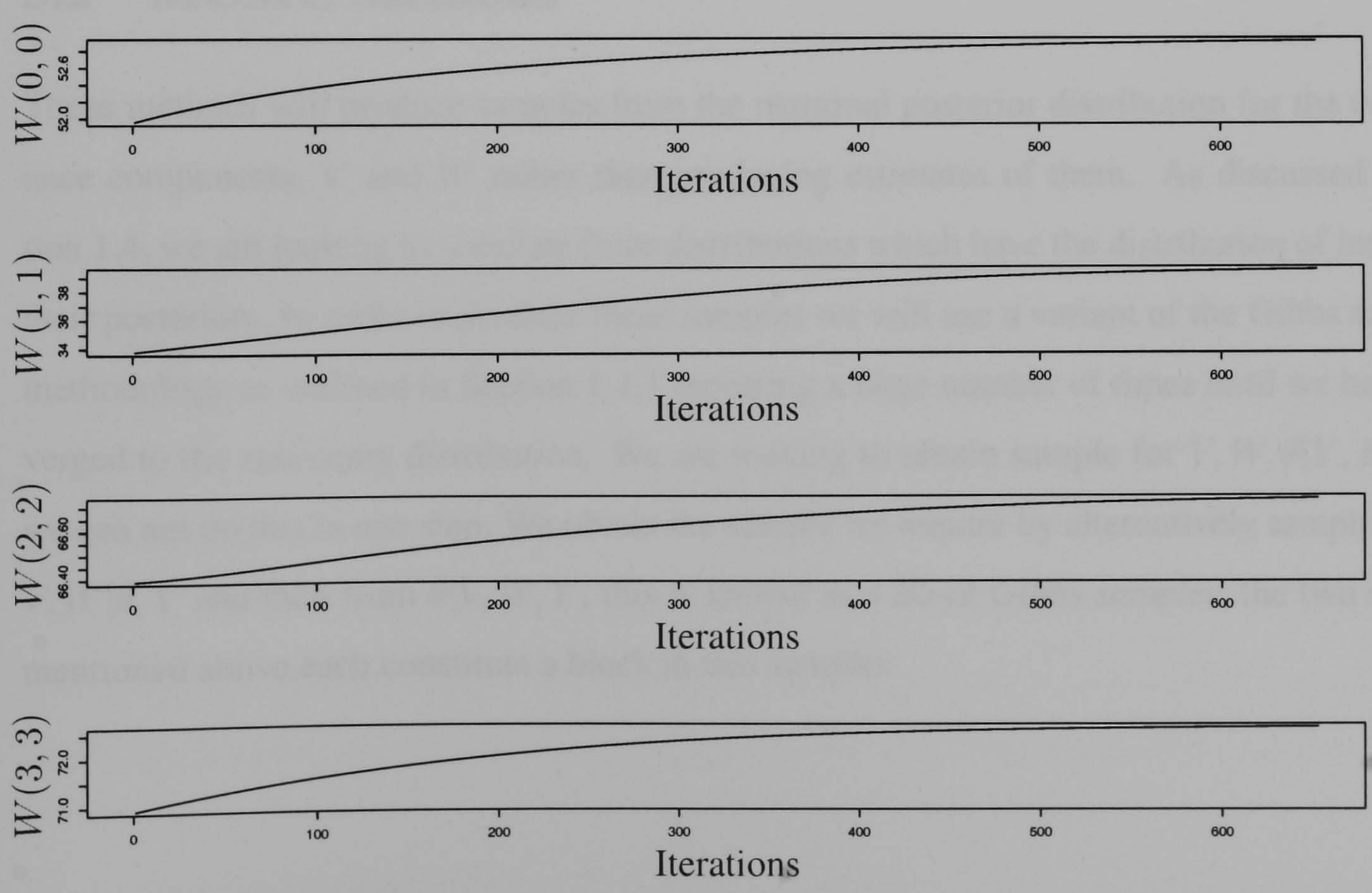


Figure 3.7: Convergence plots for System Variance of Multivariate locally linear EM Algorithm

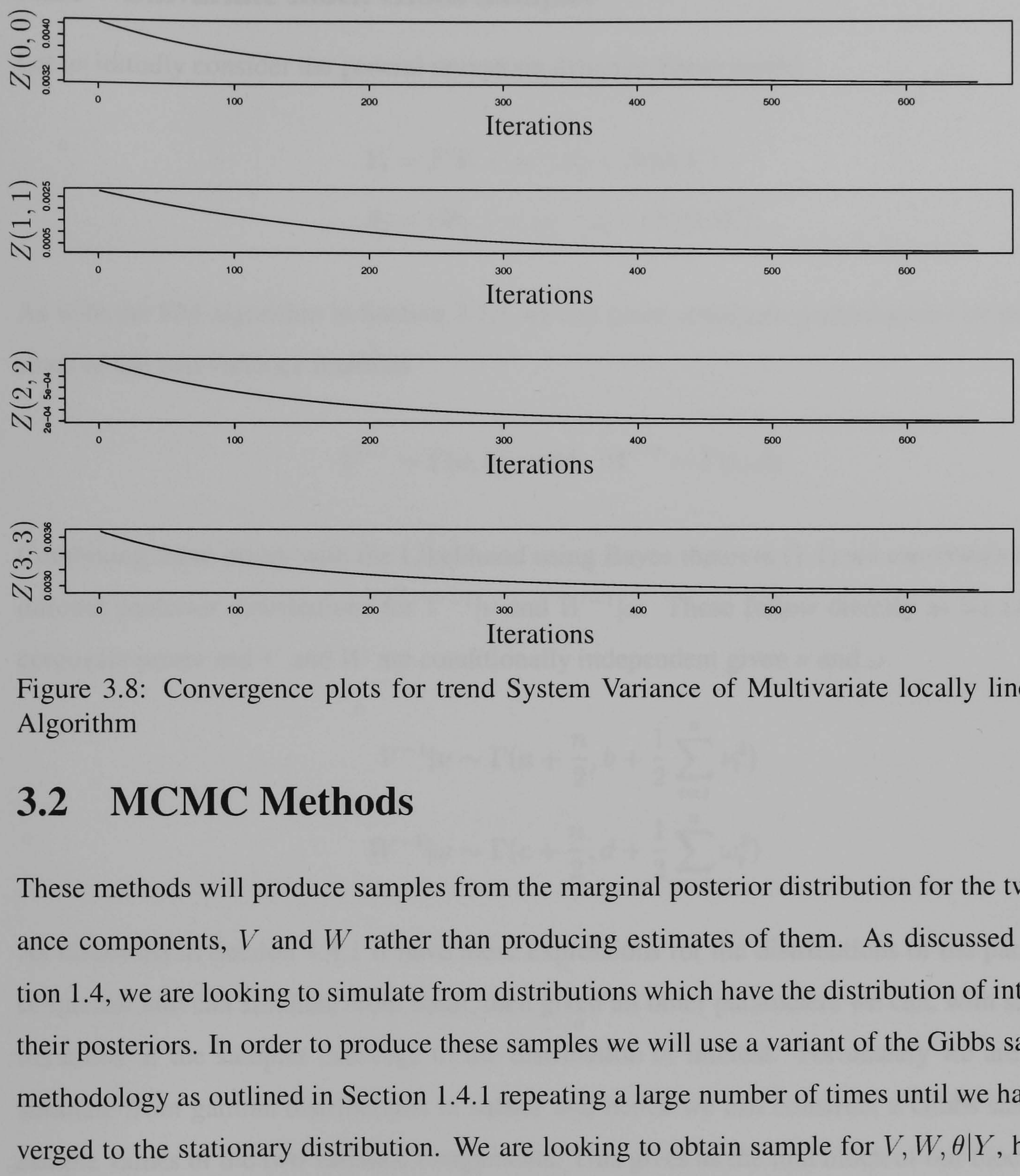


Figure 3.8: Convergence plots for trend System Variance of Multivariate locally linear EM Algorithm

3.2 MCMC Methods

These methods will produce samples from the marginal posterior distribution for the two variance components, V and W rather than producing estimates of them. As discussed in Section 1.4, we are looking to simulate from distributions which have the distribution of interest as their posteriors. In order to produce these samples we will use a variant of the Gibbs sampling methodology as outlined in Section 1.4.1 repeating a large number of times until we have converged to the stationary distribution. We are looking to obtain sample for $V, W, \theta|Y$, however we can not do this in one step. We obtain the sample we require by alternatively sampling from $V, W|\theta, Y$ and then from $\theta|V, W, Y$, this is known as a *Block Gibbs sampler*, the two samples mentioned above each constitute a block in this sampler.

3.2.1 Univariate Block Gibbs Sampler

Let us initially consider the general univariate dynamic linear model

$$Y_t = F'\theta_t + \nu_t \quad \nu_t \sim N(0, V)$$
$$\theta_t = G\theta_{t-1} + \omega_t \quad \omega_t \sim N(0, W).$$

As with the EM algorithm in Section 3.1.1 we can place conjugate gamma priors on the precisions of the two variance matrices

$$V^{-1} \sim \Gamma(a, b) \quad \text{and} \quad W^{-1} \sim \Gamma(c, d).$$

Combining these priors with the Likelihood using Bayes theorem (1.1) we can obtain the conditional posterior distributions for $V^{-1}|\nu$ and $W^{-1}|\omega$. These follow directly as we are using conjugate priors and V and W are conditionally independent given ν and ω.

$$V^{-1}|\nu \sim \Gamma(a + \frac{n}{2}, b + \frac{1}{2}\sum_{t=1}^{n} \nu_t^2) \tag{3.5}$$
$$W^{-1}|\omega \sim \Gamma(c + \frac{n}{2}, d + \frac{1}{2}\sum_{t=1}^{n} \omega_t^2) \tag{3.6}$$

As discussed in Section 1.4.1 if have these expressions for the distributions of the parameters of interest and can simulate from them, then given all other parameters we can, with sufficient iterations of the sampler converge to the distribution of interest. Fortunately we are able to simulate from gamma distributions in **sather** and hence we can construct a Gibbs sampler to sample values of the two variance components. This gives us the first block of our block Gibbs sampler, sampling from $V, W|\theta, Y$.

The second block of the block Gibbs sampler requires us to sample $\theta|V, W, Y$, in Section 2.3.3 we have shown that the kalman filter simulation smoother allows us to do this. The second block of the block Gibbs sampler is therefore simply a Kalman filter simulation smoother routine,

from which we can easily calculate the values of ν_t and ω_t required in the first block from,

$$\nu_t = Y_t - F'\theta_t$$

$$\omega_t = \theta_t - G\theta_{t-1}.$$

The necessary summations are simply calculated in **sather** . We can alternate between these two blocks until we reach convergence.

To see this more clearly consider the following algorithm.

The Algorithm

1. Run the Kalman filter and simulation smoother with some initial values for V and W to sample $\theta|V, W, Y$.

2. Calculate $\sum_{t=1}^{n} \nu_t^2$ and $\sum_{t=1}^{n} \omega_t^2$. This is done on-line within the **sather** class KALMAN.

3. Simulate new values for $V, W|\theta, Y$ from the conditionals (3.5) and (3.6). In the case of a locally linear model, as in the EM algorithm, the two diagonal elements are simulated from independent Gamma distributions and then placed in the W matrix.

4. Re-run the Kalman filter simulation smoother using the new simulated values for V and W.

5. Repeat steps 2 - 4 a large number of times until the sampler has converged to the distribution of interest.

We can use the output from this sampler in a number of ways. We could take the mean of the samples for each variance component and use these as estimates of the values for these parameters. Since we have distributional information, however it might be more interesting to look at histograms of the samples from these conditional distributions as these could tell us more about the nature of our estimates. We could also look to make predictions about future values of the series using these samples, we will return to this later in Section 3.3

As discussed in Section 1.4.1 the issue of convergence of the Gibbs sampler is beyond the scope of this thesis, initially simple "marker pen" tests will be applied to the trace plots of the samples.

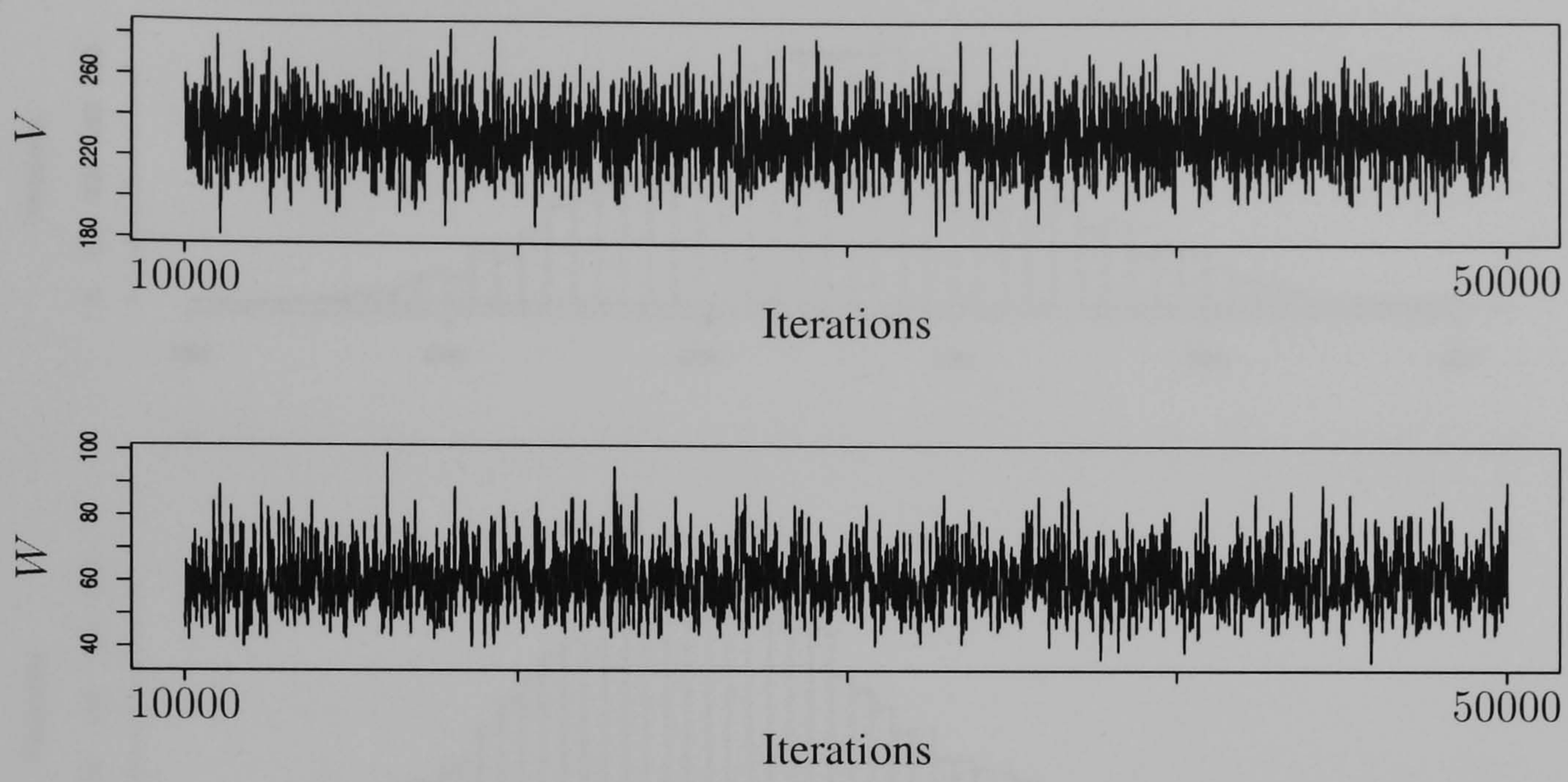


Figure 3.9: Trace plots for the samples of the variance components of locally constant dynamic linear model using a block Gibbs sampling algorithm

The "marker pen" test is a simple visual inspection of the trace plot, if the plot is a thick line which looks like it could have been drawn by a marker pen then the sampler can be considered to have converged. With variance components it is sometimes better to perform this test on the logarithm of the variance as the distribution is less skew and shorter tailed on the log scale, reducing the effects of extreme values.

3.2.2 Examples

We can look once again at our simple univariate locally constant DLM from Section 3.1.2, however this time instead of running an EM algorithm we will run the block Gibbs sampler algorithm as outlined above. This can be easily programmed in **sather** and we can sample from the posterior distributions for the two variance components.

We simulated 1000 points from this model after setting an initial value for θ_0.

We set some initial values for V and W and choose a flat $\Gamma(1, 0.0001)$ prior for both variance components. We then ran the **sather** program *umclc.sa*, a copy of which can be found in Appendix A which performs the block Gibbs sampling algorithm for the univariate locally constant dynamic linear model.

Figure 3.9 shows the trace plots for the two variance components of the locally constant DLM.

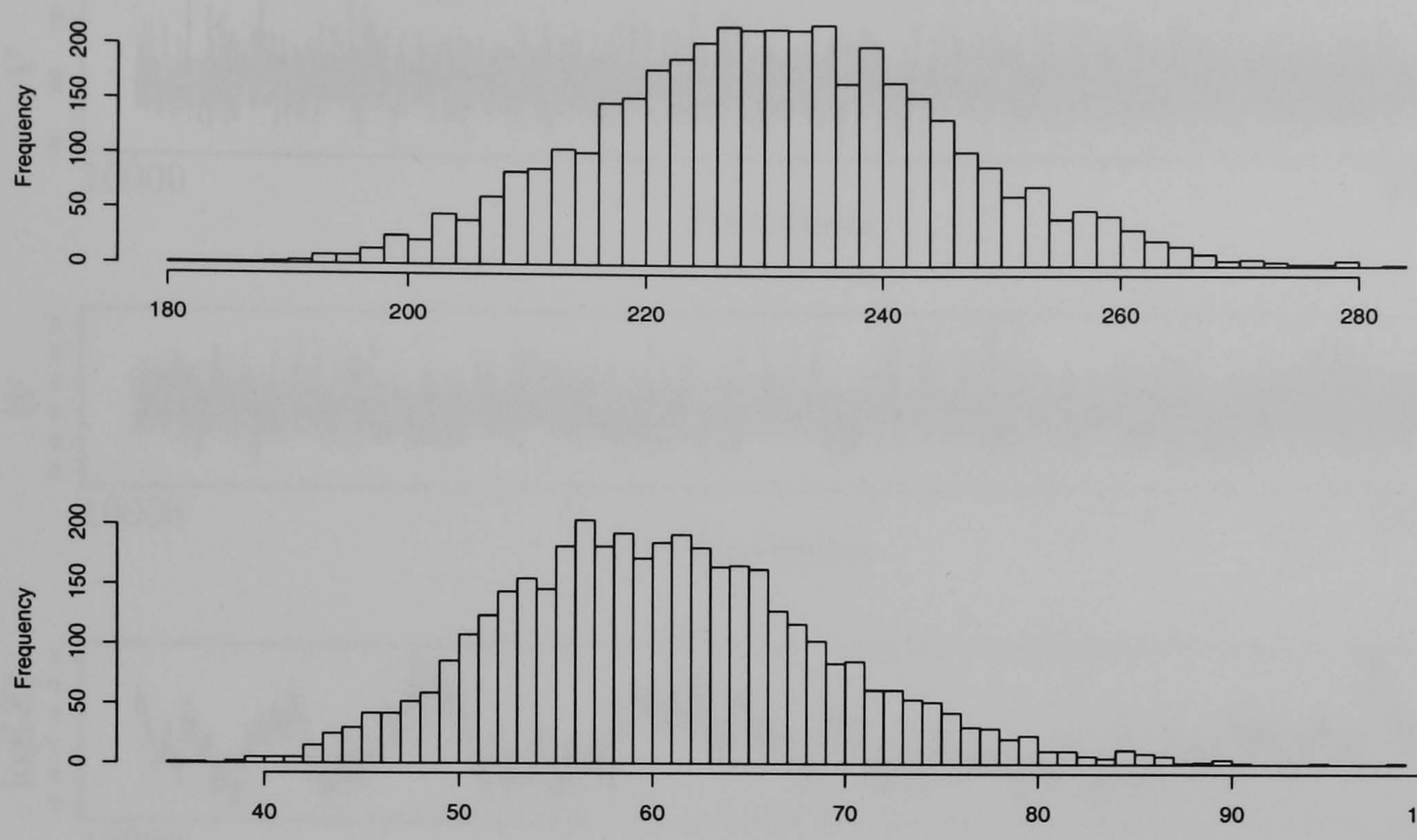


Figure 3.10: Histogram of the variance components of locally constant DLM

These plots are based on a run of $50,000$ with the first $10,000$ discarded as a burn-in. The remaining data was thinned by a factor of 10, where only every tenth observation is kept and plotted. These plots clearly show the convergence of the Gibbs sampler as they produce classic "marker pen" plots.

Figure 3.10 shows the histograms of the samples drawn from the conditional posterior distributions. The histogram for V appears approximately symmetrical centred around a value of approximately 230, close to the true value of 225. The 95% confidence interval for this is $(202.4, 260.9)$.Although not highly peaked this histogram has a narrow spread with most samples falling in the range of the confidence interval. The histogram for W is slightly asymmetrical with a longer right hand tail, again the range is quite compact with a 95% confidence interval of $(45.5, 78.9)$. The modal value appears close to that obtained using the EM algorithm in Section 3.1.2 of nearly 60. For both variance components the histograms appear to correspond well with the results obtained using the EM algorithm.

We can also consider our locally linear model from Section 3.1.2, simulating 1000 points from this model after setting an initial value for θ_0.

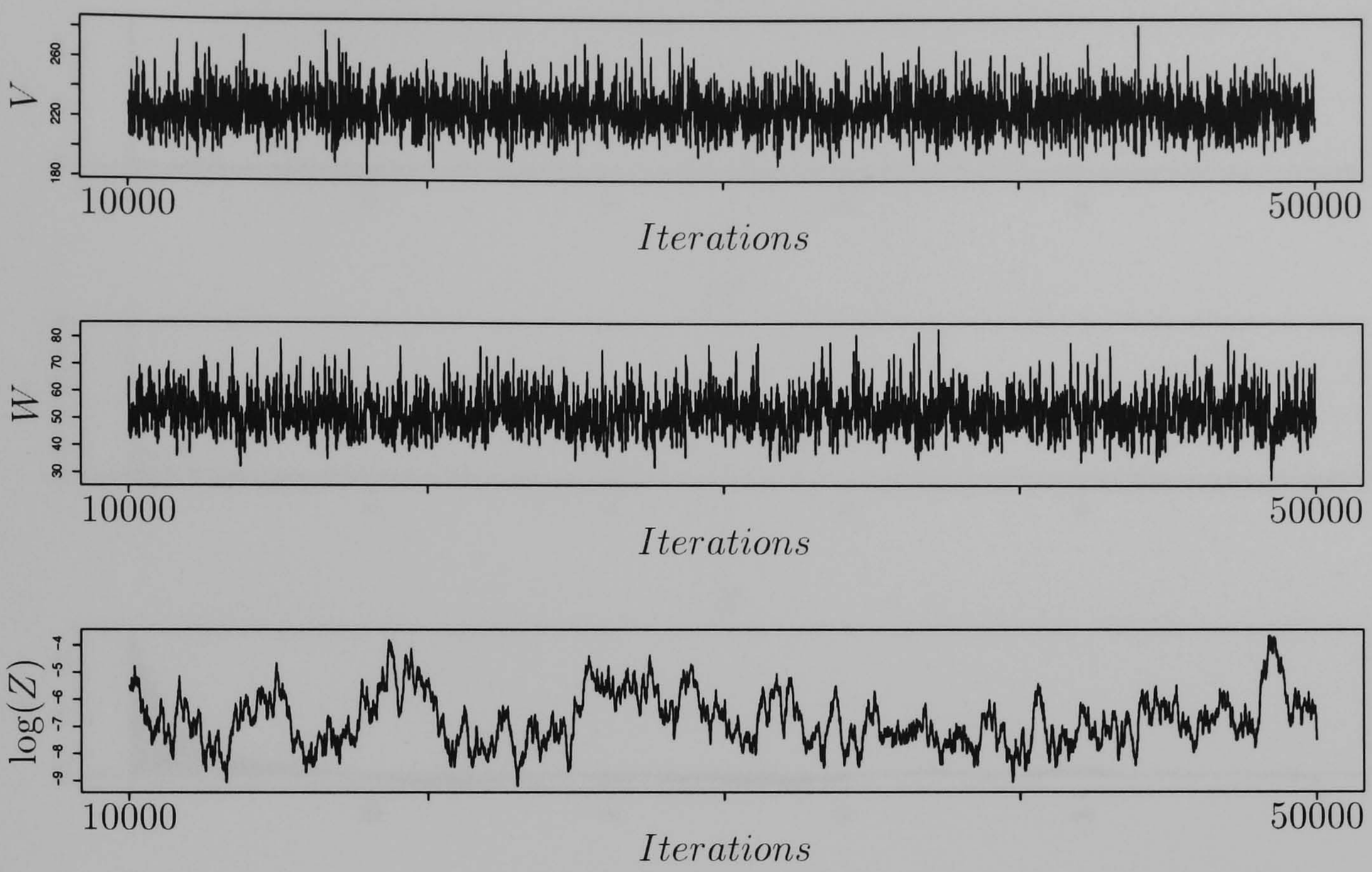


Figure 3.11: Trace plots for variance components of locally linear DLM

We set some initial values for V and W^* and choose flat independent $\Gamma(1, 0.0001)$ priors for all three variance components. We can then ran the **sather** program *umcll.sa*, a copy of which can be found in Appendix A which performs the block Gibbs sampling algorithm for the univariate locally linear model.

Figure 3.11 shows the trace plots for the three variance components of this locally constant dynamic linear model. These plots are based on a run of $50,000$ with the first $10,000$ dropped as a burn-in. The remaining data was thinned by a factor of 10. The plots for V and W suggest that the sampler has converged which is supported by the autocorrelation plots in Figure 3.12. For the trend component Z the logs are taken reduce the effects of extreme values, both the trace plot and the autocorrelation plot which exhibits long lags suggest that this component has not fully converged to the stationary distribution. One solution might be to run the sampler for longer and thin by a higher factor.

Figure 3.13 shows the histograms of the samples drawn for the marginal distributions. The histogram for V appears approximately symmetrical centred around a value of approximately 225, the true value being 225. Although not highly peaked this histogram has a narrow spread

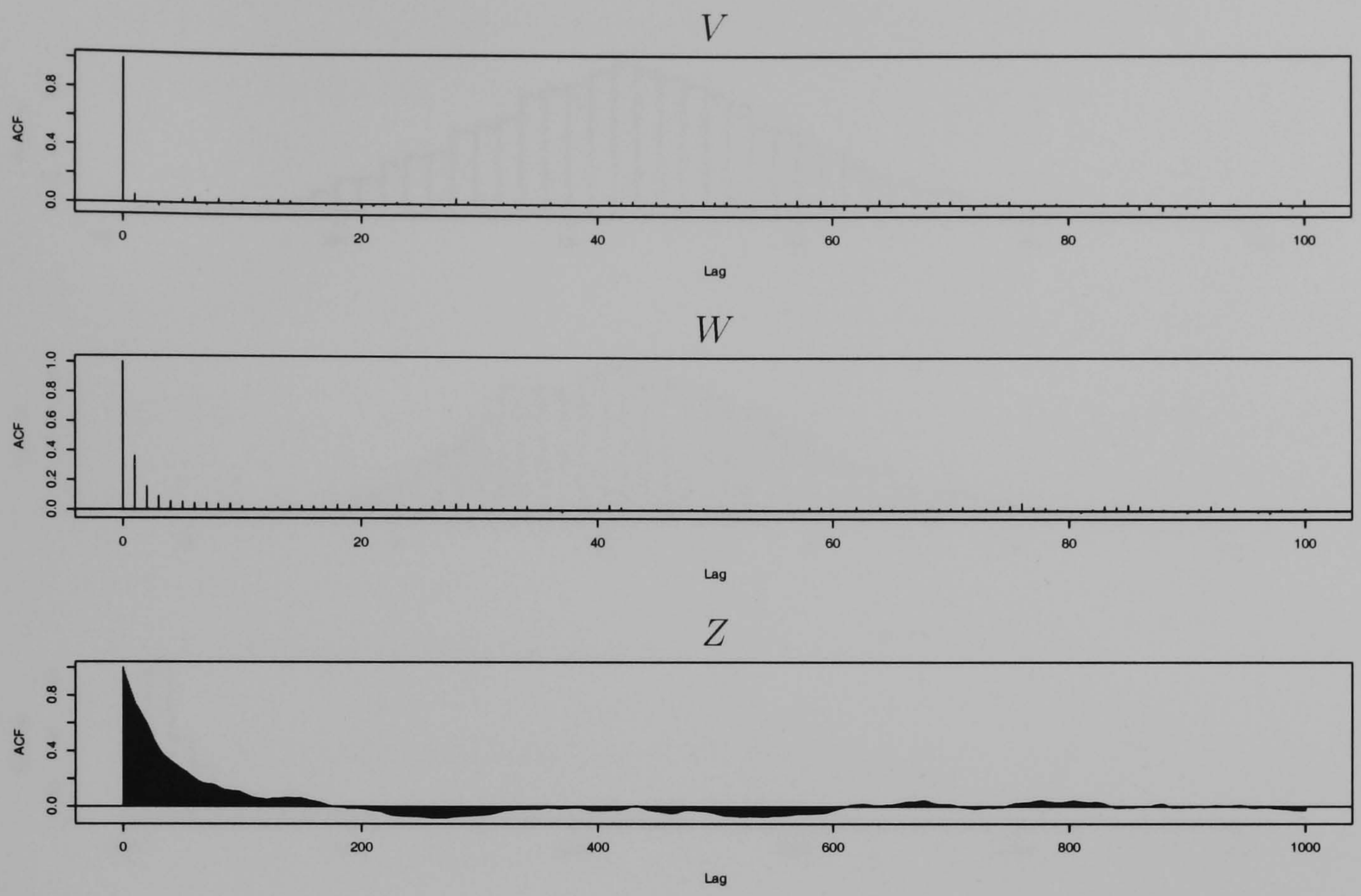


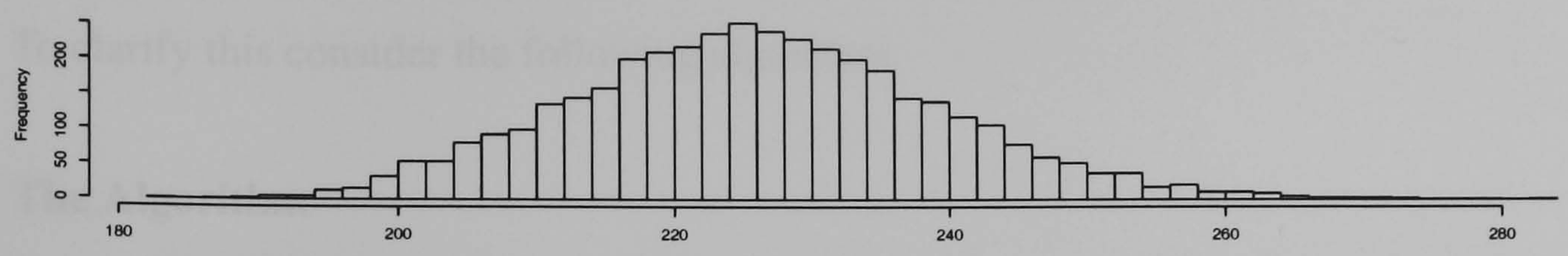
Figure 3.12: Autocorrelation plots for variance components of locally linear DLM

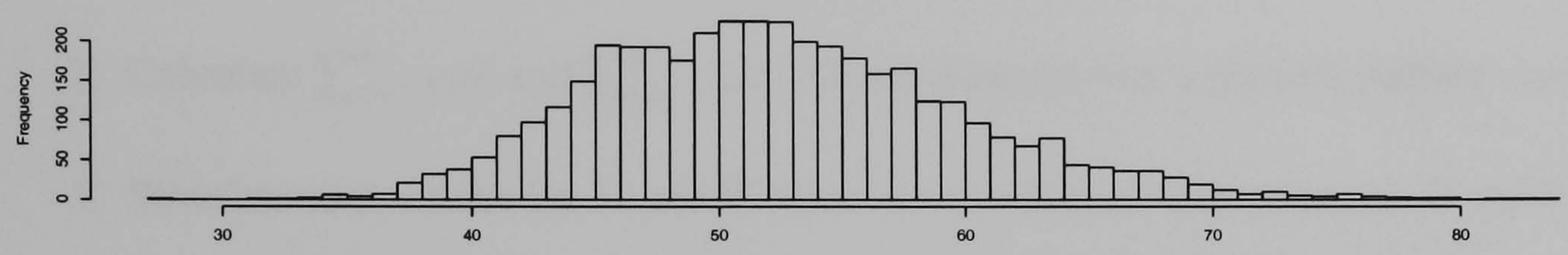
with most samples falling in the range of the 95% confidence interval of (200.5, 253.2). The histogram for W is slightly asymmetrical with a longer right hand tail, again the range is quite compact with most samples falling in the range of the 95% confidence interval (39.7, 68.0), with a modal value slightly higher than the true value of 49. The histogram for Z is skewed to the left with a long tail, its modal value appears lower than the true value and a 95% confidence interval of (0.0002, 0.010) In all cases the histograms appear to be consistent with the results obtained using the EM algorithm. The value obtained for Z is like in the EM algorithm not very close to the true value and might suggest the need for a more informative prior in such models.

3.2.3 Multivariate Block Gibbs Sampler

In reality as with the EM Algorithm we are more likely to be interested in multivariate series. This is again just a natural extension of the univariate case, where we create a block Gibbs sampler to sample $V, W, \theta|Y$ by alternately sampling $V, W|\theta, Y$ and then $\theta|V, W, Y$.

We sample $V, W|\theta, Y$ by placing inverse Wishart priors on the two variance matrices V and W,





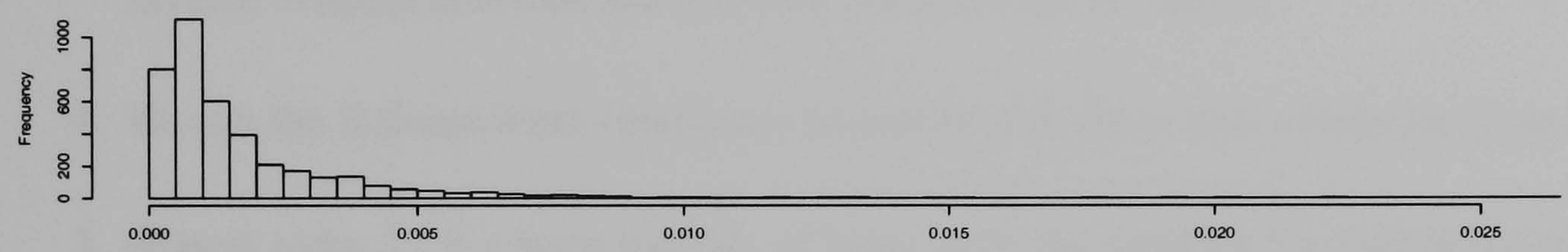


Figure 3.13: Histogram of the variance components of locally constant DLM

hence

$$V \sim IW(\delta_V, A_V) \quad \text{and} \quad W \sim IW(\delta_W, A_W)$$

Since

$$\nu_t \sim N(0, V) \quad \text{and} \quad \omega_t \sim N(0, W)$$

Combining these priors with the likelihoods using Bayes theorem (1.1) we get the following marginal conditional posterior distributions.

$$V|\nu_1, \ldots, \nu_n \sim IW(\delta_V + n, A_V + \nu_1\nu_1{}' + \cdots + \nu_n\nu_n{}') \tag{3.7}$$

$$W|\omega_1, \ldots, \omega_n \sim IW(\delta_W + n, A_W + \omega_1\omega_1{}' + \cdots + \omega_n\omega_n{}') \tag{3.8}$$

Again we sample $\theta|V, W, Y$ by running the kalman filter simulation smoother using the current values for V and W as the second block of the block Gibbs sampler. We can then calculate

$\sum_{t=1}^{n} \nu_t \nu_t'$ and $\sum_{t=1}^{n} \omega_t \omega_t'$ to allow us to produce new samples for V and W.
To clarify this consider the following algorithm.

The Algorithm

1. Run the Kalman filter and simulation smoother with some initial values for V and W.
2. Calculate $\sum_{t=1}^{n} \nu_t \nu_t'$ and $\sum_{t=1}^{n} \omega_t \omega_t'$. This is done on-line within the **sather** class KALMAN.
3. Simulate new values for V and W from the conditional posteriors (3.7) and (3.8). In the case of a locally linear model, the two diagonal matrices are simulated from independent inverse Wishart distributions and then placed in the W matrix.
4. Re-run the Kalman filter simulation smoother with these new values for V and W
5. Repeat steps 2 - 4 a large number of times until the sampler has converged to the distribution of interest.

As with the univariate case we need to ensure that convergence has been achieved. We can again take the mean of our samples as an estimate of the variance components or look at the histograms of the variance components. In the multivariate case we will be interested in the off-diagonal elements of the variance matrices because we are interested in the relationship between series as well as within series when we consider multivariate series. Indeed this is the advantage of a multivariate approach.
An alternative approach to modelling multivariate time series is to use multiregression dynamic models, we have not adopted that approach here but a fuller treatment can be found in Queen (1994).

3.2.4 Examples

We can look once again at our multivariate locally constant DLM from Section 3.1.4, however this time instead of running an EM algorithm we will run the block Gibbs sampler algorithm as outlined above. This can be easily programmed in **sather** and we can sample from the posterior distributions for the two variance components.

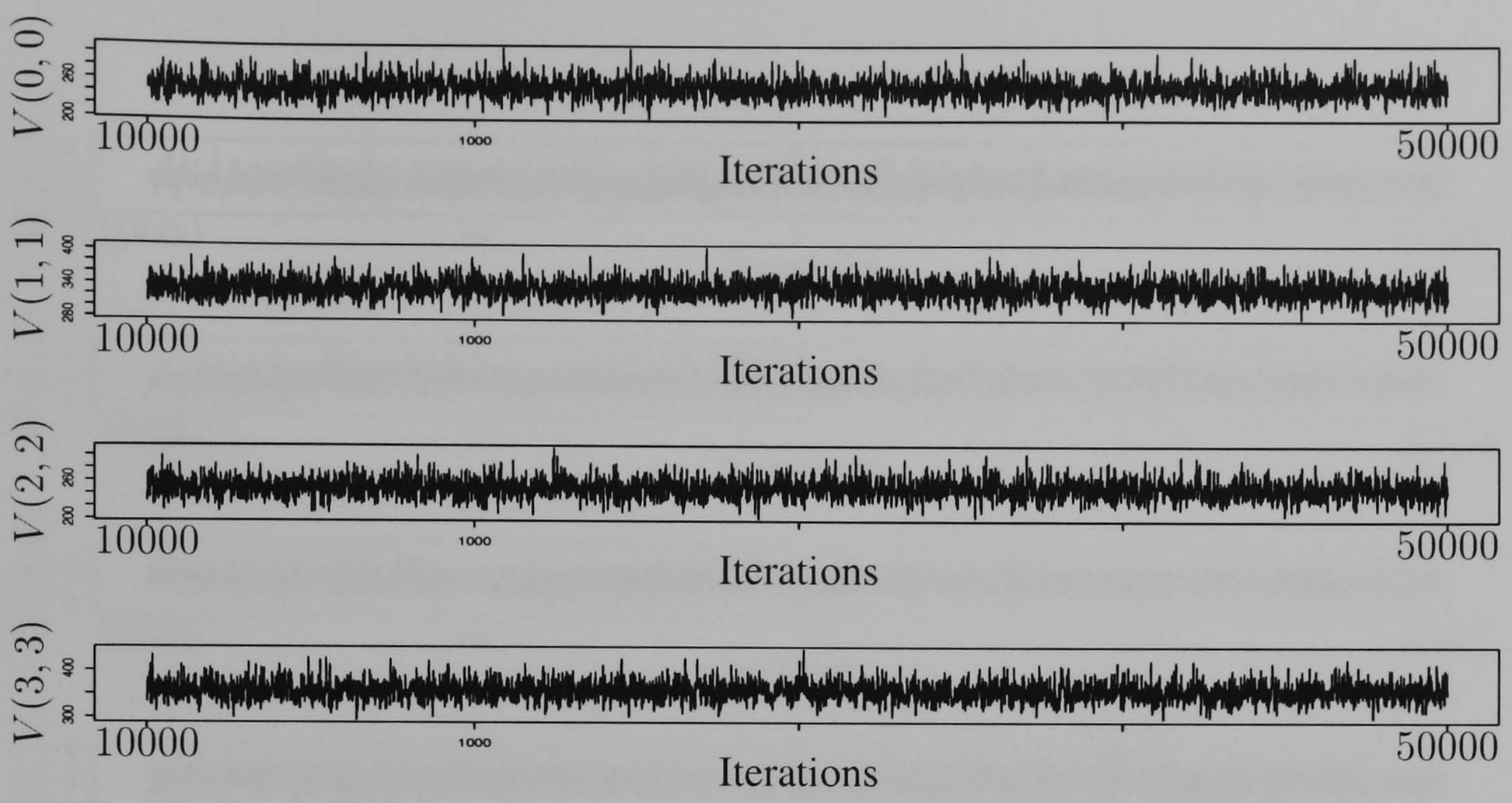


Figure 3.14: Trace plots for the samples of the observational variance components of locally constant dynamic linear model using a block Gibbs sampling algorithm

We use the same 1000 simulated data points as before.

We set initial values for V and W of 10, however we could perhaps have used the EM algorithm first as a "tuning" stage and then used the converged output from this as our initial values. We choose a non-informative Inverse Wishart prior for both variance matrices as previously in Section 3.1.4. We then ran the **sather** program *mmclc.sa*, a copy of which can be found in Appendix A which performs the block Gibbs sampling algorithm for the multivariate locally constant dynamic linear model.

Figure 3.14 and Figure 3.15 show the trace plots for the diagonal elements of the two variance matrices of the locally constant DLM. These plots are based on a sampling run of $50,000$ with the first $10,000$ dropped as a burn-in. The remaining data was thinned by a factor of 10. These plots clearly show the convergence of the Gibbs sampler as they produce classic "marker pen" plots, this is supported by looking at the autocorrelation plots Figure 3.20 which show a rapid tailing away.

Figure 3.17 and Figure 3.18 shows the histograms of the diagonal elements of the samples drawn from the conditional distributions for the two variance matrices of our model. The histograms of the observational variances, V all appear approximately symmetrical centred around the true values for these components. Although not highly peaked these histograms all have relatively

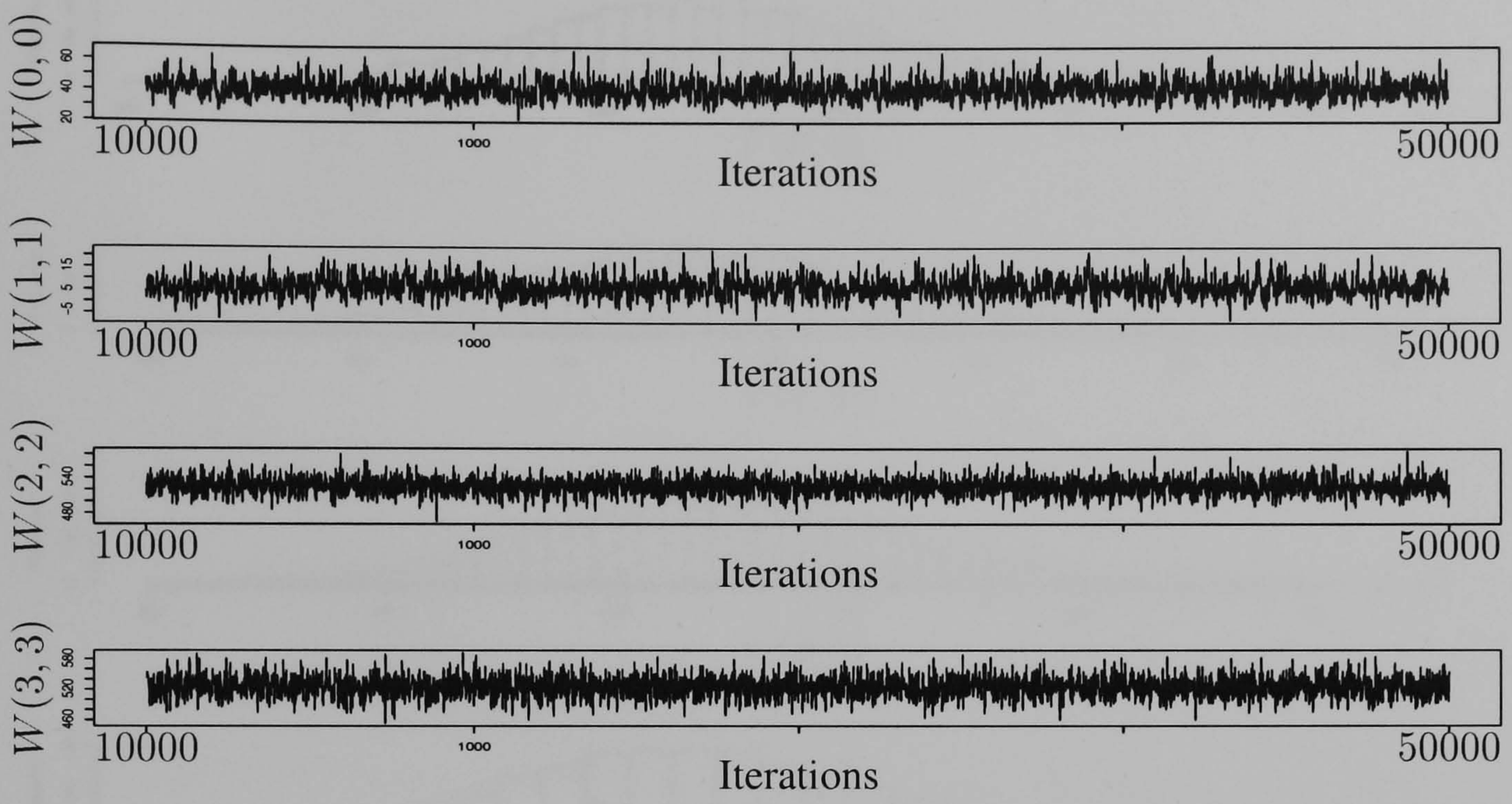


Figure 3.15: Trace plots for the samples of the system variance components of locally constant dynamic linear model using a block Gibbs sampling algorithm

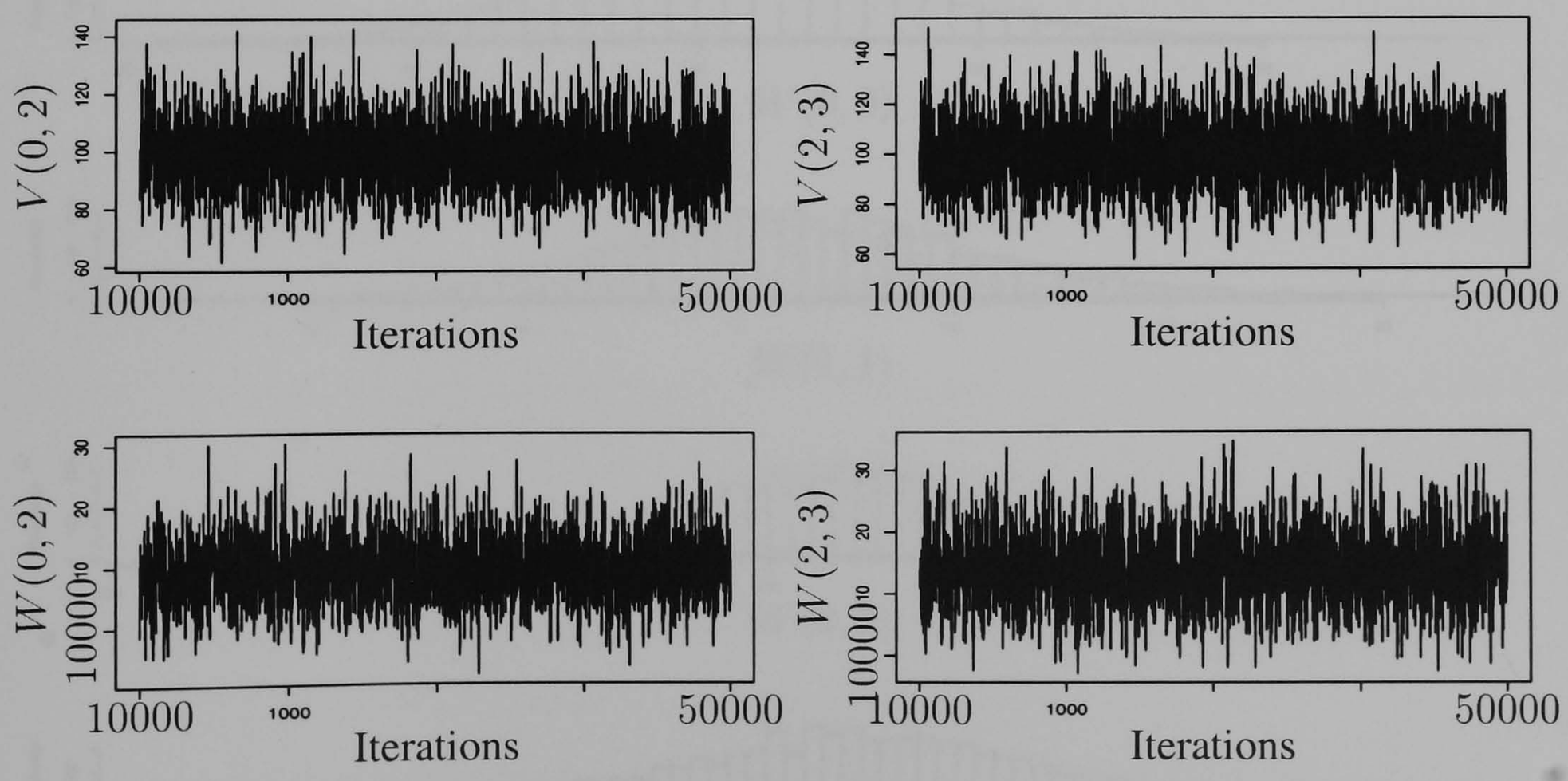


Figure 3.16: Trace plots for the samples of the selected covariance terms of locally constant dynamic linear model using a block Gibbs sampling algorithm

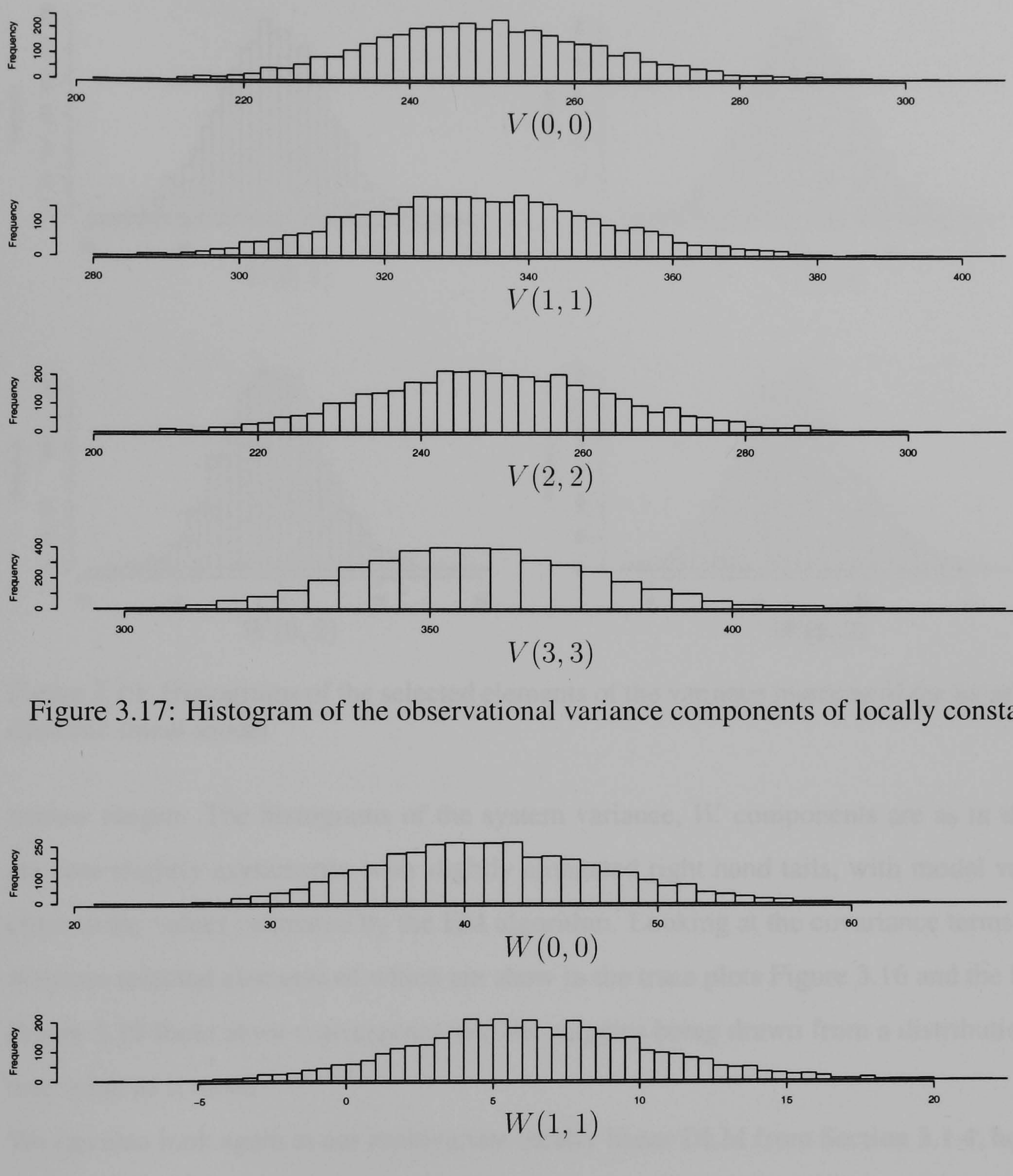
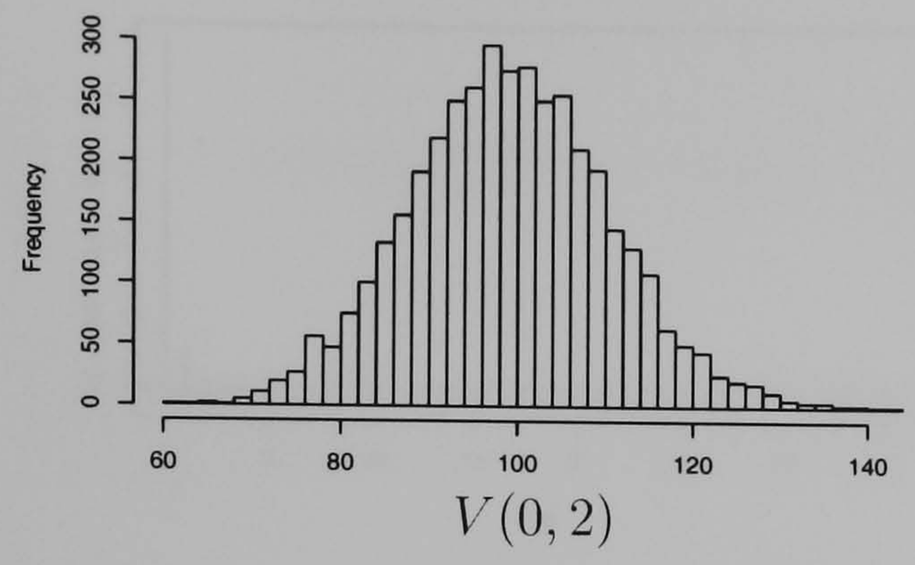
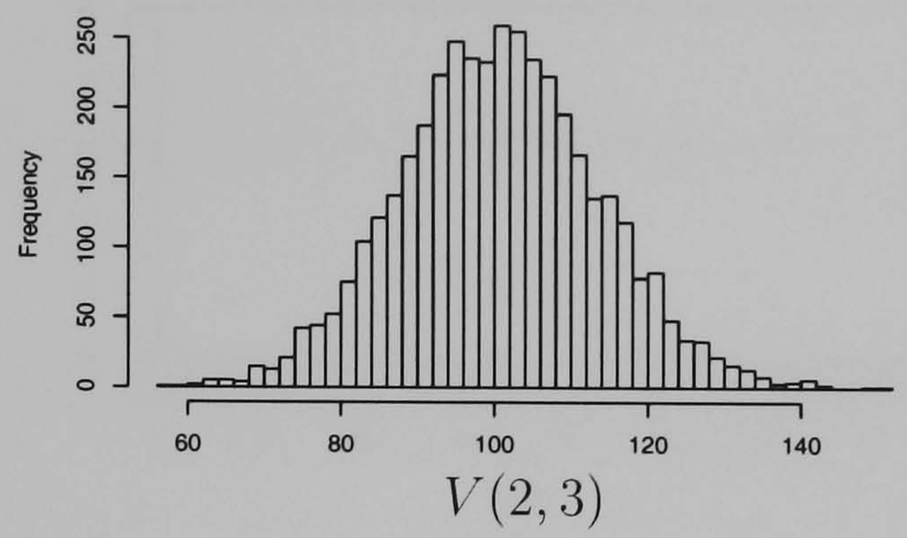
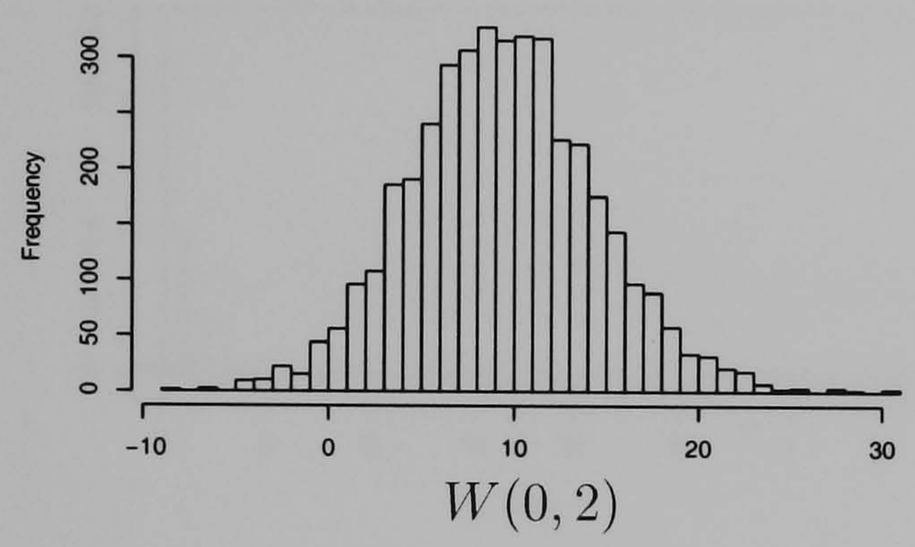


Figure 3.17: Histogram of the observational variance components of locally constant DLM

Figure 3.18: Histogram of the observational variance components of locally constant DLM







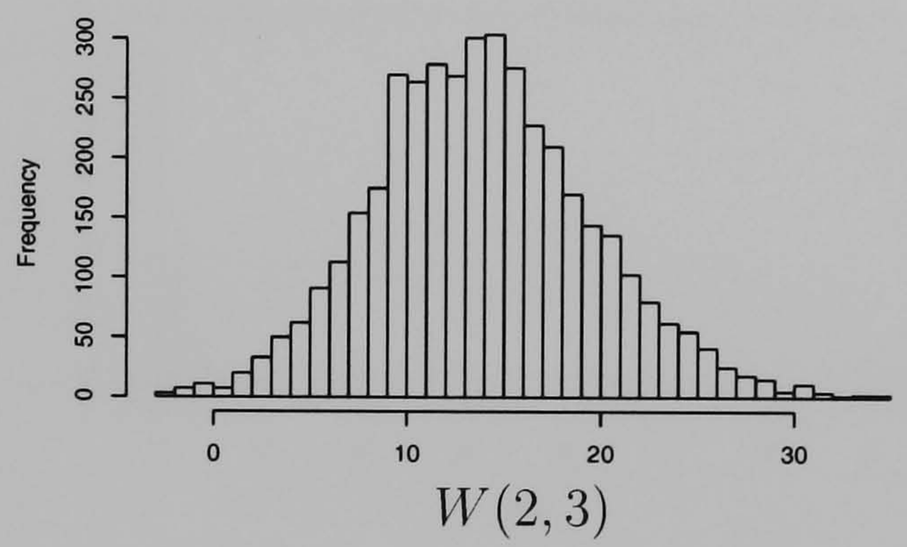


Figure 3.19: Histograms of the selected elements of the variance matrices of the locally constant dynamic linear model

narrow ranges. The histograms of the system variance, W components are as in the univariate case slightly asymmetric with slightly elongated right hand tails, with modal values lying close to the values estimated by the EM algorithm. Looking at the covariance terms in the two matrices selected elements of which are show in the trace plots Figure 3.16 and the histograms Figure 3.19 these show convergence and the samples being drawn from a distribution with the true value as it mode.

We can also look again at our multivariate locally linear DLM from Section 3.1.4, however this time instead of running an EM algorithm we will run the block Gibbs sampler algorithm as outlined above. This can be easily programmed in **sather** and we can sample from the posterior distributions for the two variance components.

We use the same 1000 simulated data points as previously.

We set some initial values for V and W and choose a non-informative Inverse Wishart prior for both variance matrices. We can then run the **sather** program *mmcll.sa*, a copy of which can be found in Appendix A which performs the block Gibbs sampling algorithm for the multivariate

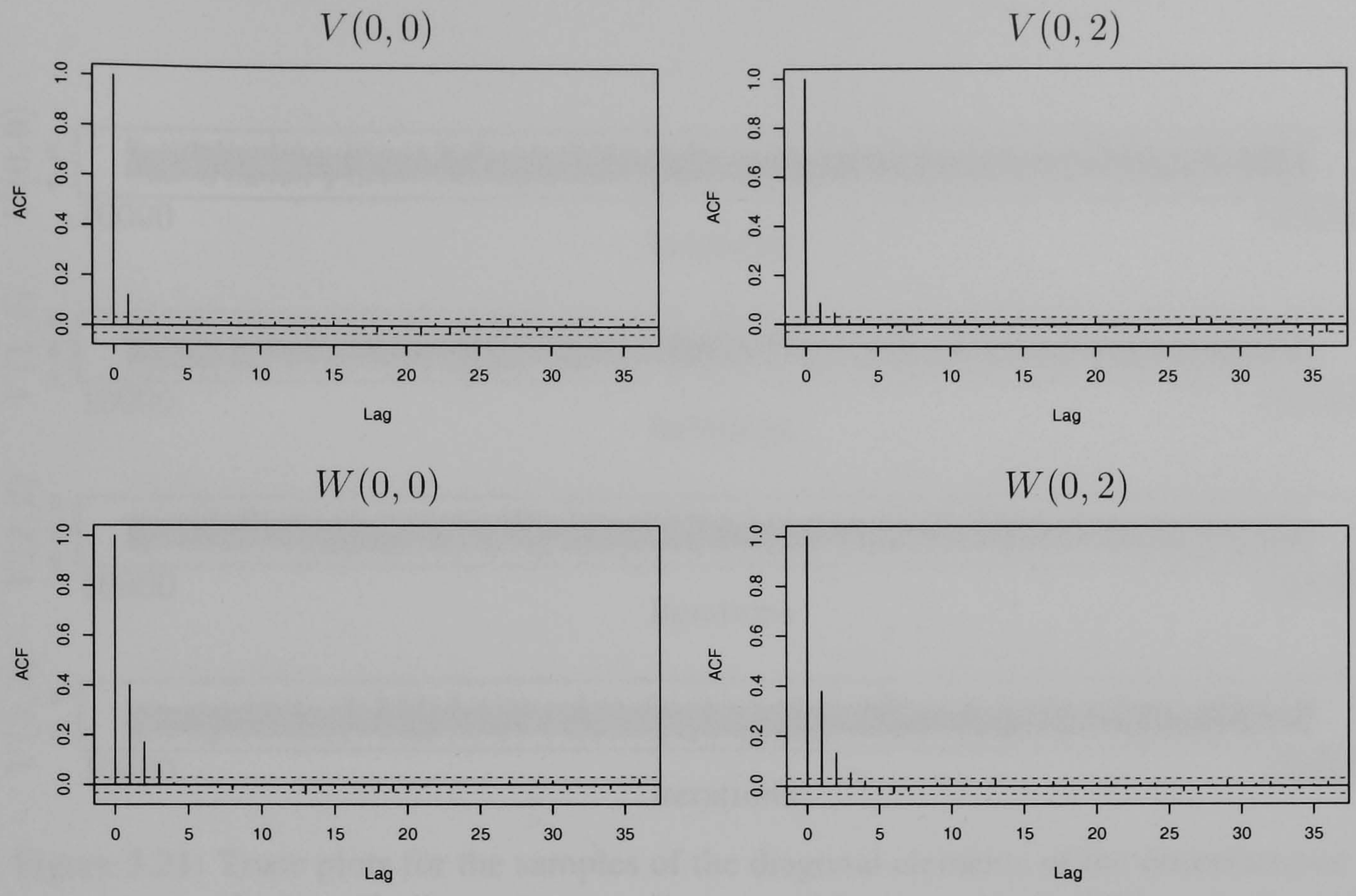


Figure 3.20: Autocorrelation plots for selected elements of V and W.

locally constant dynamic linear model.

Figure 3.21and Figure 3.22 show the trace plots for the diagonal elements of the V and W variance matrices of the locally linear dynamic linear model. These plots are based on a sampling run of $150,000$ with the first $10,000$ discarded as a burn-in. The remaining data was thinned by a factor of 10. These plots clearly show the convergence of the Gibbs sampler as they produce classic "marker pen" plots. Figure 3.23 shows the log of the samples for the diagonal elements of the Z variance matrix, the logs are taken to reduce the effects of outlying values in the samples, these suggest convergence, however looking at the autocorrelation plots Figure 3.28, suggest that the sampler could benefit from being run longer and thinned by a greater degree. Figure 3.24, Figure 3.25 and Figure 3.26 shows the histograms of the diagonal elements of the samples drawn from the marginal conditional distributions for the three variance matrices of our model. The histograms of the observational variances, V all appear approximately symmetrical centred around the true values for these components. Although not highly peaked these histograms all have relatively narrow ranges. The histograms of the system variance, W components are as in the univariate case slightly asymmetric with slightly elongated right hand

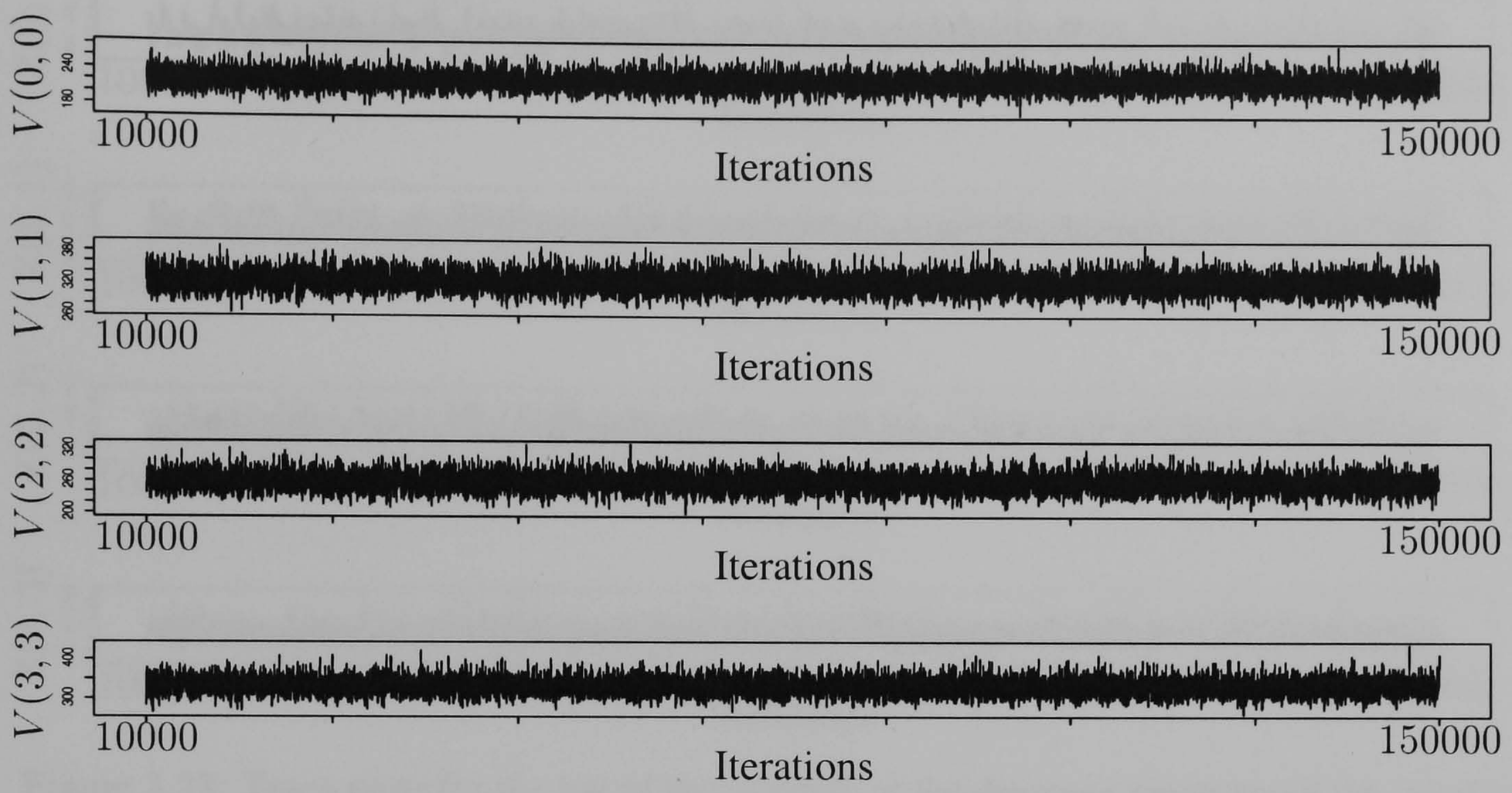


Figure 3.21: Trace plots for the samples of the diagonal elements of the observational variance components V of locally linear dynamic linear model using a block Gibbs sampling algorithm

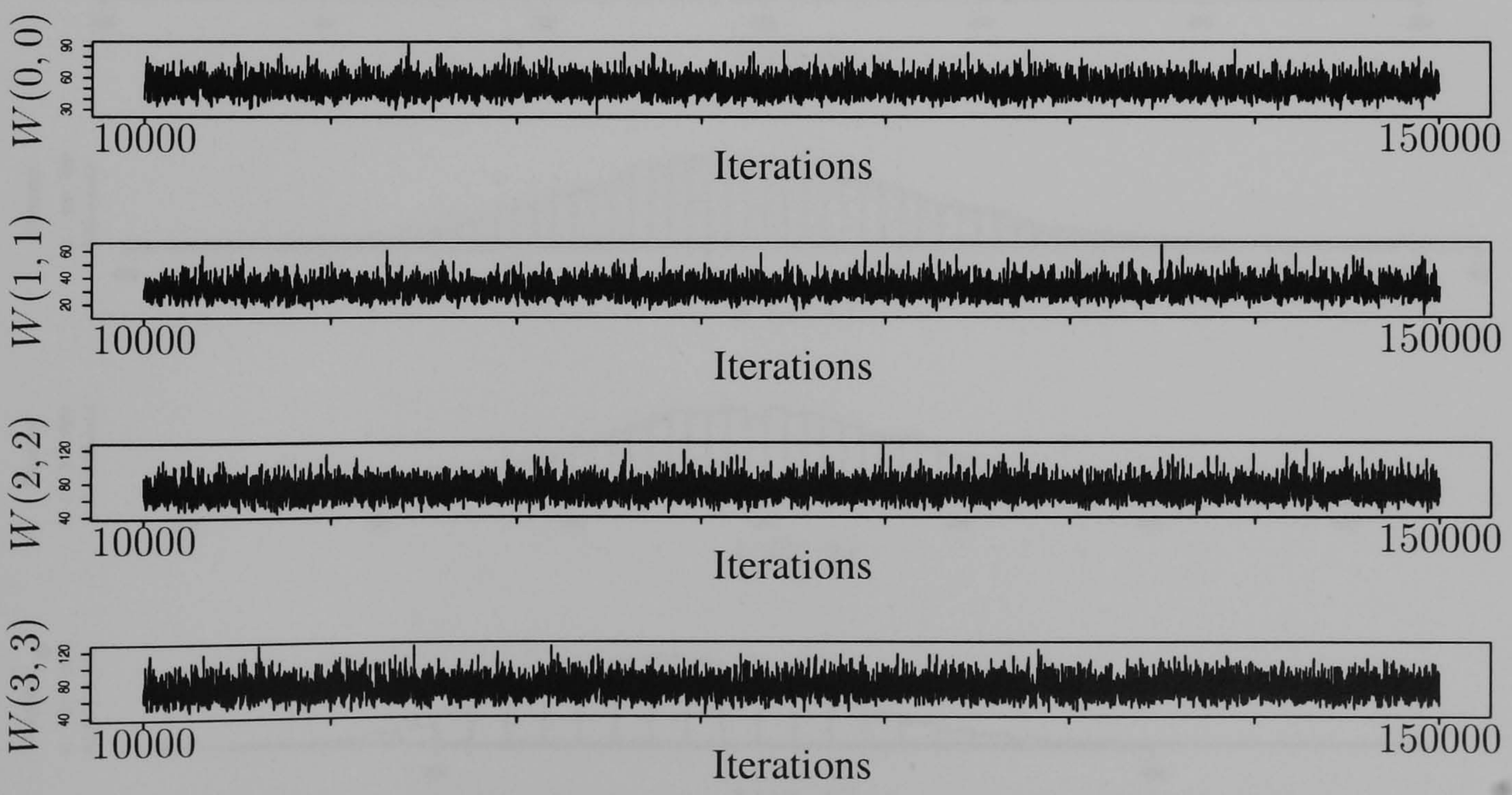


Figure 3.22: Trace plots for the samples of the diagonal elements of the system variance components W of locally linear dynamic linear model using a block Gibbs sampling algorithm

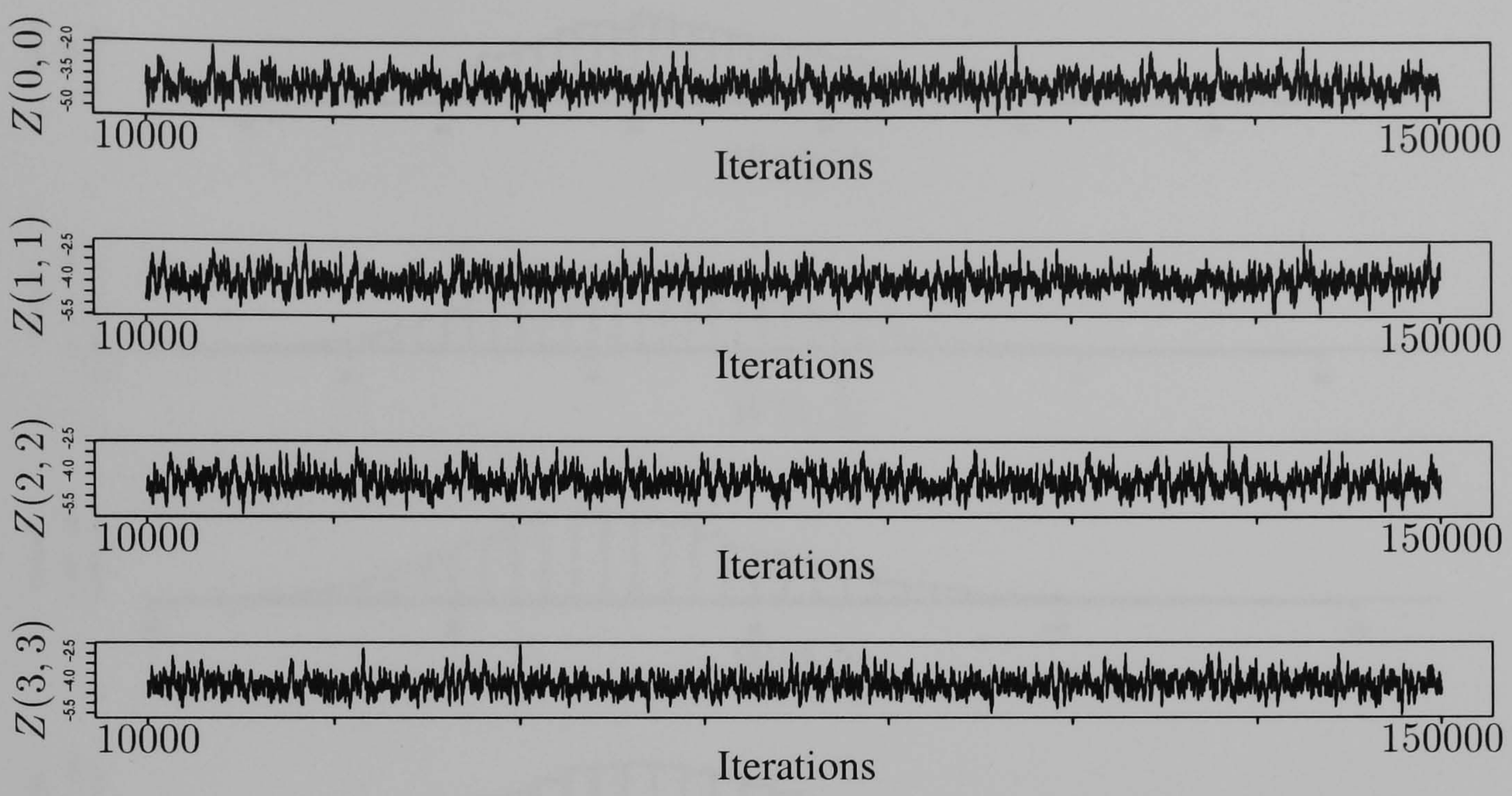


Figure 3.23: Trace plots for the log of the samples of the diagonal elements of the trend variance components Z of locally linear dynamic linear model using a block Gibbs sampling algorithm

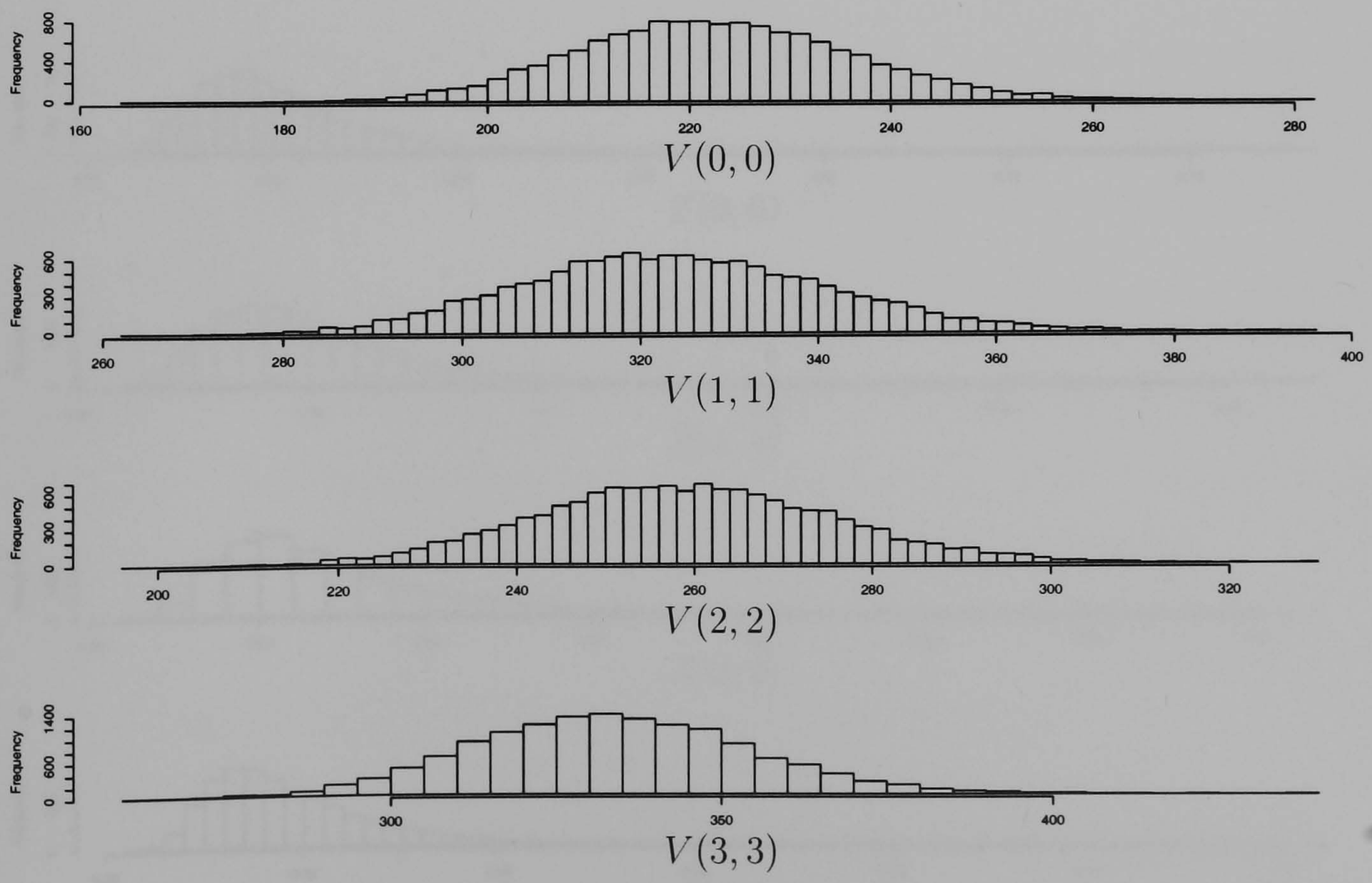


Figure 3.24: Histograms of the diagonal elements of the observational variance matrix V of the locally linear dynamic linear model.

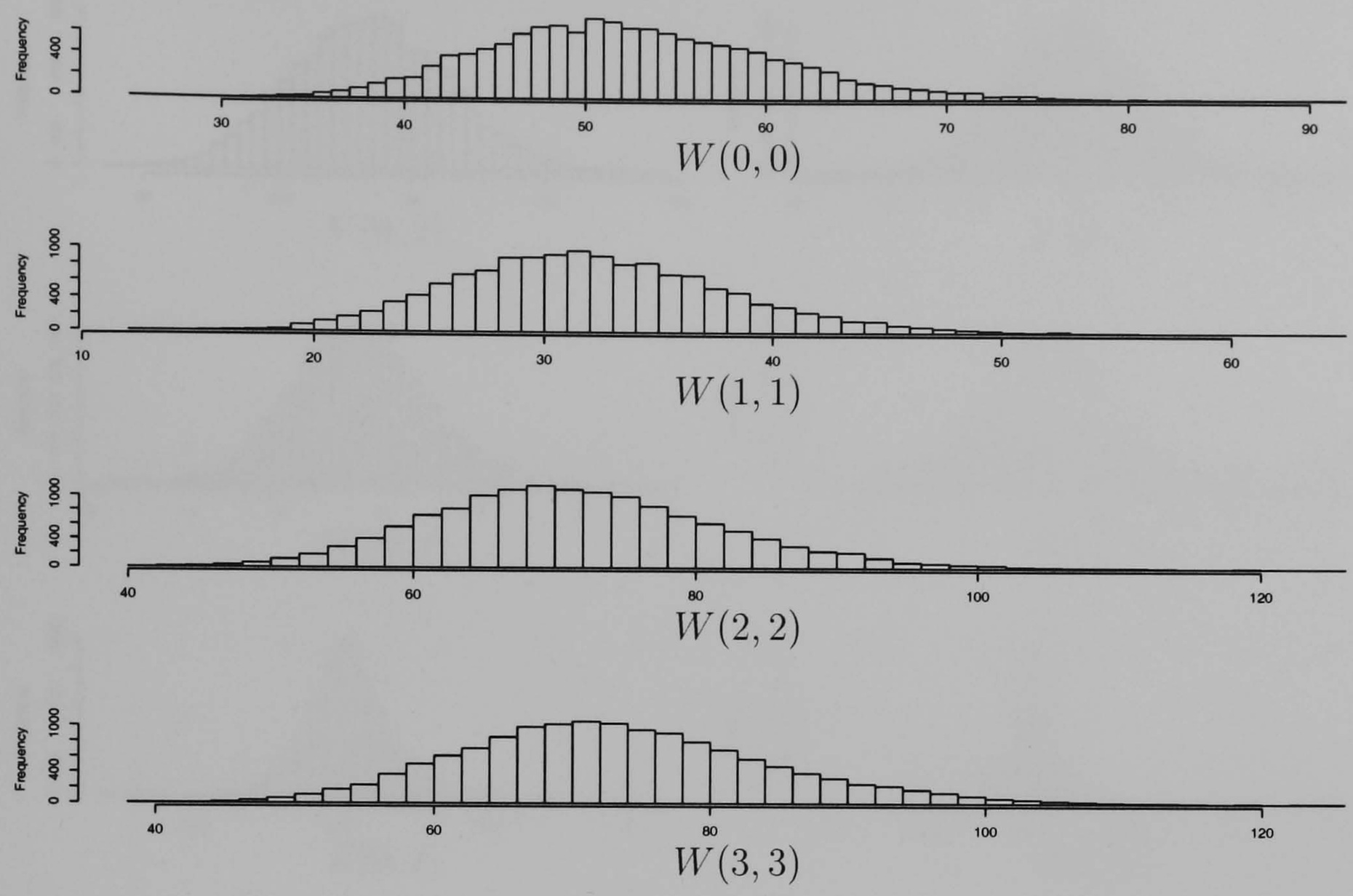


Figure 3.25: Histograms of the diagonal elements of the observational variance matrix W, of the locally linear dynamic linear model

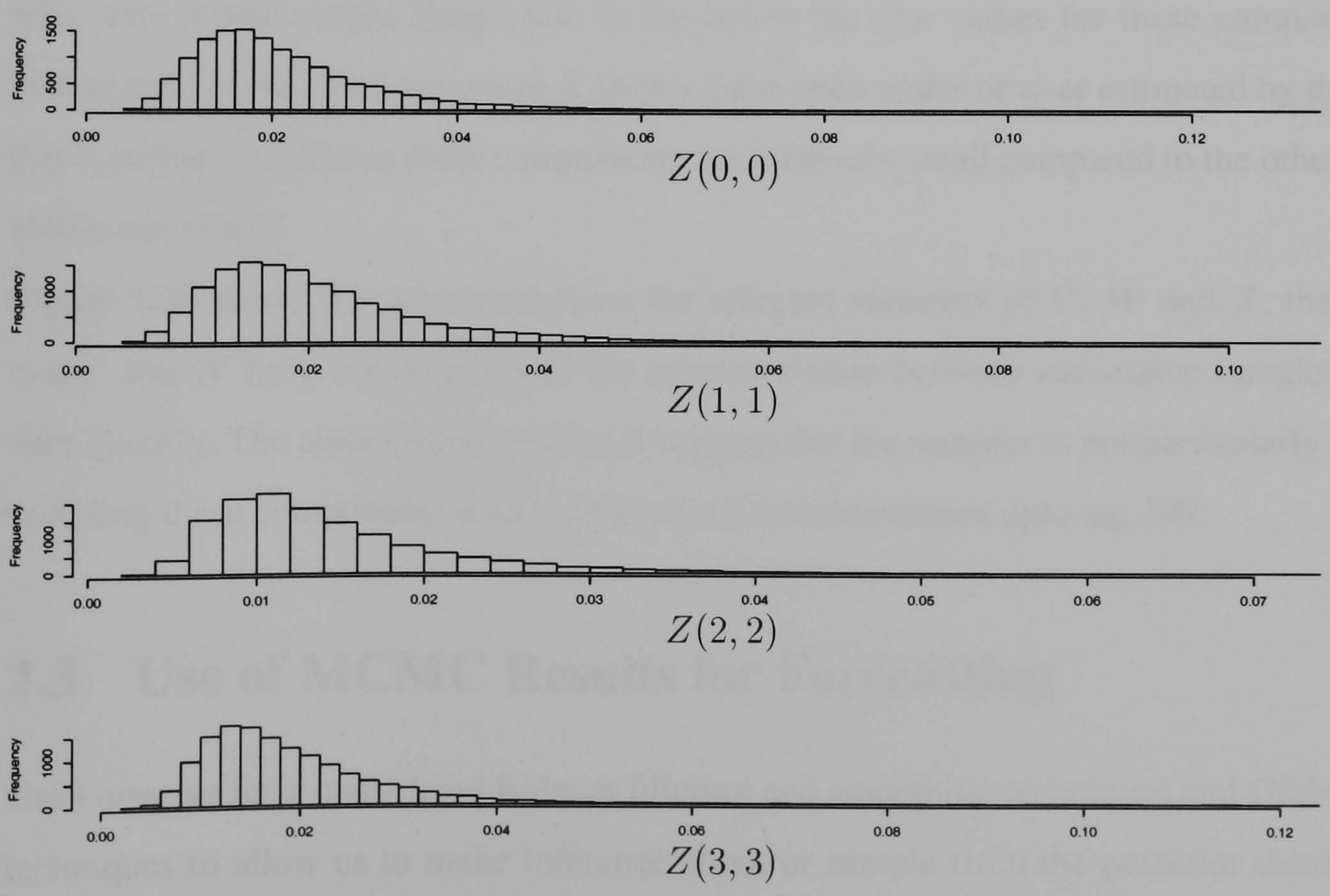


Figure 3.26: Histograms of the diagonal elements of the trend variance matrix Z, of the locally linear dynamic linear model

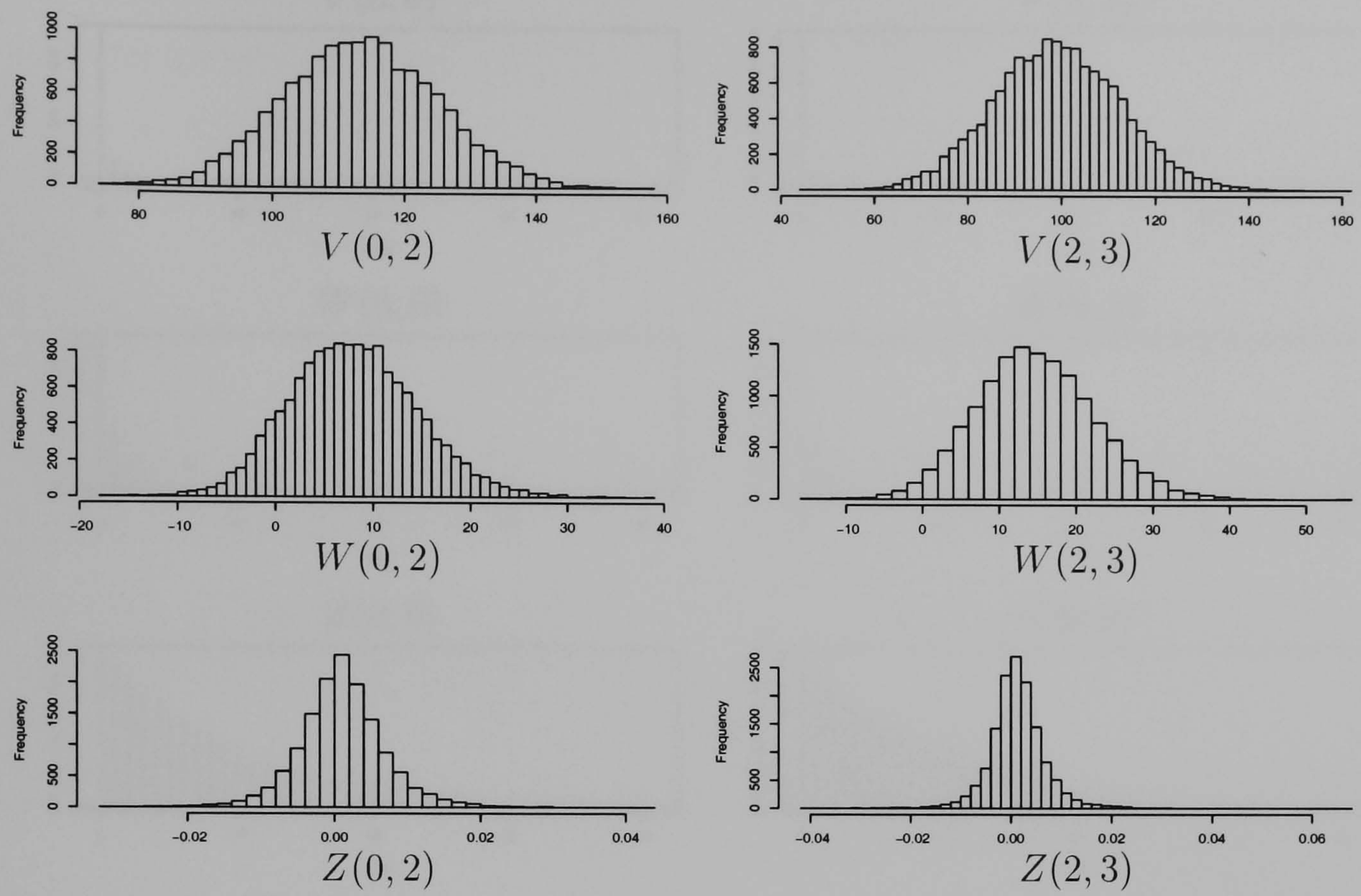


Figure 3.27: Histograms of the selected elements of the variance matrices of the locally linear dynamic linear model

tails, with modal values lying close to but below the true values for these components. The histograms for the trend variances Z shows these been under or over estimated by the sampler, this is perhaps natural as these components are relatively small compared to the other variances within our model.

Figure 3.28 shows the autocorrelations for selected elements of V, W and Z, these suggest that V and W have converged with the autocorrelation between successive samples dying off very quickly. The autocorrelations for Z suggest that the sampler is not particularly efficient at sampling these components with high levels of autocorrelation upto lag 200.

3.3 Use of MCMC Results for Forecasting

Until now we have considered Kalman filtering and smoothing techniques and Gibbs sampling techniques to allow us to make inference about or sample from the posterior distributions of the unknown and unobservable states of our model, or the variance components of the model,

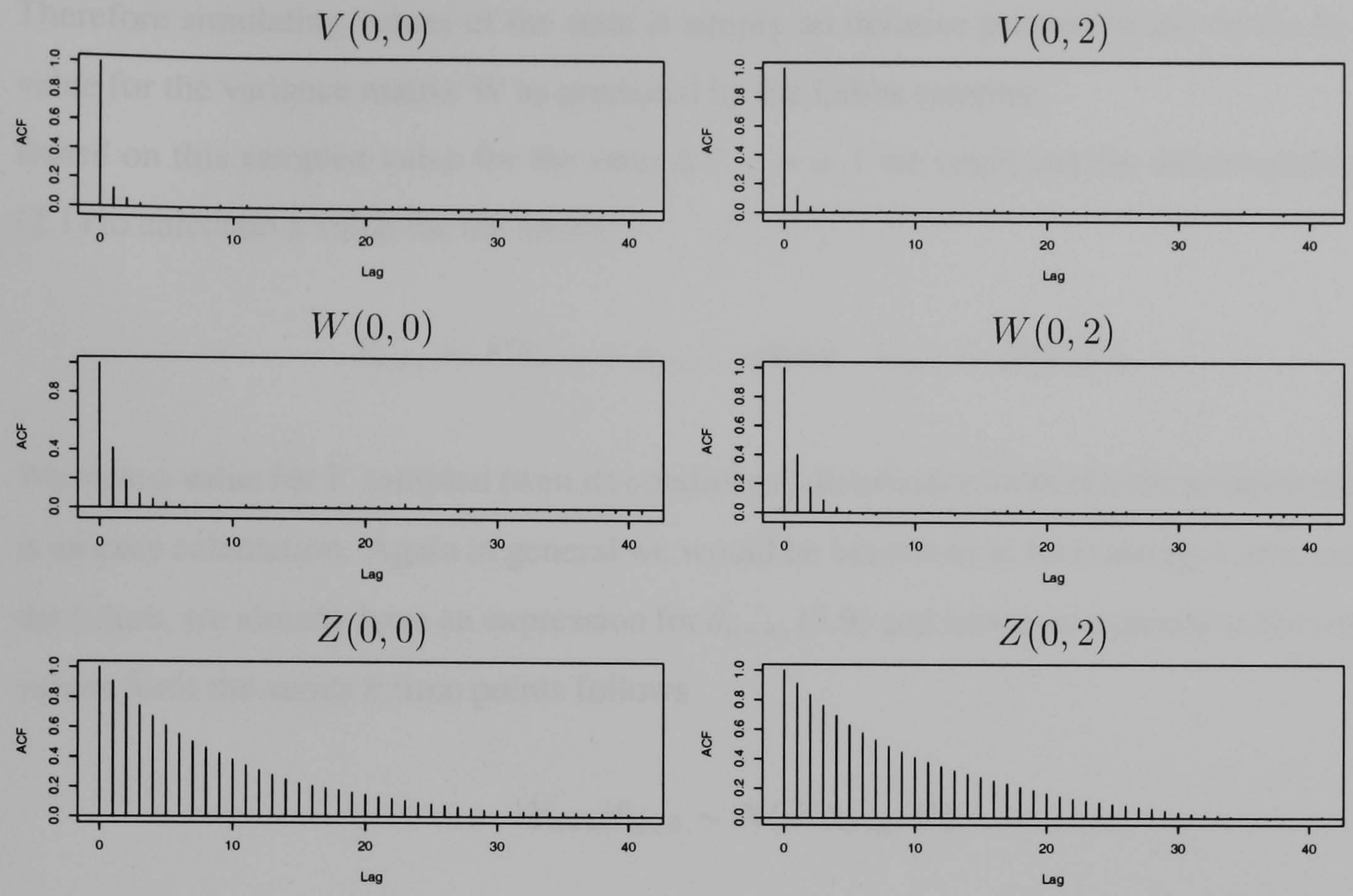


Figure 3.28: Autocorrelation plots for selected elements of V, W and Z.

within the observed time period, that is up to time $t = n$.

We might, however also be interested in saying something about the evolution of the states or of the series after the last collected data point, that is forecasting future values. We have shown we can simulate $\theta_n|D$ where $D = (Y_1, \ldots, Y_n)$ in Section 2.3.3. To forecast the value of the state one time point in the future we can use the state evolution equation (2.2) to calculate this, hence

$$\theta_{n+1} = G\theta_n + \omega_{n+1} \quad \text{where} \quad \omega_{n+1} \sim N(0, W)$$

From the simulation smoother we have a sample value for the state, θ_n and from the block Gibbs sampler a simulated value for W and hence we can calculate a value for the state, θ_{n+1}. More generally if we wish to simulate values for the state k time points into the future we use the following expression

$$\theta_{n+k}|\theta_{n+k-1} \sim N(G\theta_{n+k-1}, W). \tag{3.9}$$

Therefore simulating values of the state is simply an iterative process based on the θ_n and the value for the variance matrix W as produced by the Gibbs sampler.

Based on this sampled value for the state at $t = n + 1$ we could use the observation equation (2.1) to calculate a value for the series,

$$Y_{n+1} = F'\theta_{n+1} + \nu_{n+1} \quad \text{where} \quad \nu_{n+1} \sim N(0, V).$$

We have a value for V sampled from its conditional distribution in the Gibbs sampler and so this is an easy calculation. Again in general we would be interested in forecasting k time points into the future, we already have an expression for θ_{n+k}, (3.9) and hence an expression for simulating values form the series k time points follows

$$Y_{n+k}|\theta_{n+k} \sim N(F'\theta_{n+k}, V). \tag{3.10}$$

We have a sample of V from its marginal conditional distribution and hence we can easily simulate a value for the series at time $t = n + k$.

At each iteration of our block Gibbs sampler we produce different values for θ_n and V and W. Therefore if at each iteration of the sampler we produce forecasts for the state and the series and collect these, they will tell us about the future distribution of the series for the next k time points. We will return to this in more detail in Chapter 6 when we consider making decisions about the portfolio of shares to hold.

3.3.1 Examples

Let us again consider our univariate locally constant dynamic linear model as introduced in Section 2.4. If on this occasion we use only the first 950 data points when running the Kalman Filter and simulation smoother and then using the forecasting algorithm above which is built into the **sather** class KALMAN, we can forecast the values for the series 50 time points in to the future.

Figure 3.29 shows the last 150 data points of our series, the last 50 of which we assumed were unknown when we ran our sampling algorithm. The red line represents the median value of

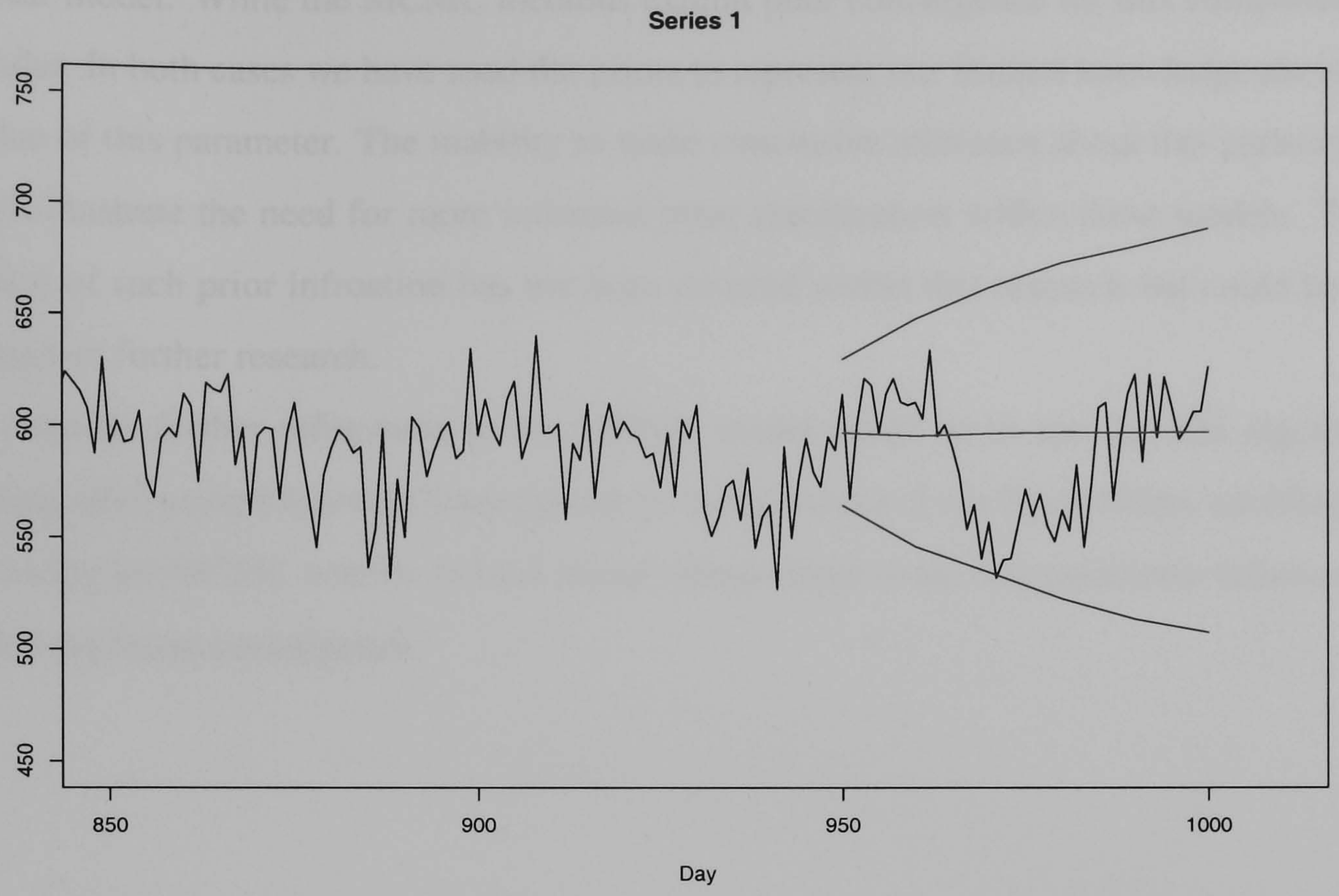


Figure 3.29: Forecasts of univariate locally constant dynamic linear model.

the samples from the forecast distribution, while the two blue lines represent the 5^{th} and 95^{th} percentiles of the sampled forecast distribution and provide a 90% predictive interval for the forecast values. We can clearly see that this predictive interval encompasses all actual values of the series over this period providing an indication of the effectiveness of this model for predicting future values of the series.

3.4 Summary

In the previous chapter we have seen how we can use a DLM structure to model a time series, in this chapter we have seen how we can use both EM and MCMC methodologies to make inference about the variance parameters of the DLM. Both methods produce results that are consistent with one another. The MCMC scheme has the advantage that we sample from a distribution for the model parameters and that we can also sample from the predictive distribution of the series, this will prove useful in Chapter 6 when we look at the portfolio selection problem. The EM algorithm has some difficulty in estimating the Z variance component of the locally

linear model. While the MCMC methods exhibit poor convergence for this component in particular. In both cases we have used flat priors to represent our limited knowledge about the true value of this parameter. The inability to make conclusive inference about this parameter could well illustrate the need for more informed prior specification within these models. The illicitation of such prior infroation has not been covered within this research but could be a useful aspect of further research.

A possible further refinement to the MCMC model could be to use the EM algorithm as a tuning step, using the results from this as the initial values of our block Gibbs sampling scheme allowing the MCMC scheme to have initial values closer to the true parameter values and hence allowing faster convergence.

Chapter 4

Stochastic Volatility

4.1 Introduction to Stochastic Volatility

To date we have been considering modelling series from a discrete time point of view, this is however only one perspective, there is a whole body of literature looking at modelling from a continuous time perspective. In this chapter we will look more closely at this perspective and see how it relates to the discrete time work we have already demonstrated.

4.1.1 Continuous Time Model

Bachelier (1900) first proposed the concept that share prices could be considered to be Markov processes related to a Wiener process. If a variable z follows a Wiener process then it has the following two properties

1. The change in z, Δz_t during a small time interval Δt is

$$\Delta z_t = \epsilon_t \sqrt{\Delta t} \quad \epsilon \sim N(0,1)$$

That is ϵ_t is a "white noise" process.

2. The values of Δz_t for any two different time intervals Δt are independent.

It follows from property 1 that $\Delta z_t \sim N(0, \Delta t)$ and from property 2 that z_t follows a Markov process. When $\Delta t \to 0$ this becomes, $dz = \epsilon_t \sqrt{dt}$. If

$$z_0 = 0 \quad \text{then} \quad z_t \sim N(0, t),$$

That is the price of a stock, Z_t is a function of time and hence the change in a stock price is also a function of the time between the two quotes, i.e.

$$\text{and} \quad z_t - z_s \sim N(0, (t-s)), \quad \text{for} \quad t > s.$$

This model for share price movements has a drift rate of zero and a volatility of one. The zero drift rate means that the expected value of z at any future time point is the same as currently. The variance of one means that the variance of the change in z in a time interval of length $t - s$, equals $t - s$. The model can however be extended to incorporate a non-zero drift rate. A *Generalised Wiener Process* for a variable x can be defined in terms of dz

$$dx = adt + bdz \tag{4.1}$$

where:-

a and b are constants

z is a standard Wiener process, so that dz is a Gaussian increment.

Initially let us only consider the first part of (4.1),

$$dx = adt \quad \text{this implies} \quad \frac{dx}{dt} = a \quad \text{or} \quad x = x_0 + at$$

where x_0 is the value of x at time zero. This implies that in a period of time T, x increases by an amount aT.

Adding back the second term bdz, this can be regarded as adding noise or variability to the model for x.

The amount of noise is proportional to the constant b, which multiplies a standard Wiener process.

Hence in a small amount of time Δt the change in the value of x is given by

$$\Delta x = a\Delta t + b\epsilon_t\sqrt{\Delta t}$$

where $\epsilon_t \sim N(0,1)$.

Hence x has an expected drift of a and a variance rate of b^2 per unit of time.

$$\text{if} \quad X_0 = 0, \quad \text{then} \quad X_t \sim N(at, b^2 t)$$

and the change in X between s and t is

$$X_t - X_s \sim N(a(t-s), b^2(t-s)), \quad \text{for} \quad t > s.$$

4.1.2 An Itô Process

Consider a *Generalised Wiener Process* where a and b are functions of the underlying variable, x and time, t

$$dx = a(x,t)dt + b(x,t)dz. \tag{4.2}$$

Hence both the expected drift rate and variance rate are liable to vary in time, this is known as an *Itô Process*. In the small time interval between t and $t+\Delta t$ the variable of interest x changes from x to $x + \Delta x$, where

$$\Delta x = a(x,t)\Delta t + b(x,t)\epsilon_t\sqrt{\Delta t}$$

this relationship requires the assumption that the drift and variance rate of x remain approximately constant during the time interval between t and $t + \Delta t$.

4.1.3 The Process for Stock Prices

It would be nice and simple if stock prices followed a generalised Wiener process, however such a model would fail to capture a key aspect of stock prices, namely that an investor's expected percentage return from a stock is independent of its price. Hence the constant expected drift rate assumption is not valid and we replace it with the assumption that the expected return is constant. The expected return of a stock is the expected drift divided by the stock price. If we designate the price of a stock at time t as S then the expected drift rate could be assumed to be μS for some constant parameter μ. Hence in a short time interval Δt the expected increase in S is $\mu S \Delta t$, the parameter μ represents the investor's expected return rate. If we assume that the variability of the percentage return on the stock price is the same regardless of the price, we can write the change in stock price as an Itô process,

$$dS = \mu S dt + \sigma S dz$$

or

$$\frac{dS}{S} = \mu dt + \sigma dz \tag{4.3}$$

where:-

- μ is expected rate of return
- σ is the volatility of the stock price.

Equation (4.3) is the most widely model for stock prices, Hull (1999). This has perhaps most famously been used in the Black-Scholes-Merton differential equation for pricing of derivatives. Knowledge of μ, the expected continuously compounded return earned by the investor per year, is assumed as is knowledge of the volatility σ which is also considered to be fixed over time. It is unlikely that volatility is in reality fixed over time and we will consider this more fully in Chapter 5, using some of the techniques we will look at more fully in Section 4.2 and Section 4.5

4.1.4 Itô's Lemma

A useful theoretical concept is Itô's Lemma. Suppose x follows an Itô Process

$$dx = a(x,t)dt + b(x,t)dz.$$

Itô's Lemma shows that G, a twice differentiable function of x and t follows the process:-

$$dG = \left(\frac{\partial G}{\partial x}a + \frac{\partial G}{\partial t} + \frac{1}{2}\frac{\partial^2 G}{\partial x^2}b^2\right)dt + \frac{\partial G}{\partial x}bdz.$$

Hence G also follows an Ito Process. How is this useful when considering stock prices? We will often consider the logarithm of share prices to account for the absolute differences in value. If we let $G = \log S$ and say S follows the Ito Process,

$$dS = \mu S dt + \sigma S dz$$

then

$$\frac{\partial G}{\partial S} = \frac{1}{S} \quad \frac{\partial^2 G}{\partial S^2} = -\frac{1}{S^2} \quad \frac{\partial G}{\partial t} = 0$$

Hence

$$dG = \left(\mu - \frac{\sigma^2}{2}\right)dt + \sigma dz.$$

Hence $\log S$ follows a Generalised Wiener Process. The change in $\log S$ between time zero and T is Normally Distributed so that

$$\log S_T - \log S_0 \sim N\left(\left(\mu - \frac{\sigma^2}{2}\right)T, \sigma^2 T\right)$$

and hence

$$\log \frac{S_T}{S_0} \sim N\left(\left(\mu - \frac{\sigma^2}{2}\right)T, \sigma^2 T\right)$$

and

$$\log S_T \sim N\left(\log S_0 + \left(\mu - \frac{\sigma^2}{2}\right)T, \sigma^2 T\right).$$

So the logarithm of the price at time T is normally distributed and related to the time passed, the starting point the expected return and the volatility. In other words the stock price has a log normal distribution.

We could consider this in discrete time, for example trading days and write the logarithm of the stock price as

$$\log S_t = \log S_{t-1} + \left(\mu - \frac{\sigma^2}{2}\right) + \omega_t \quad \text{where} \quad \omega_t \sim N(0, \sigma^2). \tag{4.4}$$

If we look at this closely it has the form of (2.2) the system or state evolution equation of our DLM models. That is the log price at time t is related to that at time $t-1$ plus some stochastic error.

From this we can see there is a direct relationship between the constant time models for stock price evolution which underlie the options pricing literature, see Hull (1999) for a more complete treatment of these, and the discrete time models we have so far applied to modelling share prices. This also carries on when we wish to consider time evolving variance terms in our dynamic linear models. Key to calculating future share prices based on (4.4) is knowledge of μ the continuously compounded return earned by the investor and most crucially σ the share price volatility. This share price volatility is crucial to the pricing of options, derivatives and other hedging instruments, see Hull (1999). Traditionally the volatility was considered to be fixed over time, this is unlikely to be true and in the next section we will consider ways in which we can model this volatility.

4.2 The Univariate Stochastic Volatility model

Many financial time series, for example the returns on shares or exchange rates exhibit changing variance. These changes tend to be serially correlated and one approach to capture this effect has been to let the conditional variance be a function of the squares of previous observations and

past variances. Such models are referred to as ARCH models, see Engle (1982) and Bollerslev (1986) for a more complete introduction to these models.

An alternative approach to modelling time evolving variance are *Stochastic Volatility* models, and it is this modelling approach we will follow in this chapter. In stochastic volatility models we establish a model containing an unobserved variance component, the logarithm of which is modelled directly as a linear stochastic process. This "is a natural discrete time approximation of the continuous time Orstein-Uhlenbeck process used in finance theory" Harvey, Ruiz, and Shephard (1994).

Let us consider the following model

$$y_t = \sigma_t \epsilon_t, \qquad \text{where} \qquad \epsilon_t \sim N(0,1) \tag{4.5}$$

is a "white noise" process The series of interest, y_t is a univariate series comprising some white noise multiplied by a standard deviation σ_t. Note that $\mathrm{E}(y_t) = 0$ and $\mathrm{Var}(y_t|\sigma_t) = \sigma_t^2$. Letting $\alpha_t = \log {\sigma_t}^2$ we can write this as a stochastic model,

$$y_t = \epsilon_t \exp\left(\frac{\alpha_t}{2}\right), \qquad \text{where} \qquad \epsilon_t \sim N(0,1). \tag{4.6}$$

If we then consider $\log {y_t}^2$, the properties of the model (4.5) can be clearly seen

$$\log {y_t}^2 = \alpha_t + \log {\epsilon_t}^2. \tag{4.7}$$

The observed series y_t is related to an unknown and unobserved state, α_t with some stochastic noise. If we approximate $\log \epsilon_t^2$ by a normal random variable this clearly follows the form of our dynamic linear model observation equation (2.1). The mean and variance of $\log {\epsilon_t}^2$ are known results, the mean being -1.27 and variance $\frac{\pi^2}{2}$.

It is possible to model the evolution of α_t through a simple AR(1) process, to give the model a dynamic element,

$$\alpha_t = \phi \alpha_{t-1} + \eta_t \qquad \text{where} \qquad \eta_t \sim N(\mu, {\sigma_\eta}^2). \tag{4.8}$$

(4.8) is known as an uncentred model for the evolution of α_t. For reasons of computational efficiency within the MCMC sampling algorithm we will follow the example of Pitt and Shephard (1999) and use the centred model (4.9),

$$\alpha_t = \mu + \phi(\alpha_{t-1} - \mu) + \eta_t \qquad \text{where} \qquad \eta_t \sim N(0, {\sigma_\eta}^2). \tag{4.9}$$

The model defined by (4.6) and (4.9) is denoted, using the notation of Pitt and Shephard (1999), as $y_t \sim ISV_n(\phi; \sigma_\eta; \mu)$, that is a series $y = (y_1, \ldots, y_n)'$ arises from a stochastic volatility model, conditionally independent of any other series. It is this form of univariate model which we will consider initially and which provides the basis for the multivariate extensions that we will look at in Section 4.5. In Chapter 5 we will use this as a more flexible model for ν_t and ω_t from our dynamic linear model.

We now have a model, the next natural step is to make inference about the unknown parameters of our model. We have adopted a Bayesian paradigm in this thesis and we will continue that here, looking to make inference on the ISV_n model using Markov Chain Monte Carlo techniques and developing some of the methodologies introduced in Chapter 2 and Chapter 3.

4.2.1 Parameter Estimation of the Univariate Stochastic Volatility Model

Consider the model defined by (4.7) and (4.9). We have stated that the mean and variance of $\log {\epsilon_t}^2$ are known results. Initially let us assume that $\log {\epsilon_t}^2$ is Gaussian, i.e. $\log {\epsilon_t}^2 \sim N(-1.27, \pi/2)$, following the example of Harvey, Ruiz, and Shephard (1994) who used this assumption to allow Kalman filtering and smoothing to produce quasi-maximum likelihood estimators of the model parameters. This assumption of normality allows us to sample from the state α_t conditioning on the data and all other model parameters using the Kalman filtering and smoothing or simulation smoothing techniques discussed in Chapter 2. To perform the Kalman filtering and smoothing we need to write the model in the following form

$$\log {y_t}^2 + 1.27 = \alpha_t + \xi_t \qquad \text{Where} \qquad \xi_t \sim N(0, \frac{\pi^2}{2})$$

$$\alpha_t = \mu + \phi(\alpha_{t-1} - \mu) + \eta_t \qquad \text{Where} \qquad \eta_t \sim N(0, {\sigma_\eta}^2)$$

Inference on α_t

The Kalman filter and simulation smoother can be run on this model after the appropriate manipulation of the data. A new **sather** class ISV has been written to allow easy analysis of these models and this contains the necessary Kalman filtering and simulation smoothing libraries. By running these we sample from the states α_t conditioning on all the data and other model parameters. For references to the ISV class in **sather** see Appendix A.

Update of σ_η

From the model we know that $\eta_t \sim N(0, {\sigma_\eta}^2)$ and hence we can use a standard conjugate Gamma update on the precision. Assuming a prior of ${\sigma_\eta}^{-2} \sim \Gamma(a, b)$, using Bayes theorem (1.1) to combine this with the likelihood we can write the conditional distribution for ${\sigma_\eta}^2$ as

$$\sigma_\eta^{-2}|\cdot \sim \Gamma\left(a + \frac{n}{2}, b + \frac{1}{2}\sum_{t=1}^{n} {\eta_t}^2\right). \tag{4.10}$$

From this full conditional we can sample a value for σ_η. The necessary summation is provided in the KALMAN class, and replicated within the **sather** class ISV.

Update of μ

In order to update μ we need to rearrange (4.9) to get an expression in terms of μ.

$$\alpha_t = \mu + \phi(\alpha_{t-1} - \mu) + \eta_t \qquad \text{where} \qquad \eta_t \sim N(0, {\sigma_\eta}^2)$$

$$\Psi_t = \alpha_t - \phi\alpha_{t-1} = \mu(1-\phi) + \eta_t$$

$$\Psi_t = \alpha_t - \phi\alpha_{t-1} \sim N(\mu(1-\phi), {\sigma_\eta}^2).$$

We now have an expression for $\alpha_t - \phi\alpha_{t-1}$ in terms of μ. We can calculate the likelihood for this in the normal way.

$$\mathrm{L}(\Psi) = \prod_{t=1}^{n} \frac{{\sigma_\eta}^{-1}}{\sqrt{2\pi}} \exp\left(-\frac{\sigma_\eta^{-2}}{2}\left(\Psi_t - \mu\left(1-\phi\right)\right)^2\right) \tag{4.11}$$

Placing a Normal prior on μ i.e. $\mu \sim N(a, b)$, we can combine this with our likelihood (4.11) using Bayes theorem (1.1) to obtain the following posterior

$$\mu|\cdot \propto \exp\left(-\frac{\sigma_\eta^{-2}}{2}\sum_{t=1}^{n}\left(\Psi_t - \mu\left(1-\phi\right)\right)^2 - \frac{b^{-1}}{2}\left(\mu - a\right)^2\right)$$

Ignoring terms not including μ and re-arranging we get

$$\mu|\cdot \propto \exp\left(-\frac{1}{2}\left(n\sigma_\eta^{-2}\left(1-\phi\right)^2 + b^{-1}\right)\mu^2 - 2\left(\sigma_\eta^{-2}\sum_{t=1}^{n}\left(\Psi_t - \phi\Psi_t\right) + b^{-1}a\right)\mu\right)$$

Letting

$$v^* = n\sigma_\eta^{-2}\left(1-\phi\right)^2 + b^{-1}$$
$$m^* = \sigma_\eta^{-2}\sum_{t=1}^{n}\left(\Psi_t\left(1-\phi\right)\right) + b^{-1}a$$

giving

$$\mu|\cdot \propto \exp\left(-\frac{1}{2}v^*\left(\mu^2 - 2\frac{m^*}{v^*}\mu\right)\right).$$

Completing the square, the full conditional for $\mu|\cdot$ is

$$\mu|\cdot \sim N(m, v) \tag{4.12}$$

where

$$m = v\left(\frac{a}{b} + \frac{\sum_{t=1}^{n}(\alpha_t - \phi\alpha_{t-1})(1-\phi)}{{\sigma_\eta}^2}\right) \qquad v = \left(\frac{1}{b} + n\frac{(1-\phi)^2}{\sigma_\eta^2}\right)^{-1}$$

We can sample μ from its full conditional distribution. Again the necessary summations are easily calculated and a function is provided within the **sather** ISV module to return these.

Update of ϕ

We could assume a semi-conjugate normal prior for ϕ and hence we would be able to sample from the full conditional using a simple Gibbs sampling update. Rearranging (4.9) and letting$\Psi_t = \alpha_t - \mu$

$$\begin{aligned} \alpha_t &= \mu + \phi(\alpha_{t-1} - \mu) + \eta_t \\ \Rightarrow \Psi_t &= \phi\Psi_{t-1} + \eta_t. \end{aligned}$$

Hence

$$\Psi_t \sim N(\phi\Psi_{t-1}, {\sigma_\eta}^2). \tag{4.13}$$

Unfortunately this leads to identifiability and convergence problems, due to the fact that such a semi conjugate normal prior is not restricted to the range $(0, 1)$ which would capture the prior belief that volatility process is stationary with positive dependence. This belief of stationarity and positive depends is bourne out by empirical studies dating back to Fama (1965).
To overcome this we assume a Beta prior for ϕ, i.e. $\pi(\phi) \sim \beta(a, b)$ as this is a flexible class of distributions in the range $(0, 1)$. This is a non-conjugate prior and so simple Gibbs sampling is not possible. Using (4.13) we can find the likelihood of ϕ

$$\begin{aligned} \mathrm{L}(\phi|\Psi_1 \ldots \Psi_n, \sigma_\eta, y) &\propto \prod_{t=2}^{n} \exp\left(\frac{-1}{2\sigma_\eta}\left(\Psi_t - \phi\Psi_{t-1}\right)^2\right) \\ &\propto \exp\left\{\frac{-1}{2\sigma_\eta}\left(\sum_{t=2}^{n}\Psi_t^2 - 2\phi\sum_{t=2}^{n}\Psi_t\Psi_{t-1} + \phi^2\sum_{t=2}^{n}\Psi_{t-1}^2\right)\right\}. \end{aligned}$$

We can ignore the multiplicative terms not involving ϕ, manipulate and complete the square which leaves us with a likelihood for ϕ which is normal of the following form

$$\mathrm{L}(\phi|\cdot) = N(m, v)$$

where

$$m = v\left(\frac{\sum_{t=2}^{n}(\alpha_t - \mu)(\alpha_{t-1} - \mu)}{{\sigma_\eta}^2}\right) \quad v = \left(\frac{\sum_{t=2}^{n}(\alpha_{t-1} - \mu)^2}{\sigma_\eta^2}\right)^{-1}.$$

We have assumed a beta prior for ϕ i.e. $\pi(\phi) \sim \beta(a, b)$

$$\pi(\phi) = \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)}\phi^{(a-1)}(1-\phi)^{(b-1)} \qquad 0 < \phi \leq 1.$$

Using Bayes theorem (1.1) we can combine the prior and the likelihood to obtain the following conditional posterior for ϕ

$$\pi\left(\phi|\cdot\right) \propto \exp\left(\frac{-1}{2v}\left(\phi - m\right)^2\right)\phi^{(a-1)}(1-\phi)^{(b-1)} \tag{4.14}$$

where m and v are from the likelihood above.

This is of course not a simple distribution from which we can directly sample a value for ϕ, however we can use a random walk Metropolis-Hastings scheme to sample from this posterior. A random walk Metropolis-Hastings scheme relies upon the symmetric nature of the random walk to reduce the acceptance probability to $min\{1, \pi(\phi^*|\cdot)/\pi(\phi|\cdot)\}$, where ϕ^* is a new value proposed for ϕ from the random walk and ϕ is the current value. The symmetry of the random walk means that no transition kernel is required in the acceptance probability. That is the proposed value is simulated from a distribution which has the current value as its expectation and a small constant variance. The algorithm for the update of ϕ is as follows

1. Simulate a proposed ϕ called ϕ^* from a random walk based on the current value of ϕ and some constant small variance. That is simulate a value from $N(\phi, \sigma^2)$ where σ^2 is some chosen small variance.

2. Calculate the acceptance probability using the posterior we have for ϕ and compare with a random uniform and accept the new value if this is smaller than the acceptance probability.

For stability reasons at the computational level it is necessary to work with the log of the accep-

tance probability, which is as follows, and compare this with a log uniform value

$$\log(A) = \log(\pi(\phi^*|\cdot)) - \log(\pi(\phi|\cdot)).$$

We only need the distribution up to the constant of proportionality as this will drop out in the acceptance probability. Hence the logged acceptance probability with which we make the comparison is

$$\begin{aligned}
\log(A) &= \left(\frac{-1}{2v}(\phi^* - m)^2\right) + (a-1)\log\phi^* + (b-1)\log(1-\phi^*) \\
&\quad - \left[\left(\frac{-1}{2v}(\phi - m)^2\right) + (a-1)\log\phi + (b-1)\log(1-\phi)\right] \\
&= \frac{-1}{2v}\left((\phi^* - m)^2 - (\phi - m)^2\right) + (a-1)(\log\phi^* - \log\phi) \\
&\quad + (b-1)\left(\log(1-\phi^*) - \log(1-\phi)\right).
\end{aligned}$$

The **sather** class ISV has a function to perform this update and return a sample from the conditional of ϕ.

The Updating algorithm

As with the dynamic linear model in Section 3.2.1 we can establish a block Gibbs sampler to sample $\alpha, \sigma_\eta, \mu, \phi|Y$ by alternatively sampling $\alpha_t|\sigma_\eta, \mu, \phi, Y$, $\sigma_\eta|\alpha, \mu, \phi, Y$, $\mu|\alpha, \sigma_\eta, \phi, Y$ and $\phi|\alpha, \sigma_\eta, \mu, Y$.

1. Run the Kalman filter and simulation smoother with initial values for all model parameters and sample from the state $\alpha|\sigma_\eta, \mu, \phi, Y$.

2. Sample from the full conditional for σ_η, (4.10), given the states and all other model parameters, i.e. sample $\sigma_\eta|\alpha, \mu, \phi, Y$.

3. Sample from the full conditional for μ, (4.12), given the states and all other model parameters, i.e. sample $\mu|\alpha, \sigma_\eta, \phi, Y$

4. Sample from the full conditional for ϕ, (4.14) using a random walk Metropolis-Hastings scheme, given the states and all other model parameters, i.e. sample $\phi|\alpha, \sigma_\eta, \mu, Y$.

5. Re-run the Kalman filter simulation smoother using the new values for the model parameters to sample from the state $\alpha|\sigma_\eta, \mu, \phi, Y$.

6. Repeat steps 2 to 5 until convergence.

4.3 Correcting for the Normal Approximation

In the above algorithm the samples of α produced by the Kalman filter simulation smoothing routine are dependent upon the Gaussian approximation used for $\log \epsilon_t{}^2$. It is possible to correct for this approximation using a mixture of Normals as per Kim, Shephard, and Chib (1998); however we have chosen to adopt the approach of Pitt and Shephard (1999) incorporating a Metropolis-Hastings step into our algorithm to correct for this approximation.

We know we have a target density, $p(\alpha|\cdot)$ under the true model, so we can propose values for α using the normal approximation and only accept a proportion of these proposed moves using a Metropolis-Hastings acceptance probability. The proposed value is the value of the state α, generated by the Kalman filter simulation smoother under the current values of each of the model parameters. We will call this proposed value α^* and the normal approximation from which it is produced we will denote by $q(\alpha|\cdot)$. The Metropolis-Hastings acceptance probability is $\min\{1, A\}$where A is

$$A = \frac{p(\alpha^*|\cdot)}{q(\alpha^*|\cdot)} \Big/ \frac{p(\alpha|\cdot)}{q(\alpha|\cdot)}.$$

Since from Bayes theorem (1.1)

$$q(\alpha|y) \propto \pi(\alpha)\mathrm{L}_q(\alpha|y)$$

$$p(\alpha|y) \propto \pi(\alpha)\mathrm{L}_p(\alpha|y)$$

the priors in each fraction of the acceptance probability are the same and hence they cancel out, and we are left with an acceptance probability that is a ratio of two Likelihoods.

For reasons of computational stability we will consider the log of the acceptance probability,

$$\log A = (\log p(\alpha^*|\cdot) - \log q(\alpha^*|\cdot)) - (\log p(\alpha|\cdot) - \log q(\alpha|\cdot))$$

which simplifies to

$$\log A = (\log \mathrm{L}_p(\alpha^*|\cdot) - \log \mathrm{L}_q(\alpha^*|\cdot)) - (\log \mathrm{L}_p(\alpha|\cdot) - \log \mathrm{L}_q(\alpha|\cdot))$$

where α^* is an new proposed value from the Kalman filter simulation smoother routine which is being compared with the current value of α and being accepted or rejected according to $\log A$. In order to calculate $\log A$ we require the likelihoods under the true model and the approximation.

The True Likelihood

Since $y_t = \epsilon_t \exp(\alpha_t/2)$ where $\epsilon_t \sim N(0,1)$ then $y_t \sim N(0, \exp(\alpha_t))$ therefore the likelihood of α_t is

$$\begin{aligned}\mathrm{L}_p(\alpha|\cdot) &= \prod_{t=1}^{n} \frac{1}{\sqrt{2\pi \exp(\alpha_t)}} \exp\left(-\frac{1}{2\exp(\alpha_t)}(y_t - 0)^2\right) \\ &= \left(\frac{1}{\sqrt{2\pi}}\right)^n \sum_{t=1}^{n} \exp(-\alpha_t/2) \exp\left(-\frac{1}{2}\sum_{t=1}^{n} y_t^2 \exp\{-\alpha_t\}\right)\end{aligned}$$

Taking logarithms to get the required log likelihood,

$$\log \mathrm{L}_p(\alpha|\cdot) = -\frac{n}{2}\log 2\pi - \frac{1}{2}\sum_{t=1}^{n}\left(\alpha_t + y_t^2 \exp\{-\alpha_t\}\right) \tag{4.15}$$

Normal Likelihood

Since $\log {y_t}^2 = \alpha_t - 1.27 + \xi_t$ where $\xi_t \sim N(0, \pi^2/2)$, then $\log y_t^2 \sim N(\alpha_t - 1.27, \pi^2/2)$. The likelihood is then,

$$\begin{aligned} \mathrm{L}_q(\alpha|\cdot) &= \prod_{t=1}^{n} \frac{1}{\sqrt{\pi^3}} \exp\left(\frac{-1}{\pi^2}\left(\log y_t^2 - (\alpha_t - 1.27)\right)^2\right) \\ &= \left(\frac{1}{\pi^{\frac{3}{2}}}\right)^n \exp\left(\frac{-1}{\pi^2}\sum_{t=1}^{n}\left(\log y_t^2 - (\alpha_t - 1.27)\right)^2\right). \end{aligned}$$

Taking logs to get the log likelihood required gives

$$\log \mathrm{L}_q(\alpha|\cdot) = -\frac{3n}{2}\log \pi - \frac{\sum_{t=1}^{n}\left(\log y_t^2 - (\alpha_t - 1.27)\right)^2}{\pi^2}. \tag{4.16}$$

The acceptance probability is simply calculated by evaluating these two log likelihoods with both the current and proposed values for the state $\alpha = (\alpha_1, \ldots, \alpha_t)$. This is then compared with the log of a randomly generated uniform number and the proposed move is accepted if the value for $\log A$ is greater than this value.

To clarify this let us consider the following algorithm for this corrective step

1. Generate a sample from the state, α given the data and all other model parameters.
2. Evaluate the acceptance probability A and compare with a random uniform.
3. If A is greater than the uniform accept the proposed value for the state, α.
4. If A is less than the uniform reject the proposed value for the state, α and keep the current value.

This correction has been incorporated into the **sather** class ISV allowing the option for the modeller to use this correction.

Pitt and Shephard (1997) have noted the problems with convergence of this type of algorithm in the stochastic volatility context, in that few moves are accepted when trying to update the whole series and hence convergence is slow. The solution they propose and that which is adopted here is to use blocking of the data series and perform the sampling from the state in blocks which are

less than the size of the whole data set. Each block is sampled and accepted or rejected under the Metropolis-Hastings step in turn. The **sather** class ISV incorporates this step breaking the state into even length blocks and running the Kalman filter simulation smoother on each block and accepting or rejecting each block sequentially, eventually returning a whole state vector, α. This Metropolis-Hastings correction for the Normal approximation leads to a new updating algorithm.

1. Run the Kalman filter and simulation smoother with initial values for all model parameters and sample from the state α^* in blocks, to give proposed values for the state.

2. Calculate the likelihoods, (**??**) and (4.16) and evaluate the acceptance probability A and accept the new values for the state, α with this probability.

3. Sample from the full conditional for σ_η, (4.10), given the states and all other model parameters.

4. Sample from the full conditional for μ, (4.12), given the states and all other model parameters.

5. Sample from the full conditional for ϕ, (4.14) using a random walk Metropolis-Hastings scheme, given the states and all other model parameters.

6. Re-run the Kalman filter simulation smoother using the new values for the model parameters to sample from the state α in blocks.

7. Repeat steps 2 to 6 until convergence.

4.4 Examples

We generated 1000 data points from an $ISV(0.9, 0.01, 1.5)$ model and then run the **sather** program *exampleisv.sa* a copy of which can be found in Appendix A. This was run for 150,000 iterations and the first 5,000 discarded as a burn-in and the remaining output thinned by a factor of 10. A flat $\Gamma(0, 0.0001)$ prior was set for σ_η, a $N(0, 100)$ prior was set for μ while a slightly stronger $\beta(20, 3)$ prior was set for ϕ. This prior has a mean of 0.87 and a standard deviation

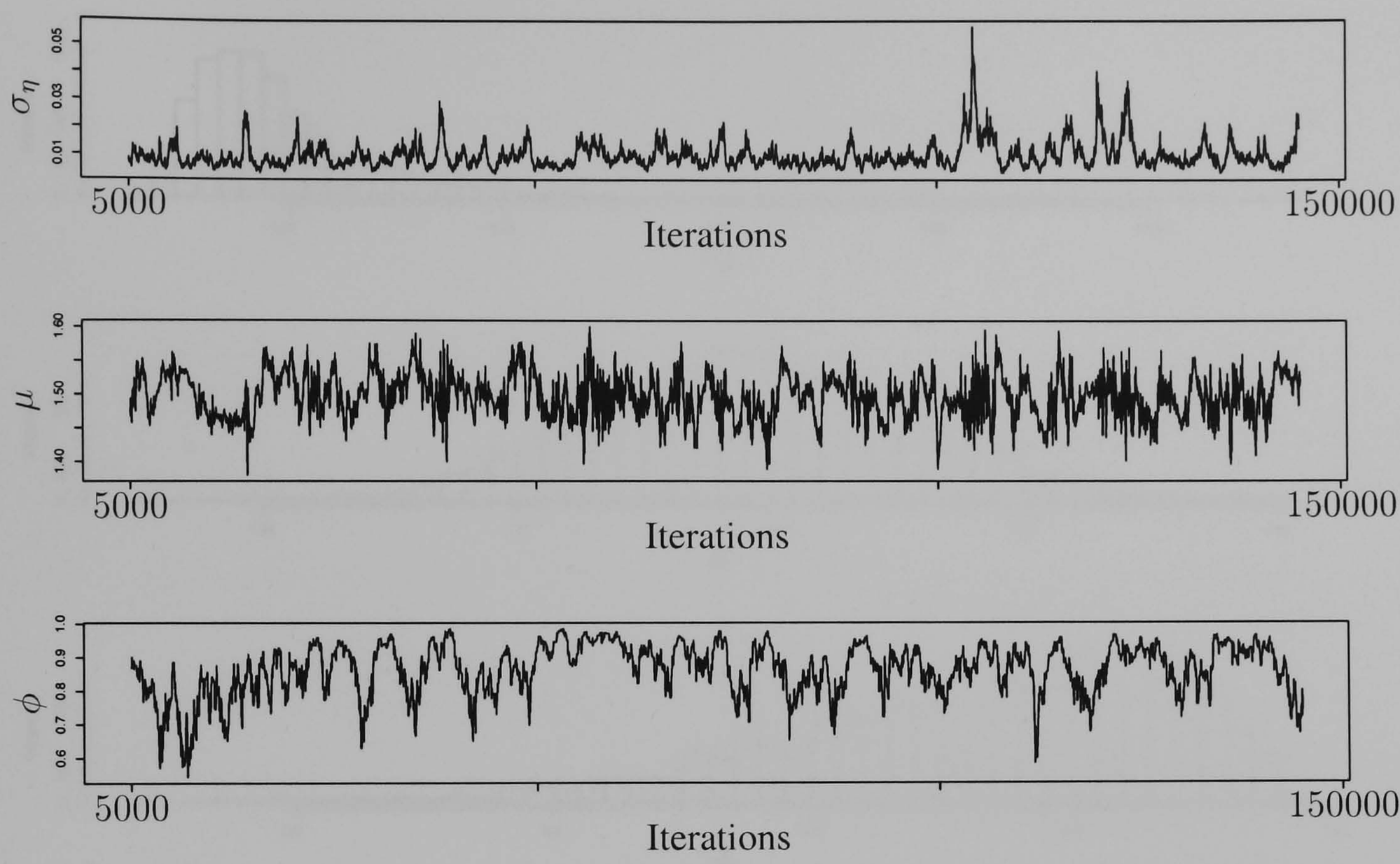


Figure 4.1: Trace plots of samples from the three parameters of the ISV model

of 0.069, this was chosen to reflect the strong prior belief that the series has strong positive dependence, The model works better with strong prior specification of this parameter.

Figure 4.1 shows the trace plots of the samples from the three parameters of the the ISV example. They suggest that the model perhaps has not yet converged, looking at the autocorrelation plots Figure 4.5 shows a long tail off in autocorrelations demonstrating poor mixing of the sampler.

Figure 4.2 shows the histograms of the samples from the block Gibbs sampler. Although the traces plots do not suggest convergence these show that the sampler with flat priors appears to have been sampling from a distribution with the true values as it modes, as the histograms are centred around the true values for these parameters. A longer run of the sampler with further thinning might be beneficial to provide more independent samples.

Figure 4.3 and Figure 4.4 show trace plots and histograms for selected values of the state α at time $t = 10, 100, 300$ and 400. These are known to be $1.46824, 1.47954, 1.48285$ and 1.50247 respectively, the traces plots suggest convergence and the histograms show these samples have modal values close to the true values.

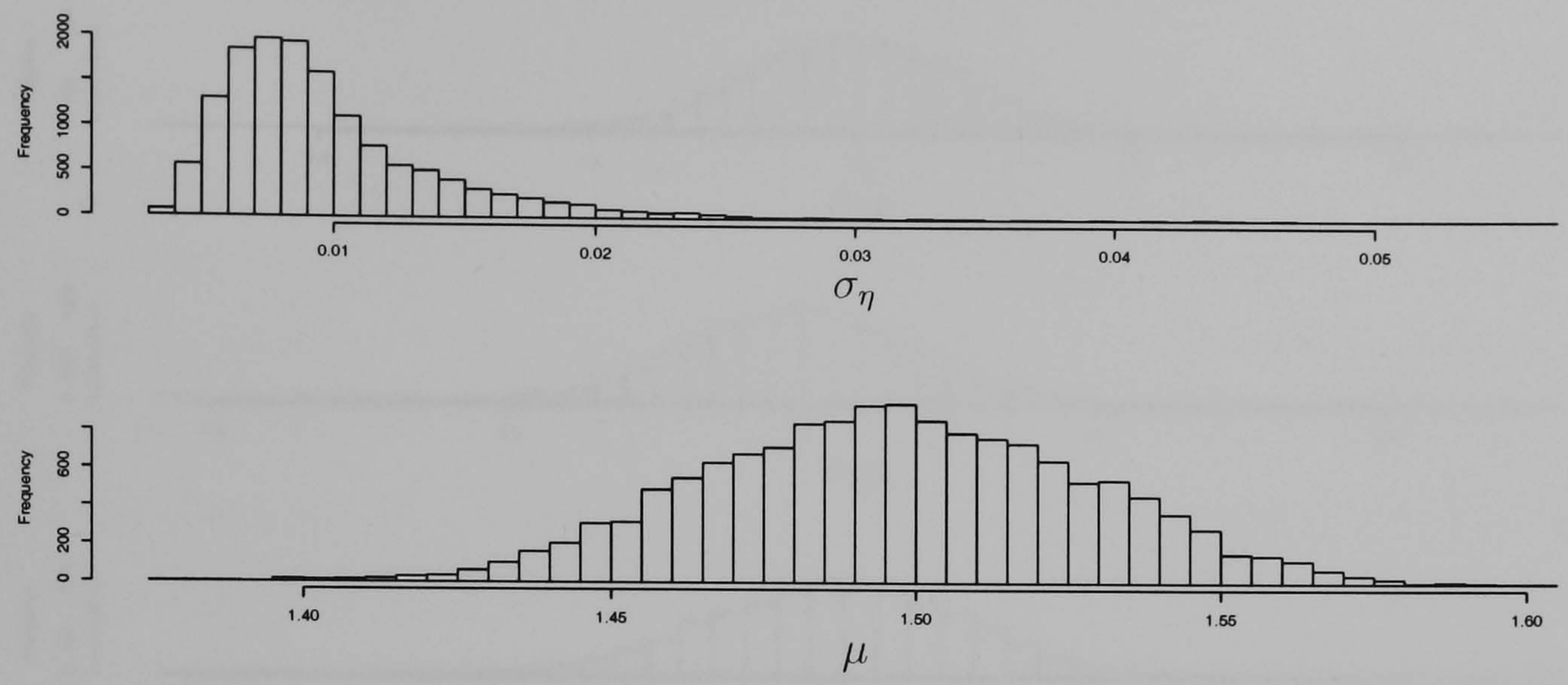


Figure 4.2: Histograms of the samples from the three parameters of the ISV model

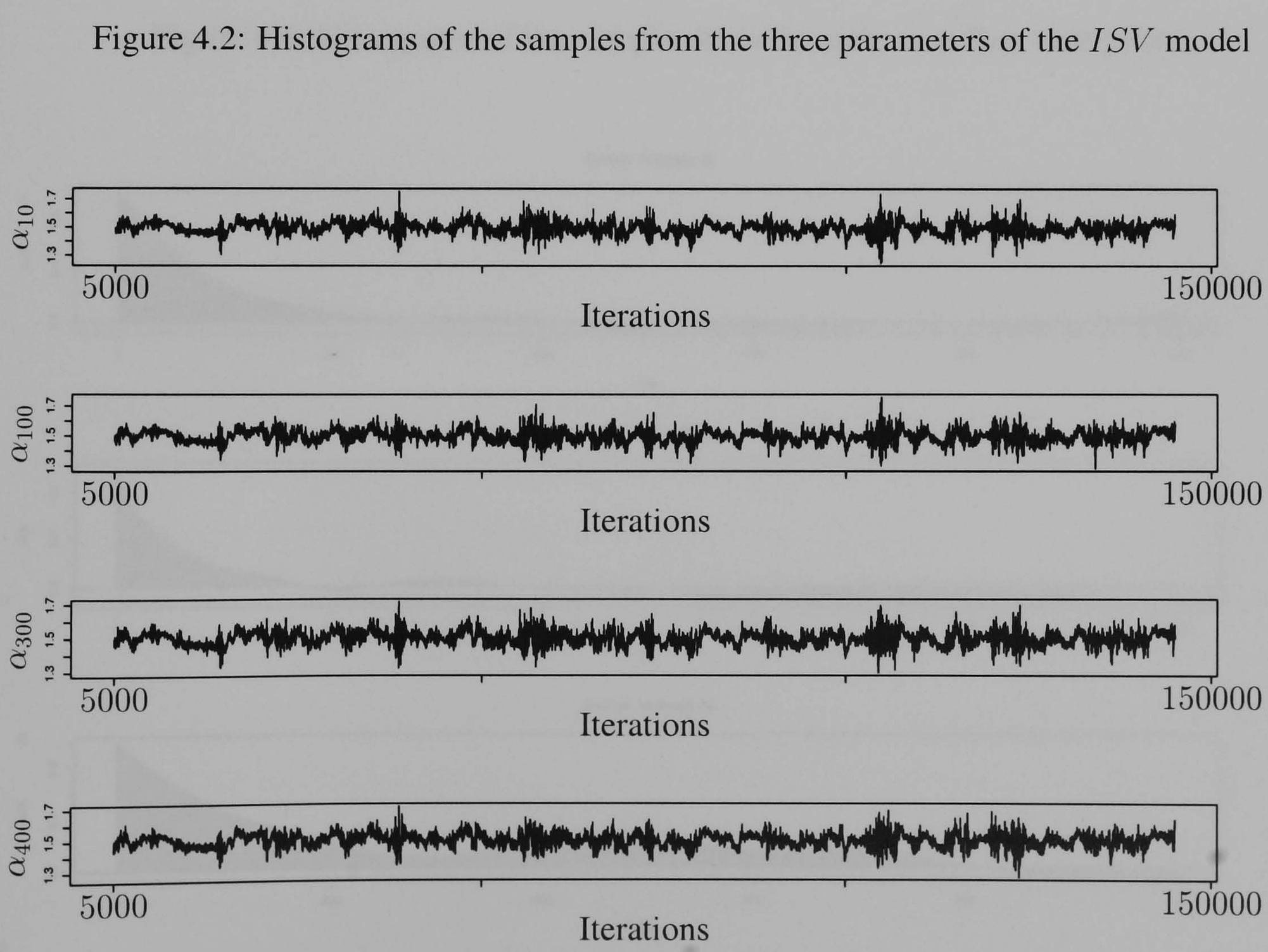


Figure 4.3: Trace plots of samples from the state α_t at four time points.

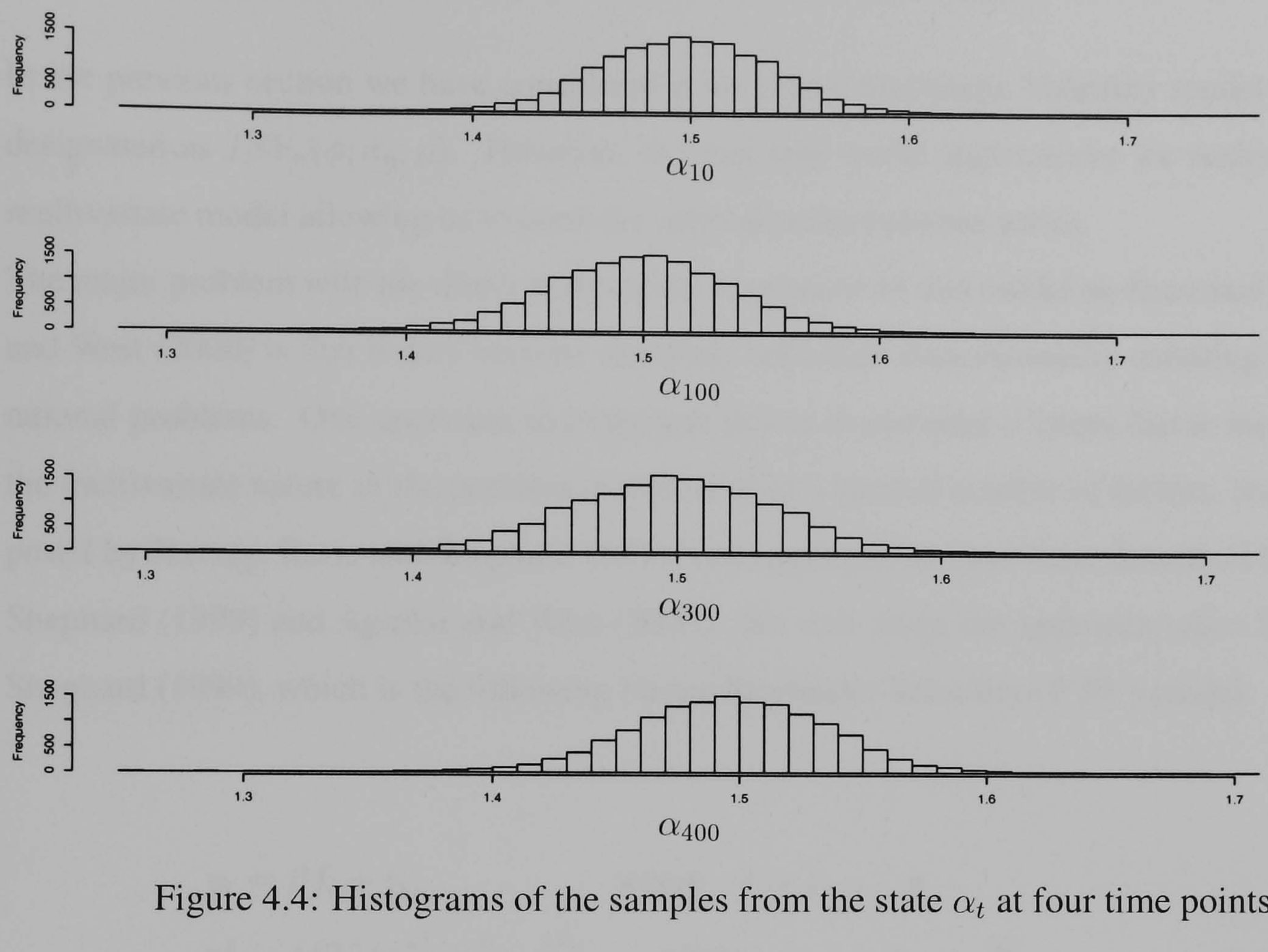


Figure 4.4: Histograms of the samples from the state α_t at four time points.

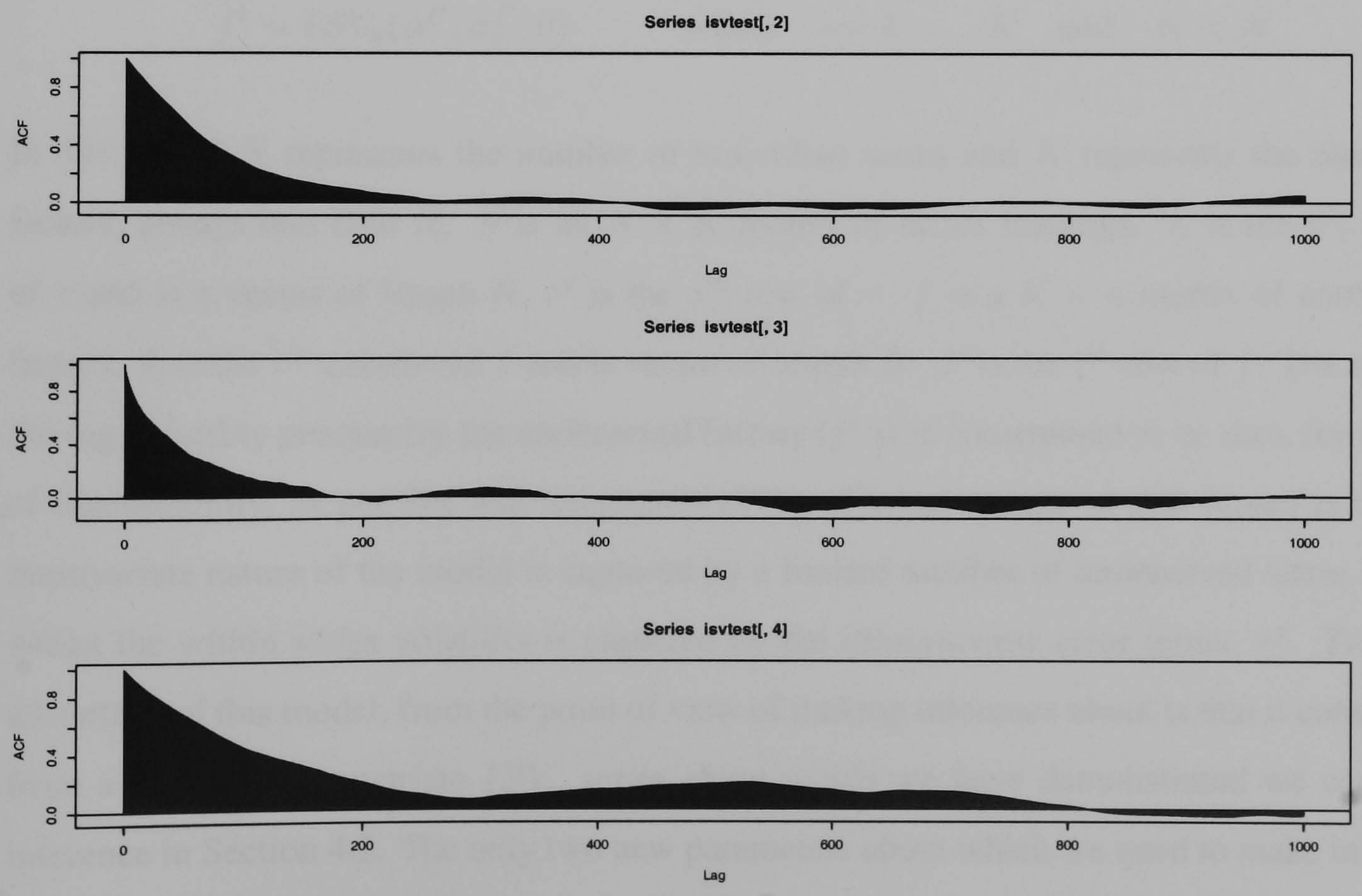


Figure 4.5: Autocorrelation plots of the three parameters of the ISV model.

4.5 Multivariate Factor Stochastic Models

In the previous section we have considered a univariate Stochastic Volatility model which we designated as $ISV_n(\phi; \sigma_\eta; \mu)$. However, in most real world applications we really require a multivariate model allowing us to consider dependencies between series.

The major problem with the direct multivariate extensions of this model as discussed in Aguilar and West (2000) is that it they become unwieldy with high dimensionality resulting in computational problems. One approach to overcome this is to consider a latent factor model where the multivariate nature of the problem is expressed in a limited number of factors, this was proposed by Harvey, Ruiz, and Shephard (1994) and applications have been described by Pitt and Shephard (1999) and Aguilar and West (2000). We will adopt the approach taken by Pitt and Shephard (1999), which is the following Factor Stochastic Volatility (FSV) model:

$$y_t = \beta f_t + \tau_t, \qquad \text{where} \quad t = 1, \ldots, n \tag{4.17}$$

$$\tau^j \sim ISV_n(\phi^{\tau^j}; \sigma_\eta^{\tau^j}; \mu^{\tau^j}) \qquad \text{where} \quad j = 1, \ldots, N \tag{4.18}$$

$$f^i \sim ISV_n(\phi^{f^i}; \sigma_\eta^{f^i}; 0) \qquad \text{where} \quad i = 1, \ldots, K \quad \text{and} \quad K < N \tag{4.19}$$

In this model N represents the number of individual series and K represents the number of factors, always less than N. β is an $N \times K$ matrix of factor loadings. τ_t is the t^{th} column of τ and is a vector of length N, τ^j is the j^{th} row of τ. f is a $K \times n$ matrix of unobserved factors. f_t is the t^{th} column of f and is vector of length K, f^i is the i^{th} row of f. The mean of the log-volatility process for the unobserved factors (f^i's) is constrained to be zero, for reasons of identifiability, as per Pitt and Shephard (1999). The advantage of this model is that the multivariate nature of the model is captured by a limited number of unobserved latent factors, whilst the within series volatility is captured by the idiosyncratic error terms, τ^j. The other advantage of this model, from the point of view of making inference about is that it constructed from a number of univariate ISV_n series about which we have demonstrated we can make inference in Section 4.2. The only two new parameters about which we need to make inference are f_t and β.

To clarify the series of interest at time t, y_t is made up of some latent factors f_t which each

evolve according to individual ISV processes and some idiosyncratic noise, τ_t which again each evolve according to individual ISV processes. There are K ISV series of factors and and N ISV series of idiosyncratic noise terms, that is one per series in the data. To clarify further consider the following simple example, we have a single factor model and so $K = 1$ and 3 series, so $N = 3$. The factor series $f \sim ISV_n(\phi^f; \sigma_\eta^f; 0)$ and there are three idiosyncratic error series

$$\tau^1 \sim ISV_n(\phi^{\tau^1}; \sigma_\eta^{\tau^1}; \mu^{\tau^1})$$
$$\tau^2 \sim ISV_n(\phi^{\tau^2}; \sigma_\eta^{\tau^2}; \mu^{\tau^2})$$
$$\tau^3 \sim ISV_n(\phi^{\tau^3}; \sigma_\eta^{\tau^3}; \mu^{\tau^3})$$

Therefore at a time point k the series of interest y is

$$\begin{pmatrix} {y_k}^1 \\ {y_k}^2 \\ {y_k}^3 \\ {y_k}^4 \end{pmatrix} = \begin{pmatrix} {\beta_k}^1 \\ {\beta_k}^2 \\ {\beta_k}^3 \\ {\beta_k}^4 \end{pmatrix} f_k + \begin{pmatrix} {\tau_k}^1 \\ {\tau_k}^2 \\ {\tau_k}^3 \\ {\tau_k}^4 \end{pmatrix}$$

Update of f_t

The f_t's are conditionally independent, so for each time point we can sample f_t from its full conditional. From (4.19) and (4.17) of our model

$$f_t|\alpha_t^f \sim N(0, F_t) \quad \text{where} \quad F_t = diag(\exp\{\alpha_t^f\})$$
$$y_t|\cdot \sim N(\beta f_t, T_t) \quad \text{where} \quad T_t = diag(\exp\{\alpha_t^\tau\})$$

The the likelihood for $f_t|y_t$ is

$$\begin{aligned} \mathrm{L}(f_t|y_t) &\propto \exp\left(-\frac{1}{2}\left(y_t - \beta f_t\right)' {T_t}^{-1}\left(y_t - \beta f_t\right)\right) \\ &\propto \exp\left(-\frac{1}{2}\left(\left(\beta f_t\right)' {T_t}^{-1}\left(\beta f_t\right) - 2\left(\left(\beta f_t\right)' {T_t}^{-1} y_t\right) + y_t' {T_t}^{-1} y_t\right)\right). \end{aligned}$$

Using Bayes Theorem (1.1) we can combine this with the prior $f_t|\alpha_t^f$ to give

$$f_t|\cdot \propto \exp\left(-\frac{1}{2}\left(\left(\beta f_t\right)' {T_t}^{-1}\left(\beta f_t\right) - 2\left(\left(\beta f_t\right)' {T_t}^{-1} y_t\right) + y_t' {T_t}^{-1} y_t + f_t' {F_t}^{-1} f_t\right)\right).$$

Re-arranging and ignoring multiplicative terms not in f_t gives

$$f_t|\cdot \sim N\left(\left(\beta' {T_t}^{-1} Y_t\right)\left({F_t}^{-1} + \beta' {T_t}^{-1}\beta\right)^{-1}, \left({F_t}^{-1} + \beta' {T_t}^{-1}\beta\right)^{-1}\right). \tag{4.20}$$

With this full conditional posterior for f_t we can sample values of this parameter for each time point t. A **sather** module FSV has been developed to make inference on multivariate factor stochastic volatility models. The FSV class draws heavily on the ISV class described previously and contains routines for updating of f_t.

Update of β

In the K factor model, the elements of β are constrained as follows $\beta_{ii} = 1, i = 1, \ldots, K$ and $\beta_{ij} = 0, \text{for} j > i$, i.e. unit diagonal lower triangular. These constraints are made to ensure identifiability of the model, for further details see Pitt and Shephard (1999) and Kim, Shephard, and Chib (1998). This means that the updating algorithm is concerned only with updating the non-fixed elements.

In the basic model (4.17) the β matrix can be considered to consist of a vector of columns.

$$\begin{aligned} y_t &= \left(\beta^{(1)}, \cdots, \beta^{(K)}\right) f_t + \tau_t \\ &= \beta^{(1)} {f_t}^1 + \cdots + \beta^{(K)} {f_t}^K + \tau_t \\ &= {f_t}^1 \beta^{(1)} + \cdots + {f_t}^K \beta^{(K)} + \tau_t \\ &= \sum_{i=1}^{K} {f_t}^i \beta^{(i)} + \tau_t \end{aligned}$$

writing $\beta^{(i)}$ as the i^{th} column of matrix β and all other columns as $\beta^{(i\backslash)}$ and denoting ${f_t}^i$ as the i^{th} element of the vector f at time t. Placing an independent normal prior on each column, $\beta^{(i)} \sim N(a, b)$ then we can show that $\beta^{(i)}|\beta^{(i\backslash)}, y_t, \alpha^\tau, f$ is also normal.

Consider the column $\beta^{(m)}$ to be the m^{th} column of β from $K = 1$ to N.

$$f_t{}^m\beta^{(m)} = y_t - \sum_{i\neq m} f_t{}^i\beta^{(i)} - \tau_t \sim N\left(y_t - \sum_{i\neq m} f_t{}^i\beta^{(i)}, T_t^{-1}\right)$$

$$\mathrm{L}_t(\beta^{(m)}) \sim N\left(\frac{1}{f_t{}^m}\left(y_t - \sum_{i\neq m} f_t{}^i\beta^{(i)}\right), \frac{1}{(f_t{}^m)^2}T_t^{-1}\right)$$

$$\mathrm{L}(\beta^{(m)}) \propto \prod_{t=1}^{n} \exp\left(-\frac{1}{2}\left\{\left(\beta^{(m)} - \frac{1}{f_t{}^m}\left(y_t - \sum_{i\neq m} f_t{}^i\beta^{(i)}\right)\right)'\right.\right.$$
$$\left.\left.(f_t{}^m)^2 T_t{}^{-1}\left(\beta^{(m)} - \frac{1}{f_t{}^m}\left(y_t - \sum_{i\neq m} f_t{}^i\beta^{(i)}\right)\right)\right\}\right)$$

Letting

$$A = \sum_{t=1}^{n} \frac{1}{f_t{}^m}\left(y_t - \sum_{i\neq m} f_t{}^i\beta^{(i)}\right)$$

$$B = \sum_{t=1}^{n} (f_t{}^m)^2 T_t{}^{-1}$$

this can be written as

$$\mathrm{L}(\beta^{(m)}) \propto \exp\left[-\frac{1}{2}\left\{\left(\beta^{(m)} - A\right)' B \left(\beta^{(m)} - A\right)\right\}\right]. \tag{4.21}$$

Combine this with the prior $\beta^{(m)} \sim N(a, b)$ using Bayes theorem (1.1), rearranging and dropping terms not in $\beta^{(m)}$ we get

$$\beta^{(m)}|\cdot \propto \exp\left[-\frac{1}{2}\left\{\beta^{(m)'}\left(B + b^{-1}\right)\beta^{(m)} - 2\beta^{(m)'}\left(A + a'b^{-1}\right)\right\}\right].$$

Hence the full conditional for $\beta^{(m)}$ is

$$\beta^{(m)}|\cdot \sim N\left(\left(A + a'b^{-1}\right)\left(B + b^{-1}\right)^{-1}, \left(B + b^{-1}\right)^{-1}\right),$$

therefore to update the whole matrix we simply iterate through $\beta^{(i)}$ for $i = 1, \ldots, K$ sampling each $\beta^{(i)}$ from it's full conditional.

We have already stated, however that the first K elements of β are fixed to ensure identifiability

of the model and therefore for reasons of computational efficiency we are only interested in updating the non-fixed elements.

Subdividing our $\beta^{(m)}$ vector into two sections, the first $\beta^{(m\uparrow)}$ which contains the first K restricted elements and $\beta^{(m\downarrow)}$ which contains the unrestricted elements, and the corresponding sub-matrices of B are likewise notated. We can perform this sub setting because of the block diagonal nature of the B matrix. The likelihood becomes

$$\mathrm{L}(\beta^{(m)}) \sim N(A, B)$$

$$\mathrm{L}(\beta^{(m)}) \sim N\left(\begin{pmatrix} \beta^{(m\uparrow)} \\ \beta^{(m\downarrow)} \end{pmatrix}; \begin{pmatrix} A^{(m\uparrow)} \\ A^{(m\downarrow)} \end{pmatrix}, \begin{pmatrix} B^{(m\uparrow)} & 0 \\ 0 & B^{(m\downarrow)} \end{pmatrix}\right)$$

$$\mathrm{L}(\beta^{(m)}) \propto \exp\left(\left(\begin{pmatrix} \beta^{(m\uparrow)} \\ \beta^{(m\downarrow)} \end{pmatrix} - \begin{pmatrix} A^{(m\uparrow)} \\ A^{(m\downarrow)} \end{pmatrix}\right) \begin{pmatrix} B^{(m\uparrow)} & 0 \\ 0 & B^{(m\downarrow)} \end{pmatrix}^{-1} \left(\begin{pmatrix} \beta^{(m\uparrow)} \\ \beta^{(m\downarrow)} \end{pmatrix} - \begin{pmatrix} A^{(m\uparrow)} \\ A^{(m\downarrow)} \end{pmatrix}\right)$$

Since we are only interested in the non-fixed terms, i.e. the $\beta^{(m\downarrow)}$ the likelihood becomes

$$\mathrm{L}(\beta^{(m\downarrow)}) \propto \exp\left(\left(\beta^{(m\downarrow)} - A^{(m\downarrow)}\right)'\left(B^{(m\downarrow)}\right)^{-1}\left(\beta^{(m\downarrow)} - A^{(m\downarrow)}\right)\right)$$

Which can be written as

$$\mathrm{L}(\beta^{(m)}) \propto N(A^{(m\downarrow)}, B^{(m\downarrow)})$$

Where $A^{(m\downarrow)}$ and $B^{(m\downarrow)}$ are the relevant subsections of A and B from (4.5). This can be combined with an appropriate prior to obtain the full conditional for $\beta^{(m\downarrow)}$. This means we only need to consider the non-fixed elements of the β matrix when conducting the update.

The **sather** class FSV includes the necessary routines to sample $\beta^{(m\downarrow)}$ from it's full conditional.

The Algorithm for Inference on a Factor Stochastic Volatility Model

1. Sample from the states α^{f^i} for $i = 1, \ldots, K$ using the ISV class as described in Section 4.2.

2. Sample from the states θ^{f^i} for $i = 1, \ldots, K$ where $\theta^{f^i} = (\phi^{f^i}; {\sigma_\eta}^{f^i})$ using the ISV class as described in Section 4.2.

3. Sample from the states α^{τ^j} for $j = 1, \ldots, N$ using the ISV class as described in Section 4.2.

4. Sample from the states θ^{τ^j} for $j = 1, \ldots, N$ where $\theta^{\tau^j} = (\phi^{\tau^j}; {\sigma_\eta}^{\tau^j}; \mu^{\tau^j})$ using the ISV class as described in Section 4.2.

5. Sample from the full conditional posterior for f, (4.20).

6. Sample from the full conditional posterior for β as described in Section 4.5.

7. Repeat steps 1 to 6 until convergence.

4.5.1 Example

The above algorithm can be programmed in **sather** and used to make inference about factor stochastic volatility models. We simulated 2000 data points from the following FSV model.

$$y_t = \beta f_t + \tau_t$$

where β is the vector $(1, 0.5, 0.33, 0.25)$ and

$$f_t \sim ISV_n(0.8, 0.01, 0)$$

$${\tau_t}^1 \sim ISV_n(0.7, 0.02, 0)$$

$${\tau_t}^2 \sim ISV_n(0.7, 0.02, 1)$$

$${\tau_t}^3 \sim ISV_n(0.7, 0.02, 2)$$

$${\tau_t}^4 \sim ISV_n(0.7, 0.02, 3)$$

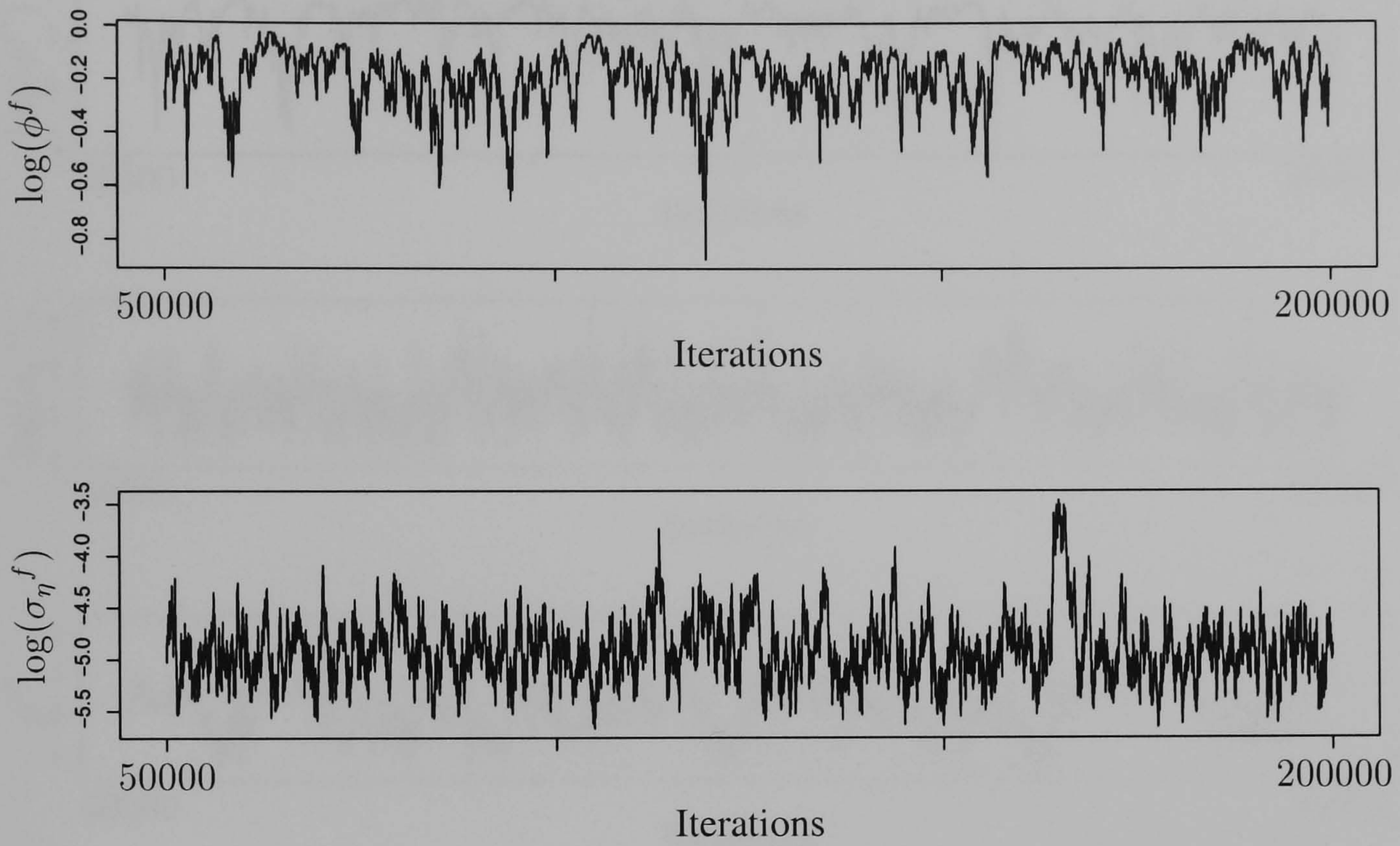
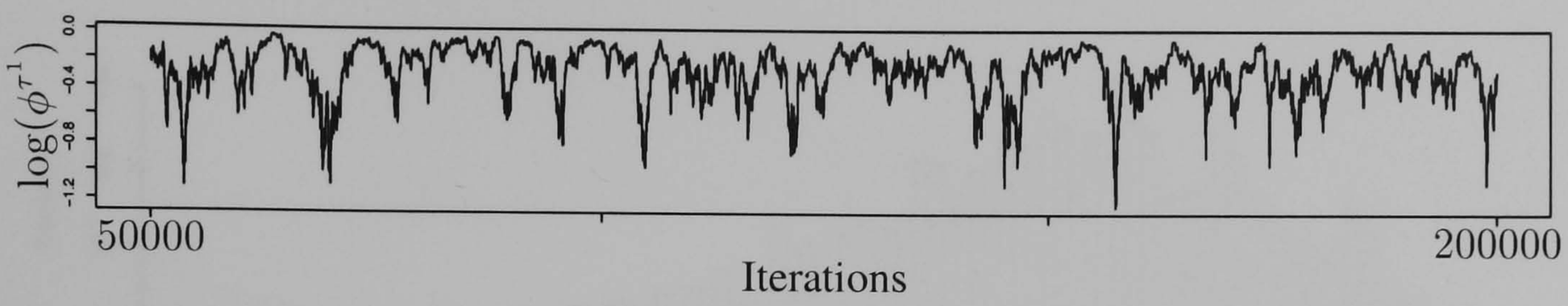


Figure 4.6: Trace plots of samples from the two parameters of the f component of the FSV model

The **sather** program *examplefsv.sa* which is included in Appendix A can the be run to make inference on this model. Independent flat $\Gamma(0, 0.0001)$ priors are set on the σ_η parameters, while a $N(0, 100)$ prior is placed on each of the μ^τ's and an identical prior is placed on the β's. A slightly stronger $\beta(20, 3)$ prior is placed on the ϕ's to reflect our prior belief that these parameters will be closer to 1 than 0. The program was run for 200,000 iterations with the first 50,000 discarded as a burn-in. The remaining data was thinned by a factor of 20.

Figure 4.6 shows the trace plots for the log of the two parameters of the $f\ ISV$ model. Looking at these and the autocorrelation plots, Figure 4.12 suggest that the sampler has not fully converged with long lags in the autocorrelations.

Figure 4.7 shows the traces of the parameters of the $\tau^1\ ISV$ model. Again looking at the autocorrelation plots Figure 4.13 suggests that the sampler might not have converged fully. The results for the other τ models are similar. However for both f and the τ's looking at the histograms of the samples Figure 4.9 and Figure 4.10 shows the samples having modal values close to the true values suggesting we are in fact sampling from the distribution of interest.



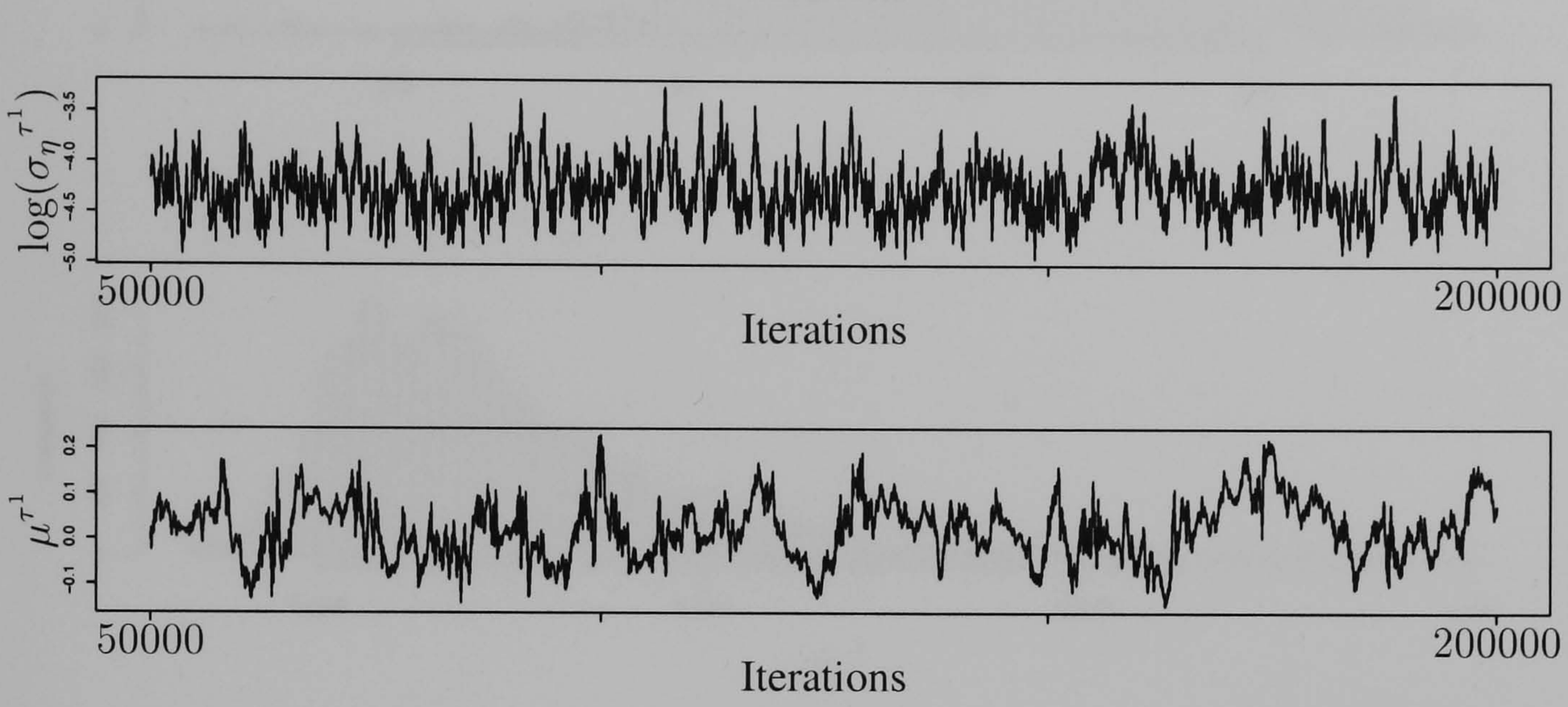
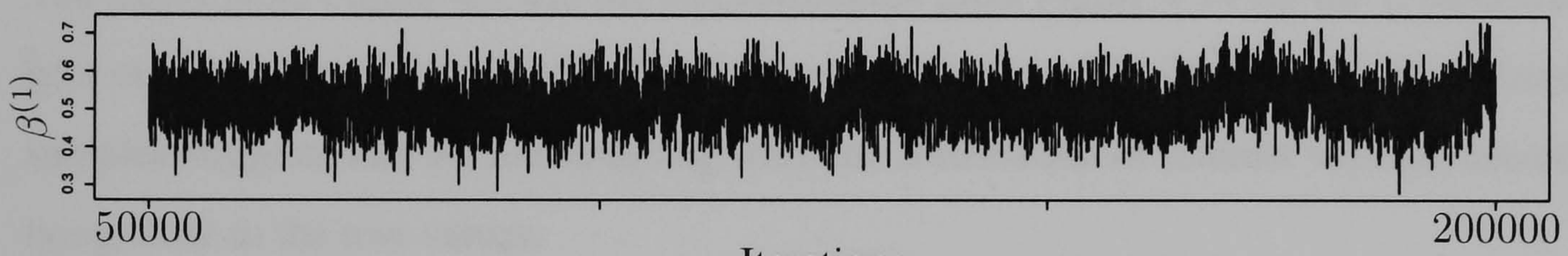
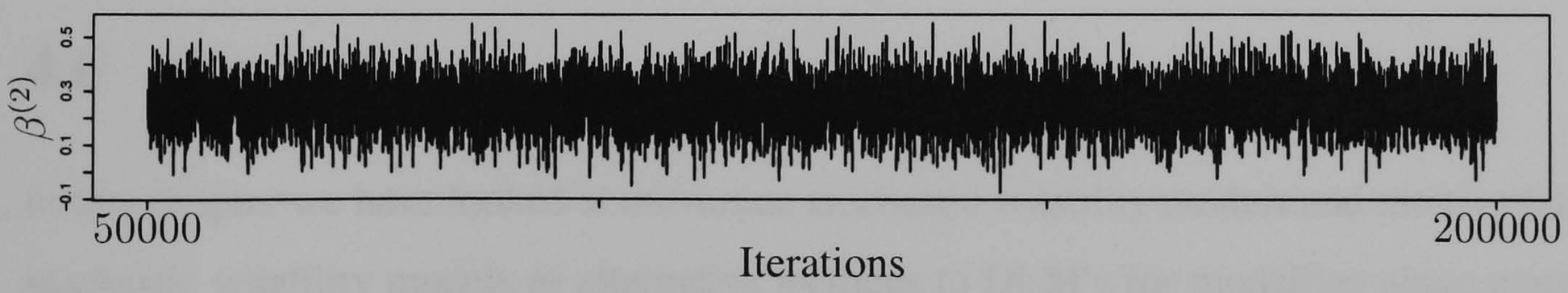


Figure 4.7: Trace plots of samples from the three parameters of τ^1 component of the FSV model





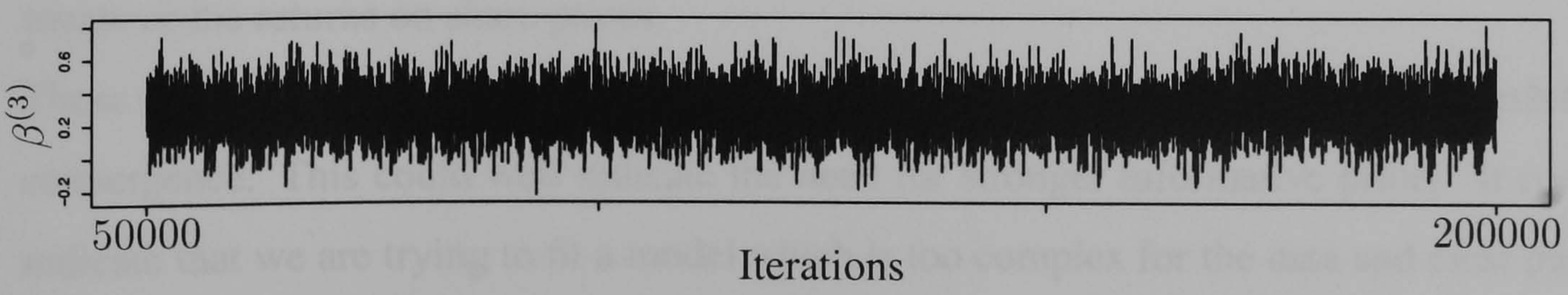


Figure 4.8: Trace plots of samples from the three components of the β parameter of the FSV model

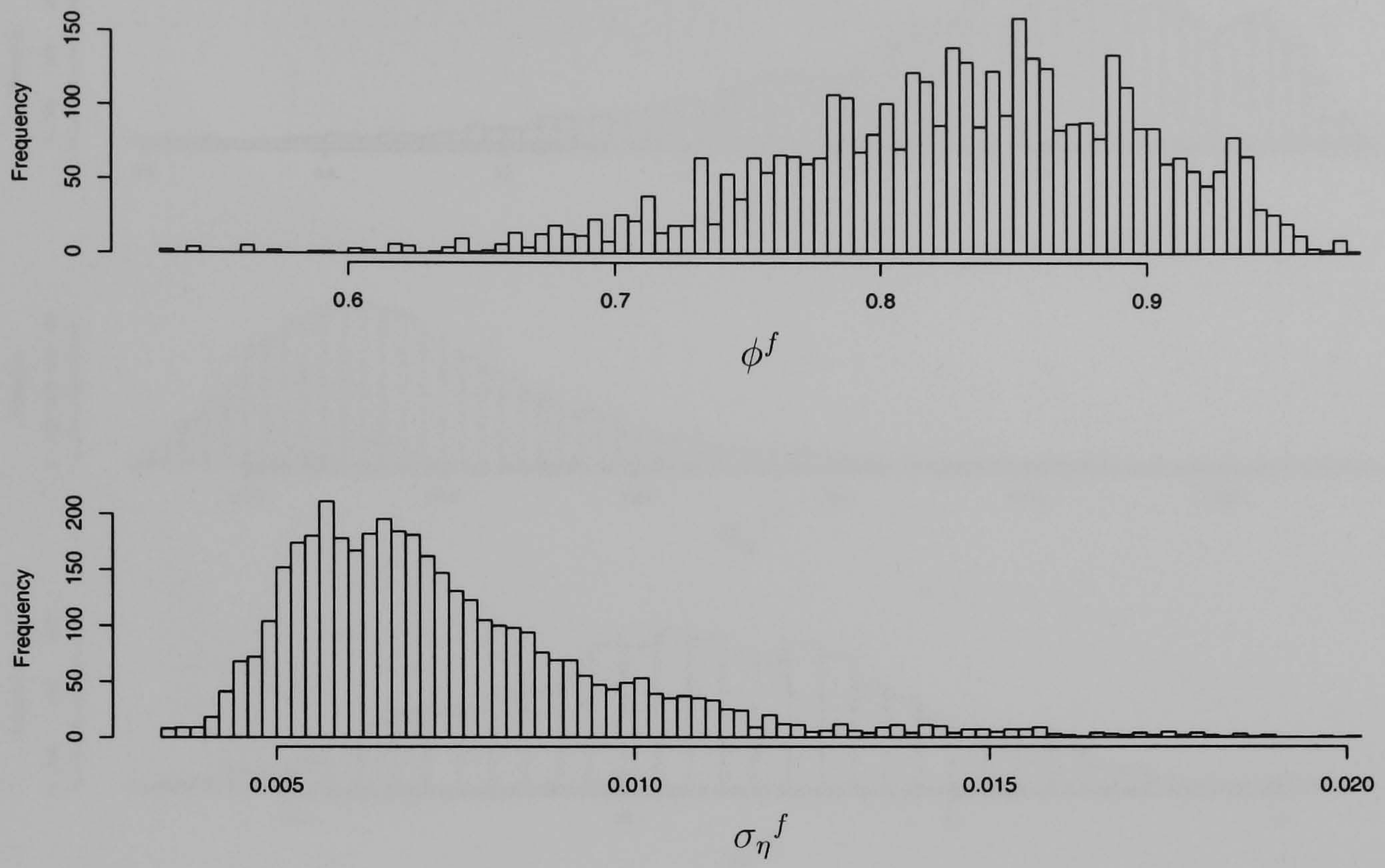


Figure 4.9: Histograms of the samples from the two parameters of the f component of the FSV model

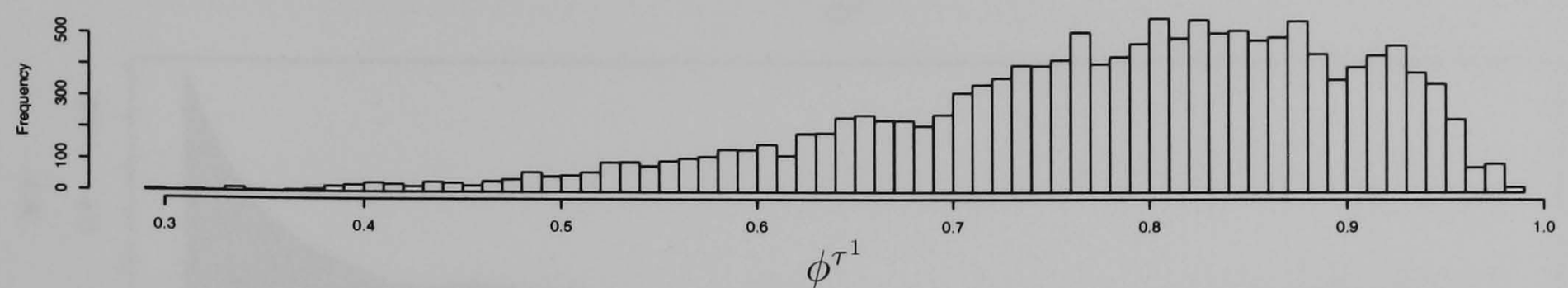
The traces plots Figure 4.8 and the autocorrelation plots Figure 4.14 for the β parameters suggest convergence of the sampler for these components and looking at the histograms of the samples suggests that we are sampling from the distribution of interest with the modal values lying close to the true values.

4.6 Summary

In this chapter we have looked at univariate stochastic volatility models and multivariate factor stochastic volatility models as alternative methods to DLM's for modelling share price movements or the returns on share prices.

These models have been applied with some success, however in many cases they exhibit poor convergence. This could well indicate the need for stronger informative priors. It could also indicate that we are trying to fit a model which is too complex for the data and clear parameter inference is therefore impossible.

In the next chapter we will look at a model which tries to combine both the DLM and factor



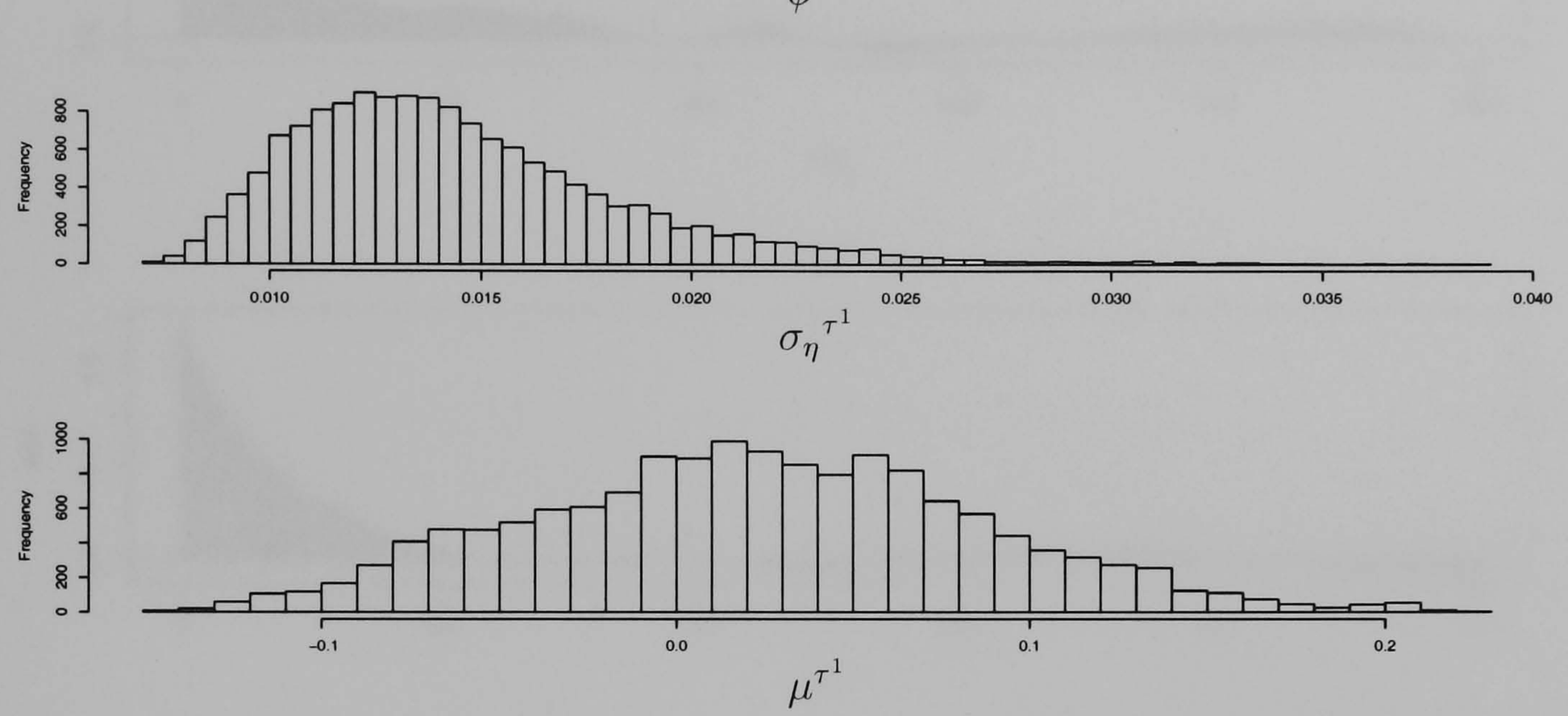


Figure 4.10: Histograms of the samples from the three parameters of the τ^1 component of the FSV model

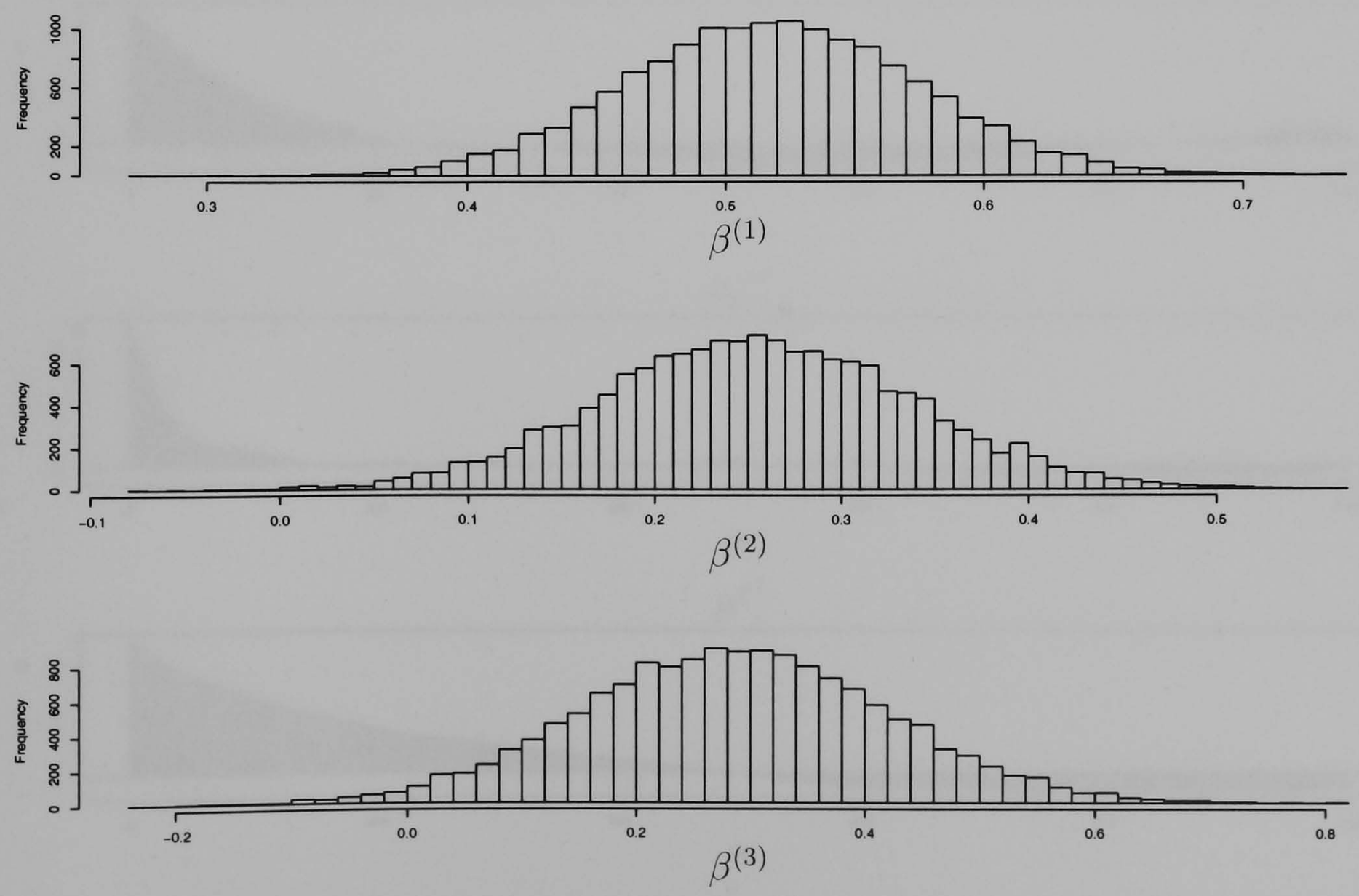


Figure 4.11: Histograms of the samples from the three parameters of the β component of the FSV model

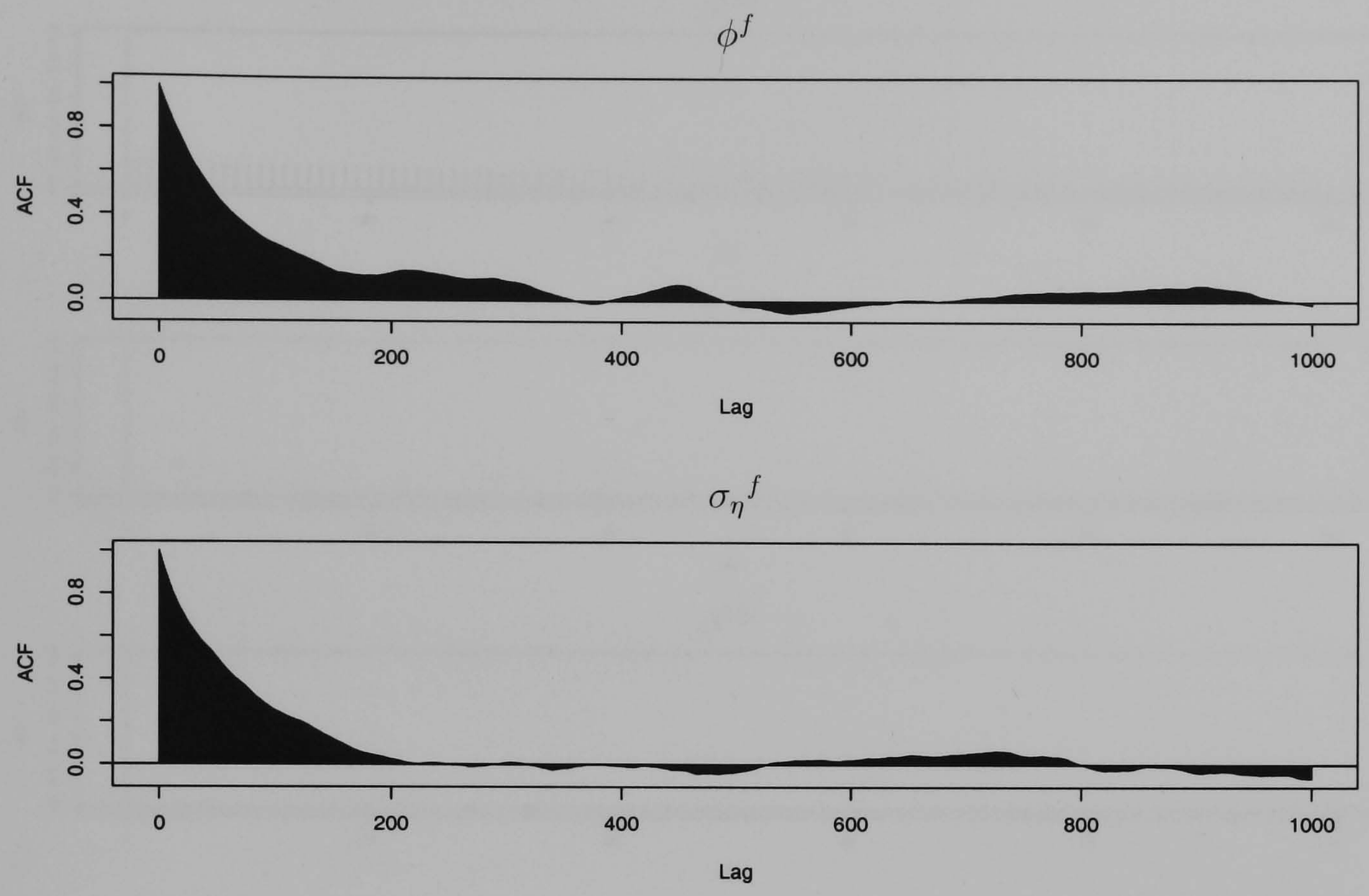


Figure 4.12: Autocorrelation plots of the two parameters of the f component of the FSV model.

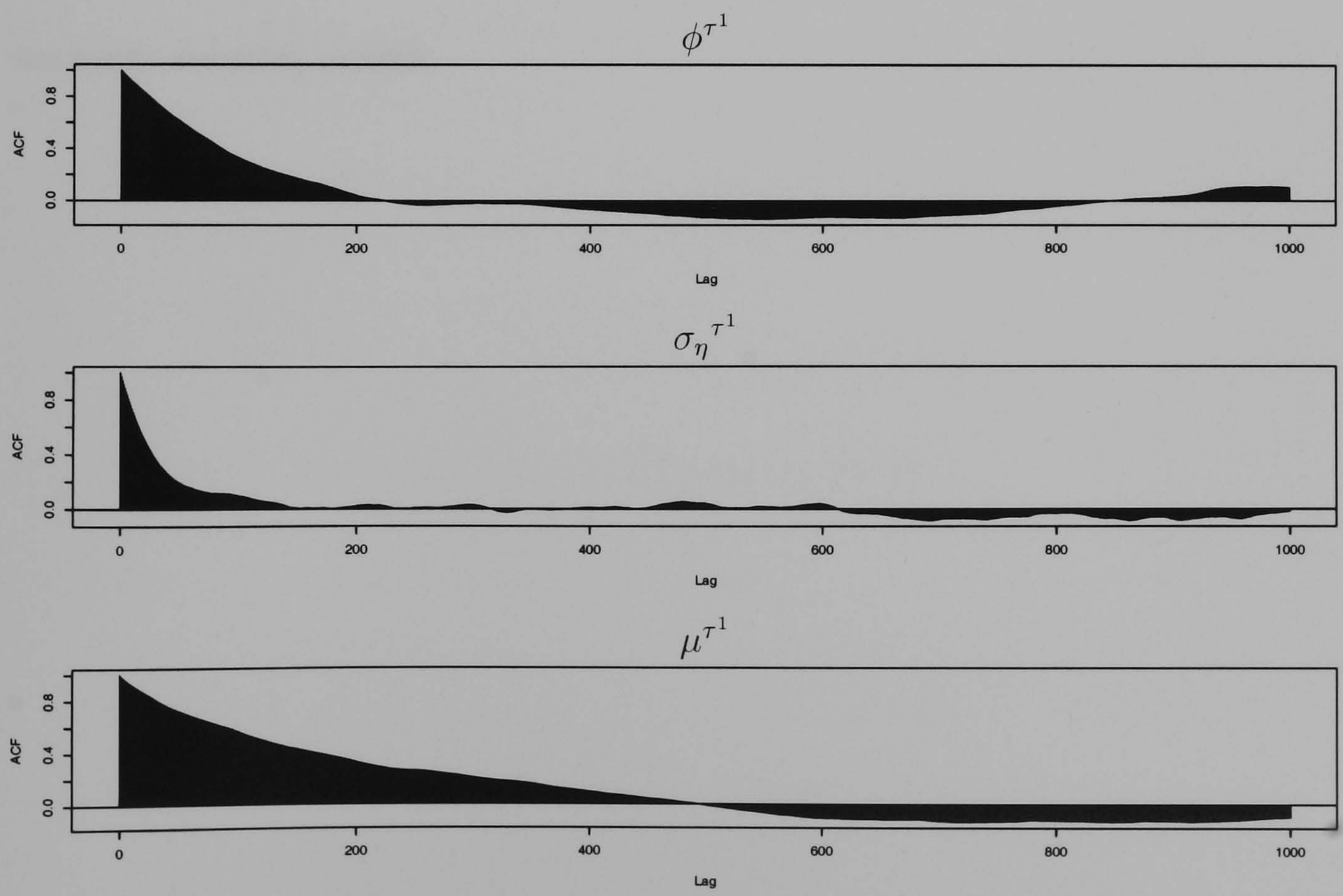


Figure 4.13: Autocorrelation plots of the three parameters of the τ^1 component of the FSV model.

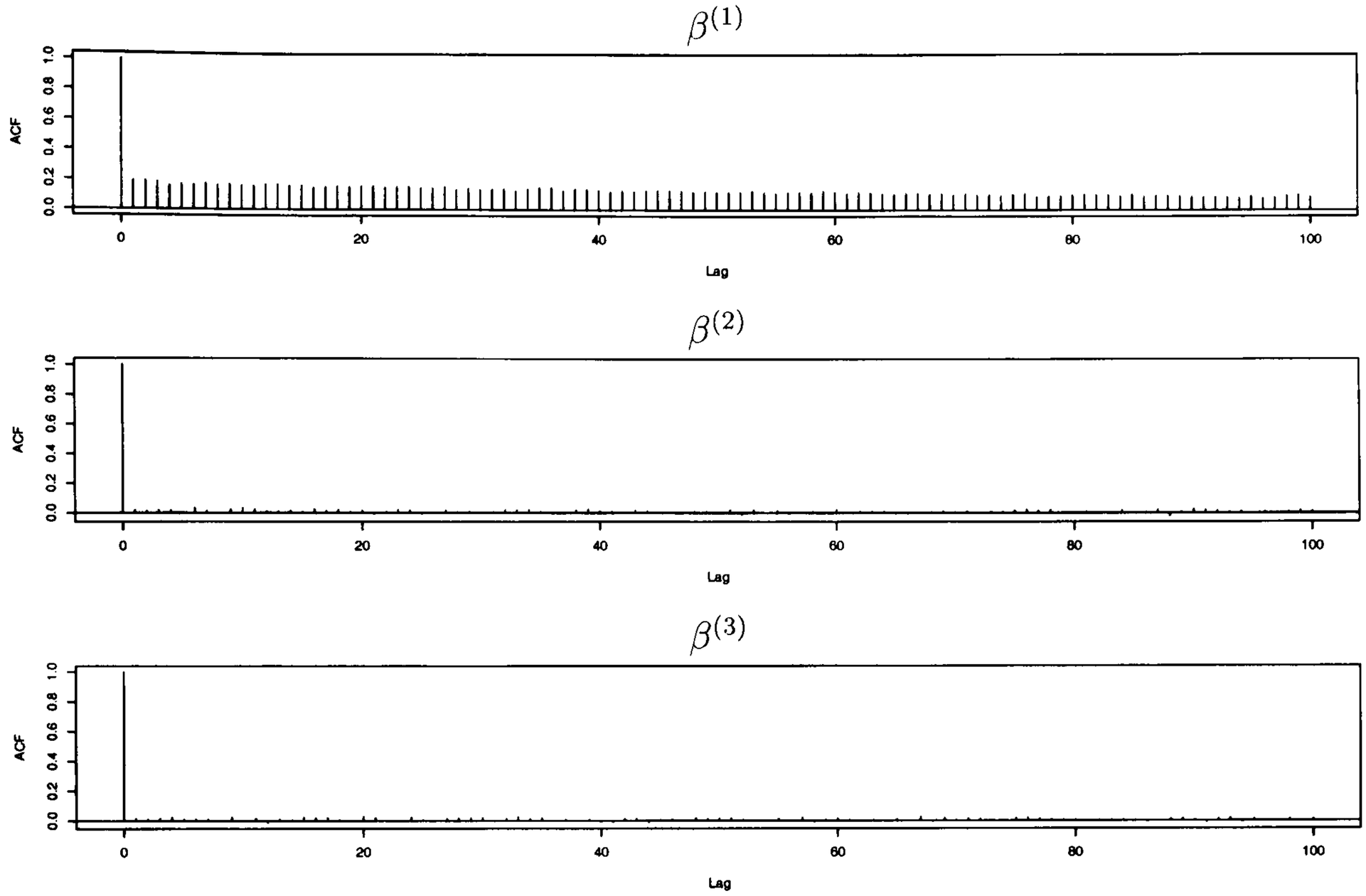


Figure 4.14: Autocorrelation plots of the three parameters of the β component of the FSV model.

stochastic volatility models.

Chapter 5

The Combined Dynamic Linear and Stochastic Volatility Models

In Chapter 2 we looked at methods whereby we could use Kalman filtering and smoothing techniques to make inference about the state, θ of a dynamic linear model, given the variance matrices V and W. In Chapter 3 we demonstrated methods of making inference about these two matrices assuming their values were fixed through time. In Chapter 4 we showed that stochastic volatility models could be used to model time evolving volatility and that we could make inference on these models. In this chapter we will show how the stochastic volatility models can be used as a more flexible model for ν in out dynamic linear model structure and thereby allowing V to evolve through time.

5.1 The Univariate Combined Model

Let us first consider a univariate combined model

$$Y_t = F\theta_t + \nu_t \qquad \nu_t \sim N(0, V_t) \tag{5.1}$$

$$\theta_t = G\theta_{t-1} + \omega_t \qquad \omega_t \sim N(0, W). \tag{5.2}$$

This at first sight appears the same as our standard dynamic linear model about which we can make parameter inference, however in this case we have allowed the V_t to evolve through time.

Since $\nu_t \sim N(0, V_t)$ this can be written as

$$\nu_t = \sqrt{V_t}\epsilon_t \qquad \epsilon_t \sim N(0,1)$$

which we can recognise as equation (4.5). Letting $\alpha_t = \log V_t$ and as in Section 4.2 looking at $\log {\nu_t}^2$,

$$\log {\nu_t}^2 = \alpha_t + \log {\epsilon_t}^2. \tag{5.3}$$

Again α_t evolves through time according to a centred model

$$\alpha_t = \mu + \phi(\alpha_{t-1} - \mu) + \eta_t \qquad \eta_t \sim N(0, {\sigma_\eta}^2) \tag{5.4}$$

Hence we can say that $\nu_t \sim ISV_n(\phi; \sigma_\eta; \mu)$ with a state, $\alpha_t = \log V_t$. This allows us to now consider parameter estimation for this combined model.

5.1.1 Parameter estimation

We now have a model defined by equations (5.1),(5.2), (5.3) and (5.4). We wish to make inference about $V, W, \theta, \phi, \sigma_\eta, \mu | Y$ which we can not achieve in a single sampling step. However we can once again construct a block Gibbs sampler to sample each of these parameters alternately to achieve the sample required. Fortunately we can do this building up the sampler from the techniques discussed in Chapter 2, Chapter 3 and Chapter 4. Perhaps the clearest way of seeing this is to consider the following algorithm.

Algorithm

1. Run the Kalman filter and simulation smoother, with some initial values for $V_1, \ldots, V_n$ and W, as discussed in Chapter 2 to sample the state, θ given all other model parameters, i.e. sample $\theta | V, W, \phi, \sigma_\eta, \mu, Y$.

2. Sample from the state of $\log {\nu_t}^2$, α_t using the Kalman filter simulation smoother provided in the ISV class in **sather** , i.e. sample $\alpha | V, W, \theta, \phi, \sigma_\eta, \mu, Y$.

3. Calculate the likelihoods, (**??**) and (4.16) and evaluate the acceptance probability A and accept the new values for the state, α with this probability.

4. Sample from the full conditional for σ_η, (4.10), given the states and all other model parameters, i.e. sample $\sigma_\eta|V, W, \theta, \phi, \mu, Y$.

5. Sample from the full conditional for μ, (4.12), given the states and all other model parameters, i.e. sample $\mu|V, W, \theta, \phi, \sigma_\eta, Y$.

6. Sample from the full conditional for ϕ, (4.14) using a random walk Metropolis-Hastings scheme, given the states and all other model parameters, i.e. sample $\phi|V, W, \theta, \sigma_\eta, \mu, Y$.

7. Take the states α_t and exponentiate to give V_t.

8. Sample from the full conditional for W^{-1}, (3.5) using an appropriate Gamma prior, i.e sample $W|\theta, V, \phi, \sigma_\eta, \mu, Y$.

9. Re-run the Kalman filter simulation smoother using the new values for V and W to sample $\theta|V, W, \phi, \sigma_\eta, \mu, Y$.

10. Repeat steps 2 to 9 until convergence.

The above algorithm uses updates that we have seen already and so to enact in **sather** this requires simply using the KALMAN and ISV classes to provide the necessary samples.

5.2 The Multivariate Combined Model

The multivariate extension of this combined model is simply a natural extension using the multivariate extension of the dynamic linear model and the multivariate factor stochastic model as discussed in Section 4.5. Consider the following model

$$Y_t = F'\theta_t + \nu_t \qquad \text{where} \qquad \nu_t \sim FSV(\phi; \sigma_\eta; \mu) \tag{5.5}$$

$$\theta_t = G\theta_{t-1} + \omega_t \qquad \text{where} \qquad \omega_t \sim N(0, W) \tag{5.6}$$

So that $\nu_t = \beta f_t + \tau_t$, where

$$\tau^j \sim ISV_n(\phi^{\tau^j}; {\sigma_\eta}^{\tau^j}; \mu^{\tau^j}) \quad \text{and} \quad f^i \sim ISV_n(\phi^{f^i}; {\sigma_\eta}^{f^i}; 0)$$

This appears to be a complicated model, however it can be rewriten as

$$Y_t = F'\theta_t + \beta f_t + \tau_t \quad \text{where} \quad \tau^j \sim ISV_n(\phi^{\tau^j}; {\sigma_\eta}^{\tau^j}; \mu^{\tau^j}) \tag{5.7}$$

$$y_t - \beta f_t = F'\theta_t + \tau_t \quad \text{where} \quad \tau_t \sim N(0, V_t) \tag{5.8}$$

recalling that $V_t = diag(\exp({\alpha_t}^{\tau^j}))$. The model is now in a dynamic linear model format and hence we are able to make consider making inference about the model parameters.

5.2.1 Parameter estimation

We now have a model defined by equations (5.6) and (5.8). We wish to make inference about $V, W, \theta, \phi, \sigma_\eta, \mu | Y$ which we can not achieve in a single sampling step, however we can once again construct a block Gibbs sampler to sample each of these parameters alternately to achieve the sample required. Fortunately we can do this building up the sampler from the techniques discussed in Chapter 2, Chapter 3 and Chapter 4. Perhaps the clearest way of seeing this is to consider the following algorithm.

Algorithm

1. Run the Kalman filter and simulation smoother, with some initial values for V and W, as discussed in Chapter 2 to sample the state, θ given all other model parameters, i.e. sample $\theta | V, W, \phi, \sigma_\eta, \mu, Y$.

2. Place the ν's into an FSV object.

3. Sample from the states α^{f^i} for $i = 1, \ldots, K$ using the ISV class as described in Section 4.2.

4. Sample from the states θ^{f^i} for $i = 1, \ldots, K$ where $\theta^{f_i} = (\phi^{f_i}; {\sigma_\eta}^{f_i})$ using the ISV class as described in Section 4.2.

5. Sample from the states α^{τ^j} for $j = 1, \ldots, N$ using the ISV class as described in Section 4.2.

6. Sample from the states θ^{τ^j} for $j = 1, \ldots, N$ where $\theta^{\tau_j} = (\phi^{\tau_i}; {\sigma_\eta}^{\tau_i}; \mu^{\tau_j})$ using the ISV class as described in Section 4.2.

7. Sample from the full conditional posterior for f, (4.20).

8. Sample from the full conditional posterior for β as described in Section 4.5.

9. Set $V_t = diag(\exp\{{\alpha_t}^{\tau^j}\})$.

10. Sample from the full conditional for W^{-1}, (3.8) using an appropriate inverse Wishart prior, i.e sample $W|\theta, V, \phi, \sigma_\eta, \mu, Y$.

11. Re-set the data for the dynamic linear model as $y_t - \beta f_t$ to conform with the new V_t matrices.

12. Re-run the Kalman filter simulation smoother using the new values for V and W to sample $\theta|V, W, \phi, \sigma_\eta, \mu, Y$.

13. Repeat steps 2 to 12 until convergence.

5.3 Example

The **sather** program *meanvar-Vfsv.sa* which can be found in Appendix A has been written to generate data from a combined dynamic linear and stochastic volatility model.
The model the program generates data from is as follows

$$y_t = \theta_t + \nu_t \qquad \nu_t \sim FSV_n$$
$$\theta_t = \theta_{t-1} + \omega_t \qquad \omega_t \sim N(0, 0.001)$$

Where

$$\nu_t = \beta f_t + \tau_t$$

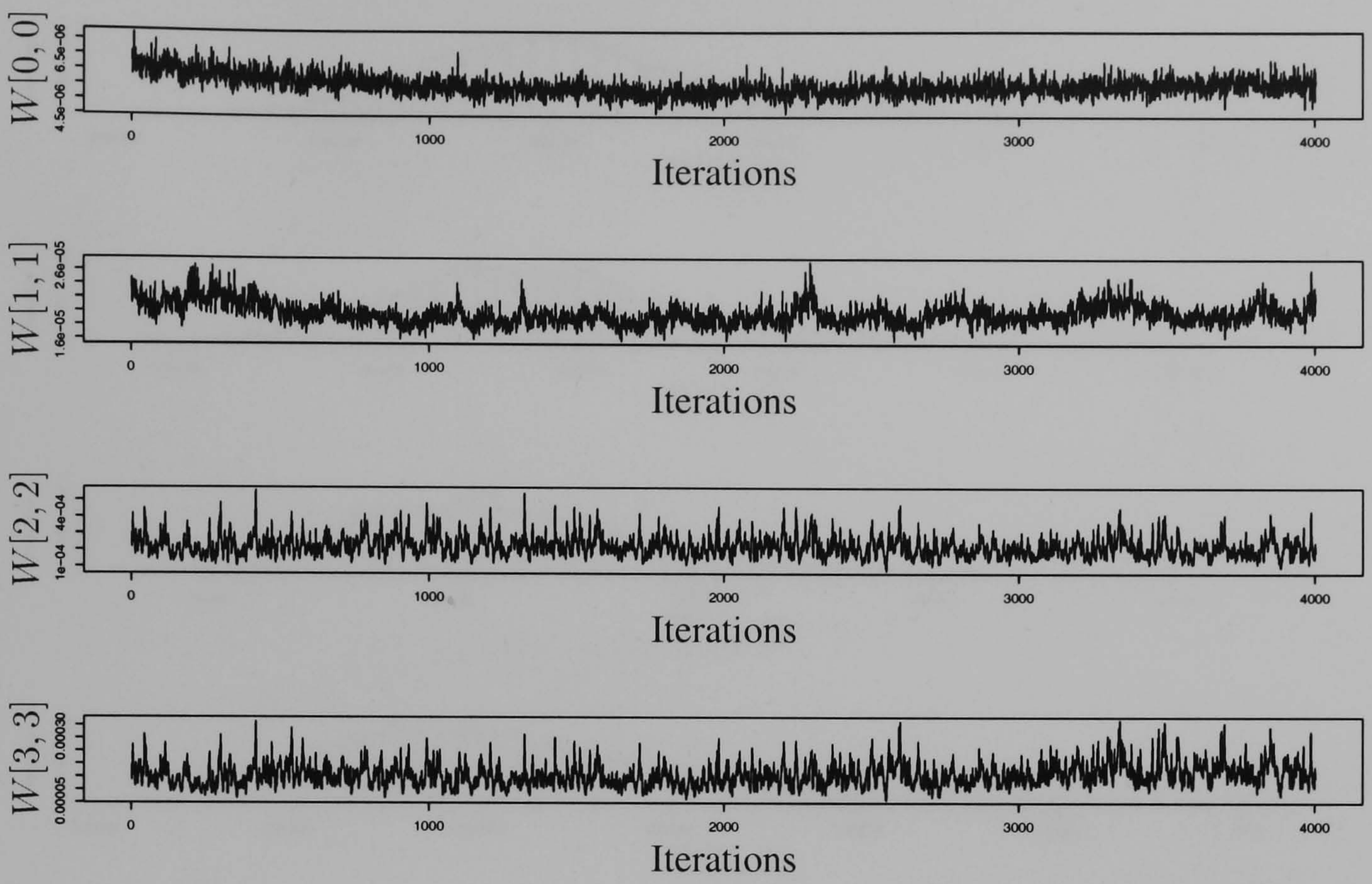


Figure 5.1: Trace plots for the diagonal elements of the state variance matrix WW

Where β is the vector $(1, 0.5, 0.33, 0.25)$ and

$$f_t \sim ISV_n(0.8, 0.01, 0)$$

$$\tau_t{}^1 \sim ISV_n(0.7, 0.02, 0)$$

$$\tau_t{}^2 \sim ISV_n(0.7, 0.02, 1)$$

$$\tau_t{}^3 \sim ISV_n(0.7, 0.02, 2)$$

$$\tau_t{}^4 \sim ISV_n(0.7, 0.02, 3)$$

A time series of 1000 data points was generated from this model and then we enact the above algorithm to sample from the posteriors of the parameters. The sampler was run for 45,000 iterations and the first 5,000 was discarded as a burn-in, the remaining data was thinned by a factor of 10. Flat $\Gamma(0, 0.0001)$ priors were placed on the σ_η's while $N(0, 1)$ priors were placed on the μ's. A slightly tighter $\beta(20, 3)$ prior was placed on the ϕ's. A Wishart prior with 6 degrees of freedom and a scale matrix which was diagonal 0.00001 was placed on the W matrix.

Figure 5.1 shows the trace plots for the diagonal elements of the state variance W, these plots

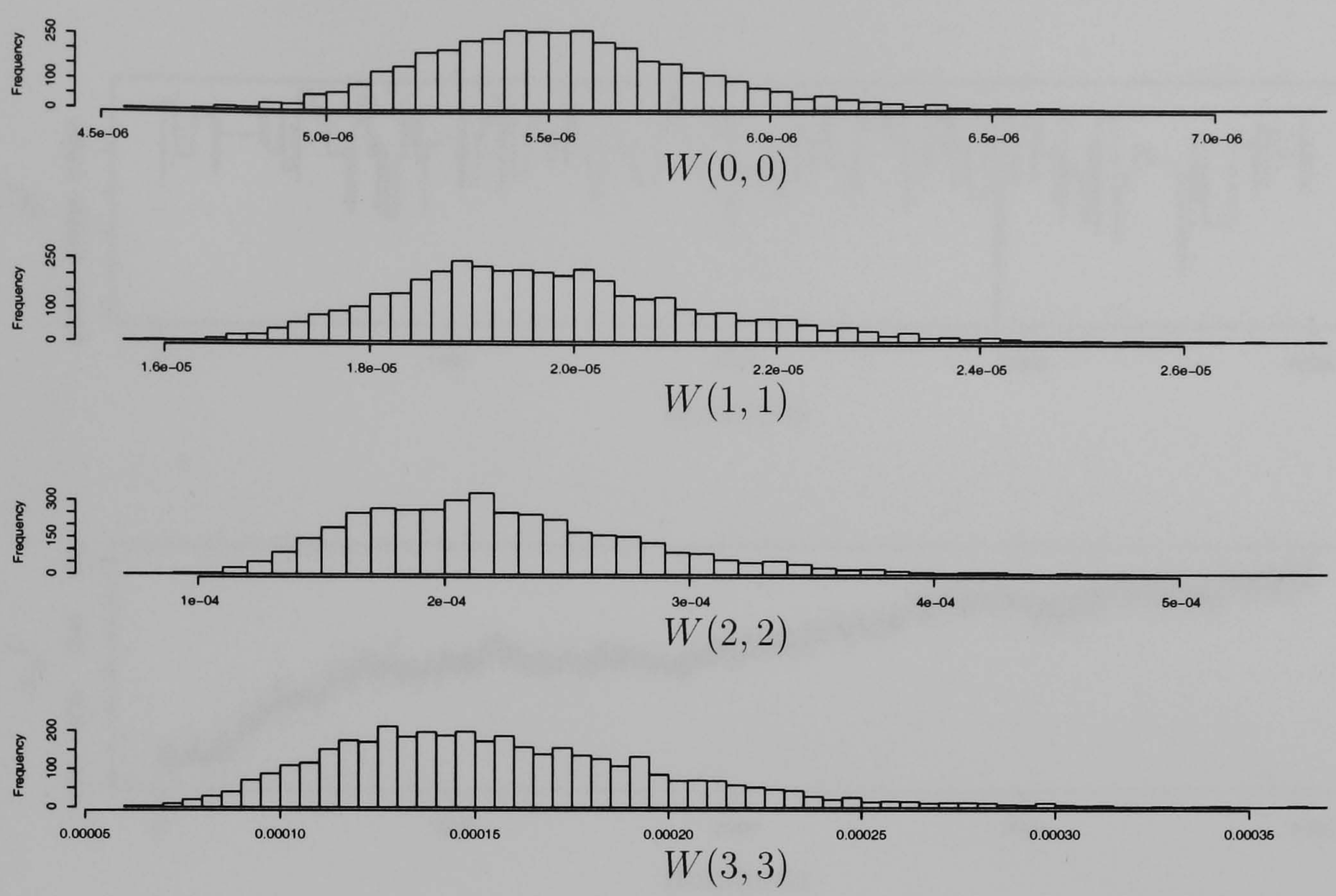


Figure 5.2: Histograms of the diagonal elements of the state variance matrix, W.

suggest convergence. Looking at the histograms Figure 5.2 for these components suggest that the model is underestimating this parameter.

Looking next at the factors, f, Figure 5.3 this shows the sampler having problems sampling efficiently for ϕ^f, with the random walk sticking at values and with a tendency to tend towards a value of 1, the true values being 0.8. The trace plot for ${\sigma_\eta}^f$ suggests convergence, however at a value well in excess of the true value of 0.01.

Looking at the τ^j's initially shows a similar picture. Figure 5.4 shows the random walk sticking for the ϕ^{τ^1} and the ${\sigma_\eta}^{\tau^1}$ converging to a value in excess of its true value, 0.01.

Looking at the τ^3 ISV shows a slightly different picture. Figure 5.5 shows the trace plots for this component, although convergence issues are apparent as when running the FSV model alone, the samples for ϕ are moving more freely, while the sample for σ_η appear to be sampling within the region of the true value, 0.01. The samples for the mean appear to be bellow their true values. This can be confirmed by looking at the histograms Figure 5.6.

Finally looking at the β parameters suggests that we are only sampling from the $N(0, 1)$ prior.

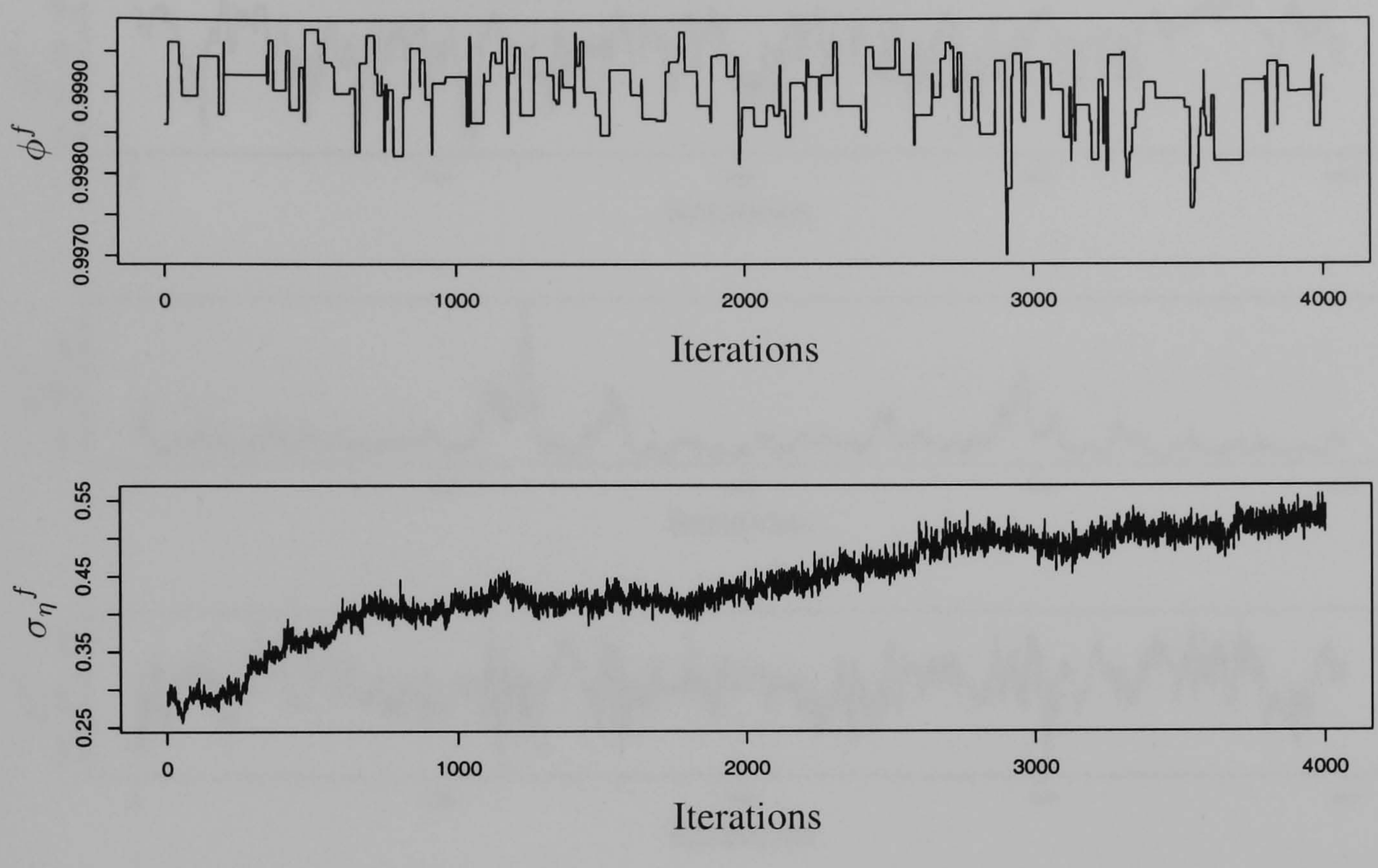


Figure 5.3: Trace plots for the two parameters of the f component of the FSV model

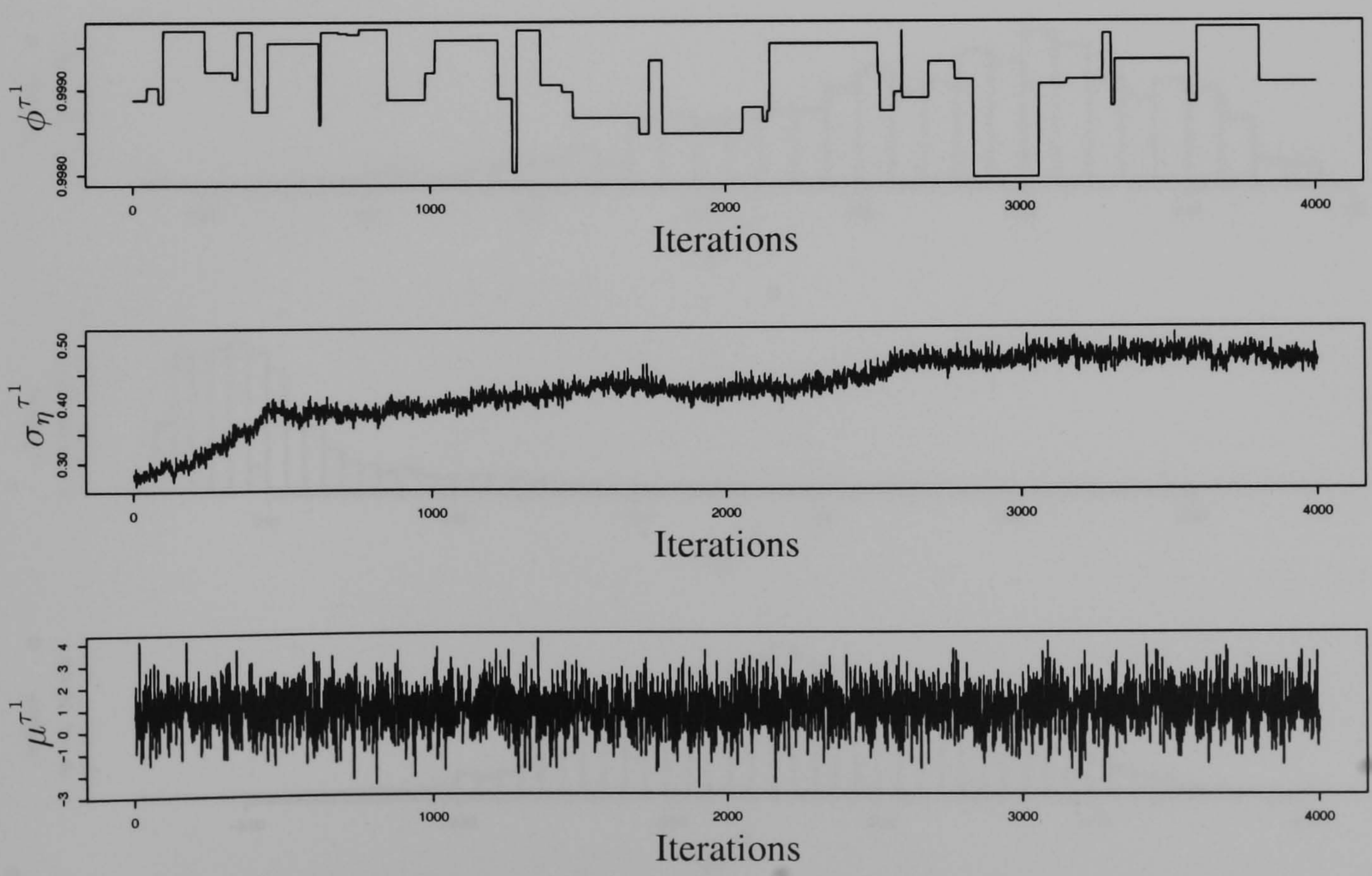


Figure 5.4: Trace plots for the three parameters of the τ^1 component of the FSV model

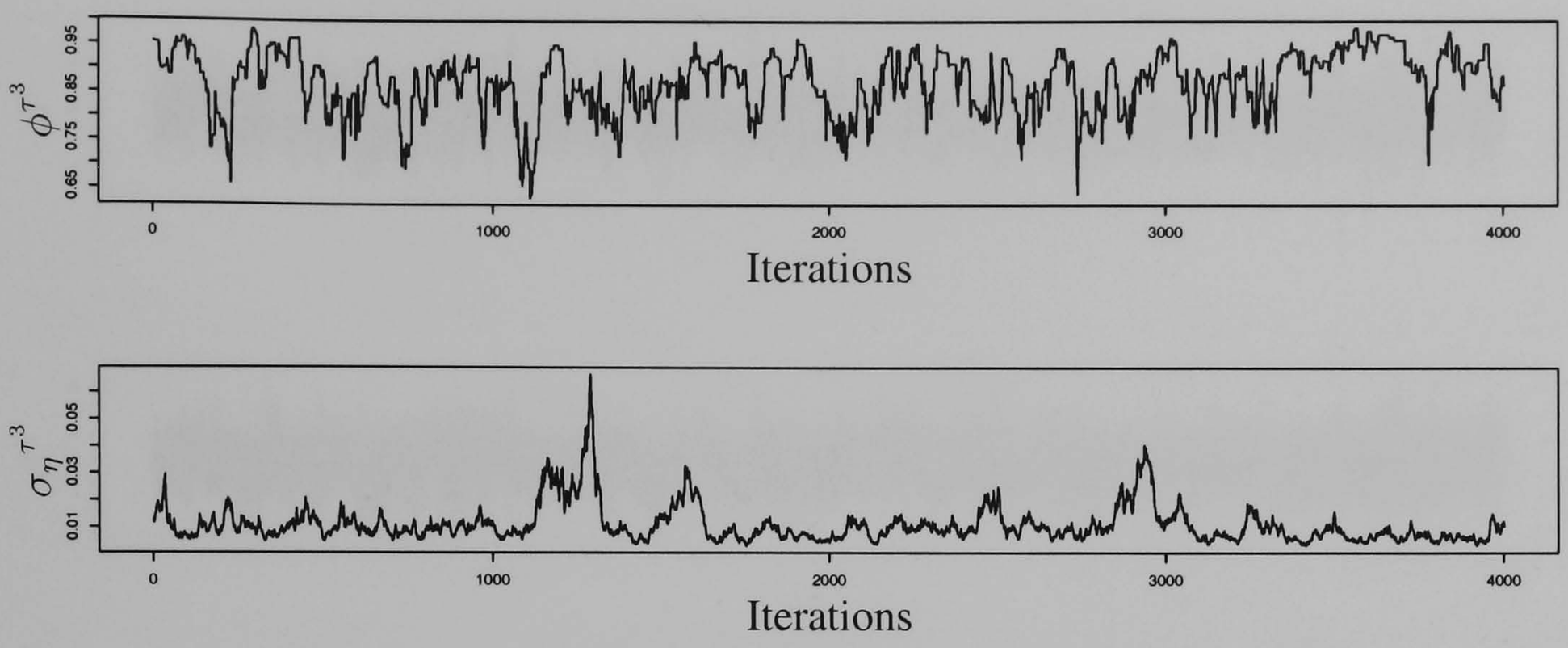


Figure 5.5: Trace plots for the three parameters of the τ^3 component of the FSV model.

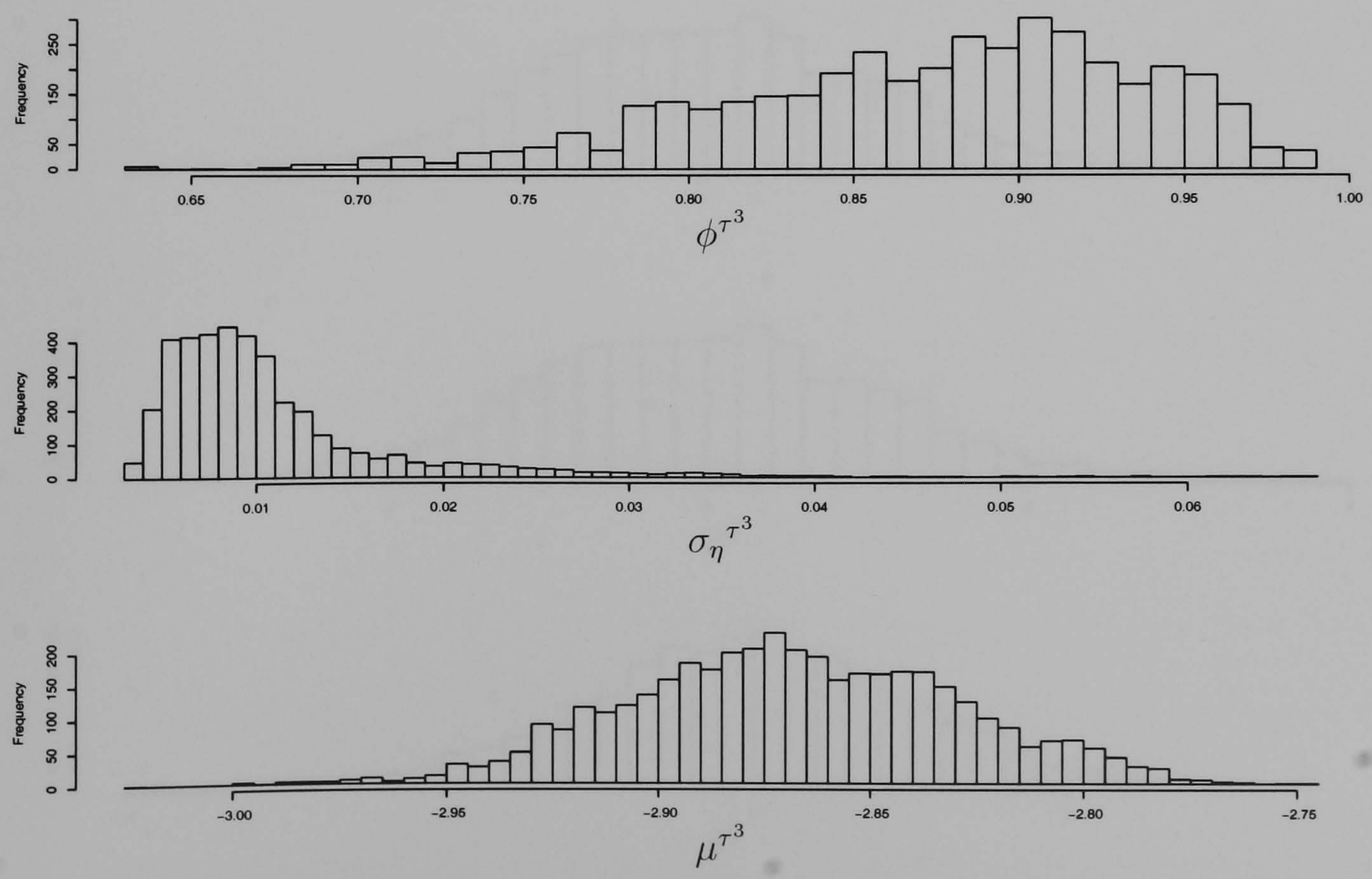


Figure 5.6: Histograms of samples from the parameters of the τ^3 ISV model.

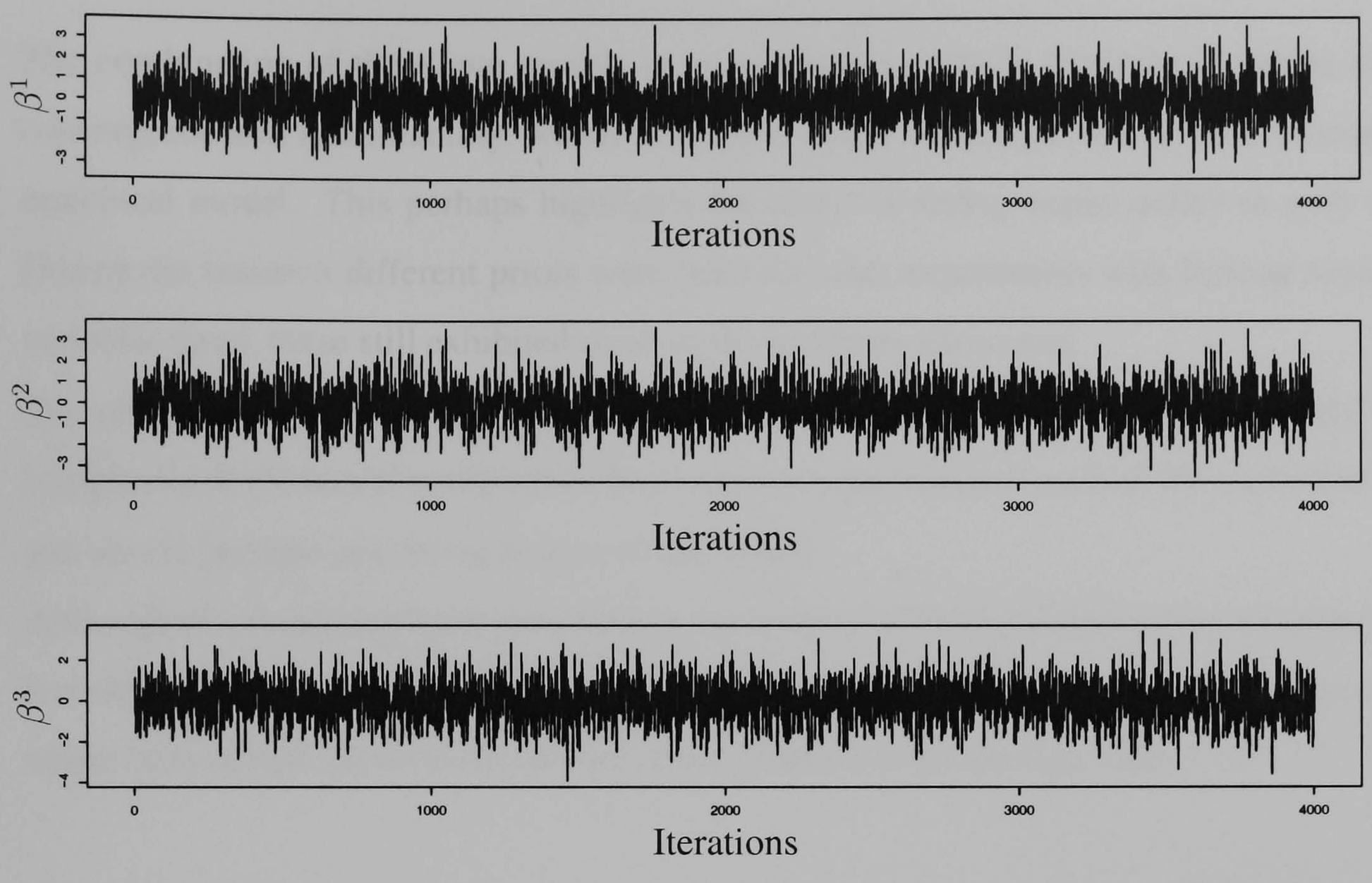


Figure 5.7: Trace plots for the three parameters of the β component of the FSV model.

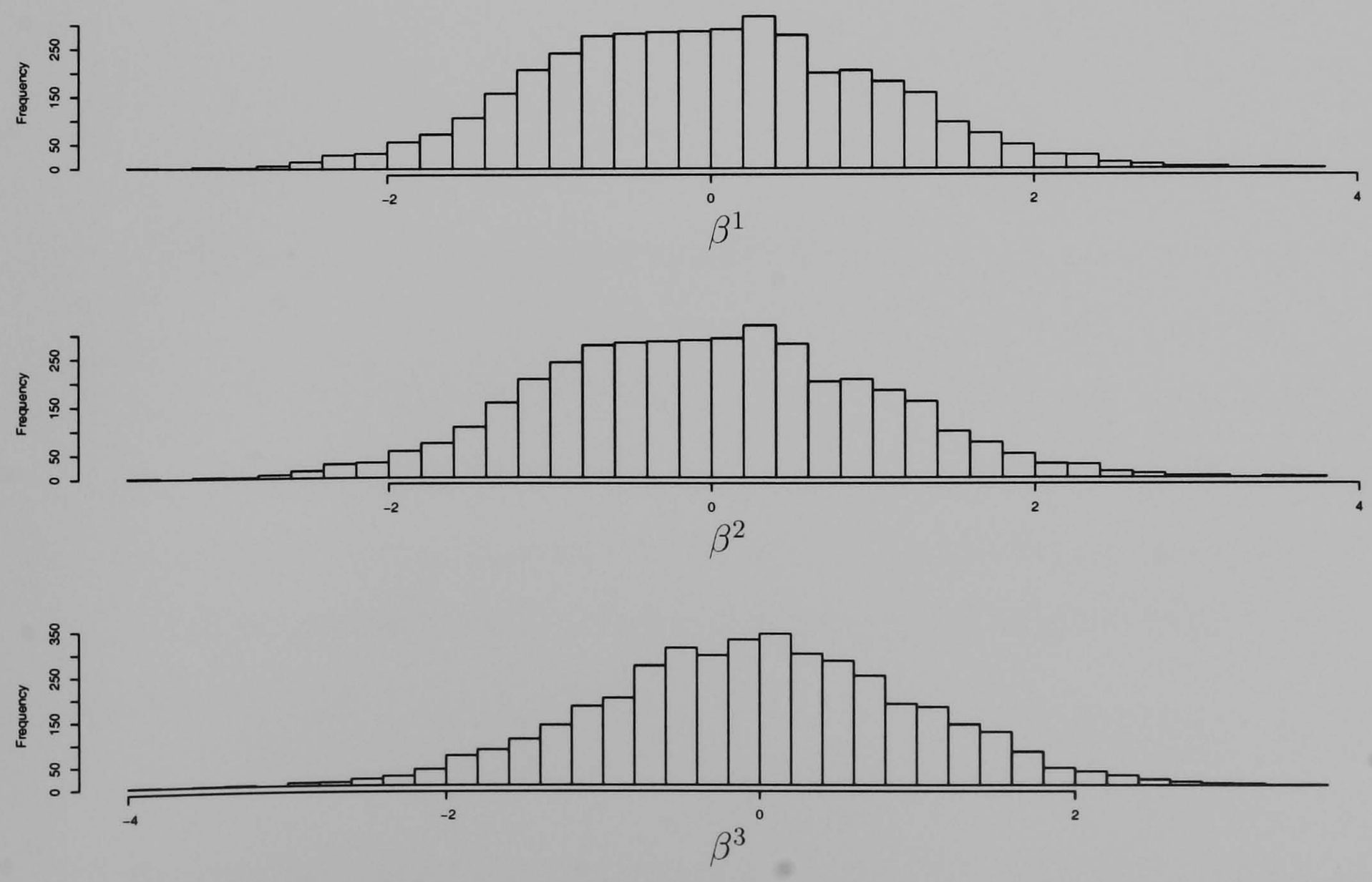


Figure 5.8: Histograms of the samples from the posterior for β.

5.4 Summary

The combination of these two models is an interesting exercise however it creates issues of convergence and identifiability, which perhaps is not to be unexpected in such a complex hierarchical model. This perhaps highlights the need for strong expert priors in such models. During the research different priors were tried and also experiments with leaving some of the variables fixed, these still exhibited some of the problems discussed.

The resolution of these issues will be left to continuing research but it seems likely that the complexity of the model compounds the problems experienced in each of the individual models and we are perhaps just trying to over fit the model.

Although this model has been used here in the context of financial time series it is possible that it could have applications beyond these and could possibly be used in many time series contexts where believe that the volatility of part of the model evolves through time.

Chapter 6

Portfolio Selection

In previous chapters we have demonstrated methods whereby we can make inference about future values of series of (log) share prices. This leads us to the obvious question, what do we do with these predictive values?

Jorion (1986) has stated that "One of the fundamental propositions of modern virtus theory of finance is that security risk is considered in the context of the portfolio." This leads to the concept that the portfolio of investments held is the natural unit of investigation. In this chapter we will look at how we can go from having predictive share prices to how we can aid an investor's decision making process.

6.1 The Mean Variance Approach

6.1.1 Modern Portfolio Theory

The origins of much portfolio theory is the work of Markowitz (1959). He believed that an investor would wish to maximise their returns on their investments for a given level of risk. The return on an investment being the proportional gain or loss on an investment over a set time period, i.e.

$$\mu_t = \frac{Y_t - Y_{t-1}}{Y_{t-1}} \tag{6.1}$$

where Y_t is the price of the investment at time t. Returns can of course be calculated over more than a single time period, however traditional approaches rely upon this single period return. The period of the return could be hourly, weekly, monthly or annually.

The nature of the investment in question will of course affect both the time scale and the ease of obtaining prices for that investment. The prices of some investment instruments, such as equities, bonds, currencies and other traded financial instruments are easily obtained as these are all traded on open markets with good information systems. The price of some other investment options are less easily obtained, for example property. In this thesis we shall only consider portfolios which comprise of equities.

The return on an equity not only depends on its price, but also on any dividends paid. This changes (6.1) to the following

$$\mu_t = \frac{Y_t - Y_{t-1} + dividend}{Y_{t-1}}.$$

We shall not consider the role of dividend in the following sections for two reasons. Firstly the time periods under consideration are less than the dividend payment points which are annual or occasionally bi-annual. Secondly the market price should reflect any dividends the market expects from the company and hence these are accounted for in the share price.

One important feature is that all an investor's investments should be considered, the interaction between the returns on investments is key to portfolio theory, a good portfolio is not necessarily one which consists of individually good investments but one which is good overall.

Risk in terms of financial instruments are those factors which cause uncertainty in the variation of the price for the instruments. "The quantification of this uncertainty gives rise to the measurement of risk." Watsham (1993).

Portfolio theory assumes that investors are *risk averse*. That is given two portfolios of investments with the same return, the investor would choose the one with the lowest risk, i.e. the one with the lowest uncertainty over future outcomes.

This leads us to consider how we could measure this risk. If risk can be thought of as uncertainty then a measure of uncertainty would be a reasonable measure of risk. Markowitz (1959) assumed that asset returns are random and that they are independently normally distributed.

This allows the investor to describe the expected return and expected risk of an investment as the expectation and variance of a random variable. In the Markowitz (1959) model the investment risk is represented by the variance of the returns distribution. This is a reasonable assumption as the larger the variance the less certain we are about the level of return and hence the greater the risk of the investment. Hence an investor's utility function can be measured simply in terms of the risks and returns or the mean and variance of the returns and hence the term "Mean Variance" approach. Consideration of the returns and variance of a portfolio as the natural unit of measure tends to lead to diversified portfolio with a range of stocks being held, this is intuitively logical.

The assumption that the returns of a well diversified portfolio are normally distributed are based the Central Limit Theorem. The probability weighted mean of a random variable will tend to normally distributed even if the individual observations of the variable are not. We will show that the expected return on a portfolio is a weighted mean of the returns on individual shares and hence the central limit theorem can be invoked to support the assumption of normality of portfolio returns, at least for a well diversified portfolio.

The Markowitz approach was to derive efficient portfolios, a portfolio is considered *efficient* if no other portfolio offers the same or higher expected returns with the same or lower level of risk, or lower risk with the same or higher expected returns. Let us look at this more precisely.

The Expected rate of return on a portfolio is defined as:-

$$E = \sum_{i=1}^{p} \alpha_i \mu_i$$

where, E is the expected return on the portfolio, α_i is the proportion of wealth invested in stock i and μ_i is the return on stock i. This also be written as:

$$E = \alpha' \mu$$

where, α is a vector of length p, where p is the number of shares to be considered for the portfolio, representing the proportions held in each stock and μ is a vector of length p representing

the returns on each stock. The Variance of a portfolio is

$$V = \sum_{i=1}^{p} \sum_{j=1}^{p} \alpha_i \alpha_j \sigma_{ij}$$

where, V is the Variance on the portfolio as a whole and σ_{ij} is the covariance between the return on the i^{th} and the j^{th} stocks. This can also be written as

$$V = \alpha' \Sigma \alpha$$

where, α is a vector of length p representing the proportions held in each stock and Σ is the $p \times p$ variance matrix of returns.

The original Markowitz (1959) monograph essentially concerns itself with the derivation of *Efficient Portfolios*. To be efficient a portfolio must meet the following criteria, it must be a legitimate portfolio, if any legitimate portfolio has the greater expected return it must also have greater variance of return and if any legitimate portfolio has a smaller variance of returns it must also have a smaller expected return. A legitimate portfolio is one which conforms to any constraints placed on it, for example some markets do not allow short sales, therefore a portfolio which contained short positions would not be legitimate. The portfolios held by banks to cover their legal liquidity requirements are regulated, with limits placed on the proportion that can be held in certain assets.

The monograph's approach is to trace the *Efficient Frontier*. These are the portfolios which lie between the extremes of the portfolio with the smallest variance and the portfolio with the maximum expected return. Each of these are *efficient portfolios* which the investor would choose would depend on their attitude to risk. If they were unconcerned about risk they would choose the portfolio which gave the maximum expected return.

The initial approach adopted in the monograph is to consider a geometric approach to *efficient sets*. The idea behind this approach is to move from the portfolio with the smallest variance to the portfolio with the highest expected return. Critical sets are defined which contain all portfolios in the subspace and which for the same expected return have larger variance than P. The set of efficient portfolios is found by moving in the direction of increasing expected return,

transferring from one critical line to another whenever they intersect until the portfolio with the maximum expected return is reached. Merton (1972) demonstrated an analytical approach to deriving the frontier portfolio, which we will demonstrate here in in vector matrix notation.
A portfolio is a frontier portfolio if it has the minimum variance amongst portfolios that have the same expected return. A portfolio P is a frontier portfolio if and only if α_P the p vector of portfolio weights is the solution to the following quadratic program

$$\min V = \alpha' \Sigma \alpha \qquad \text{subject to} \qquad \alpha' \mathbf{1} = 1 \qquad \text{and} \qquad \alpha' \mu = R$$

where, R is a set rate of return expected by the investor and $\mathbf{1}$ is a vector of length p of 1's. Forming the Lagrangian, α is the solution to:-

$$\min L = \alpha' \Sigma \alpha + \lambda(1 - \alpha' \mathbf{1}) + \gamma(R - \alpha' \mu).$$

Taking a partial derivative with respect to α

$$\begin{aligned} \frac{\partial L}{\partial \alpha} &= 2\Sigma\alpha - \lambda\mathbf{1} - \gamma\mu = 0 \\ \Rightarrow 2\Sigma\alpha &= \lambda\mathbf{1} + \gamma\mu \\ \Rightarrow \alpha &= \frac{\lambda}{2}\Sigma^{-1}\mathbf{1} + \frac{\gamma}{2}\Sigma^{-1}\mu. \end{aligned} \tag{6.2}$$

Using the two constraints

$$R = \alpha'\mu = \frac{\lambda}{2}\mu'\Sigma^{-1}\mathbf{1} + \frac{\gamma}{2}\mu'\Sigma^{-1}\mu$$

$$1 = \alpha'\mathbf{1} = \frac{\lambda}{2}\mathbf{1}'\Sigma^{-1}\mathbf{1} + \frac{\gamma}{2}\mathbf{1}'\Sigma^{-1}\mu$$

and letting

$$A = \mathbf{1}'\Sigma^{-1}\mu = \mu'\Sigma^{-1}\mathbf{1}$$

$$B = \mu'\Sigma^{-1}\mu \qquad C = \mathbf{1}'\Sigma^{-1}\mathbf{1}$$

we obtain

$$R = \frac{\lambda}{2}A + \frac{\gamma}{2}B \tag{6.3}$$

$$1 = \frac{\lambda}{2}C + \frac{\gamma}{2}A. \tag{6.4}$$

Using (6.3) and (6.4)

$$\lambda = \frac{2R - \gamma B}{A}$$
$$\Rightarrow \gamma = \frac{2RC - 2A}{BC - A^2}$$

Again using (6.3) and (6.4), solving for λ we obtain

$$\lambda = \frac{2B - 2RA}{BC - A^2}.$$

Substituting these back into our original solution for α (6.2)

$$\alpha = \frac{B - RA}{BC - A^2}\Sigma^{-1}\mathbf{1} + \frac{RC - A}{BC - A^2}\Sigma^{-1}\mu.$$

Hence the efficient portfolio can be found by calculating the value of α for all reasonable values of the expected return on the portfolio, R. This approach can be expanded to include a risk free asset. When a risk free asset is included the investor can choose a portfolio which consists of both risky and riskless assets allowing them to reduce the overall risk of their portfolio. A riskless asset has a guaranteed return and hence has zero variance. Hence for any given level of return the variance of the portfolio will be reduced if the risk free assets is included. Hence the problem becomes:-

$$\min V = \alpha'\Sigma\alpha \qquad \text{subject to} \qquad \alpha'\mu + (1 - \alpha'\mathbf{1})r_f = R$$

where:-

- α is the vector of portfolio weights in risky assets.
- r_f is the rate of return on the riskless asset.

Forming the Lagrangian, we know that α_p is the minimum of:-

$$\alpha'\Sigma\alpha + \lambda(R - \alpha'\mu - (1 - \alpha\mathbf{1})r_f).$$

Taking the partial derivative

$$\begin{aligned}\frac{\partial L}{\partial \alpha} &= 2\Sigma\alpha - \lambda\mu + \lambda\mathbf{1}r_f \\ \Rightarrow 2\Sigma\alpha &= \lambda(\mu - \mathbf{1}r_f) \\ \Rightarrow \alpha &= \frac{\lambda}{2}\Sigma^{-1}(\mu - \mathbf{1}r_f).\end{aligned}$$

From the constraint

$$\begin{aligned}&\frac{\lambda}{2}\mu'\Sigma^{-1}(\mu - \mathbf{1}r_f) + \left(1 - \frac{\lambda}{2}\mathbf{1}'\Sigma^{-1}(\mu - \mathbf{1}r_f)\right)r_f = R \\ \Rightarrow\ &\lambda\mu'\Sigma^{-1}\mu - \lambda\left(\mu'\Sigma^{-1}\mathbf{1}r_f\right) + 2r_f - \lambda\mathbf{1}'\Sigma^{-1}\mu r_f + \lambda\mathbf{1}'\Sigma^{-1}\mathbf{1}r_f^2 = 2R.\end{aligned}$$

Again letting

$$\begin{aligned}A &= \mathbf{1}'\Sigma^{-1}\mu = \mu'\Sigma^{-1}\mathbf{1} \\ B &= \mu'\Sigma^{-1}\mu \qquad C = \mathbf{1}'\Sigma^{-1}\mathbf{1}\end{aligned}$$

then:-

$$\begin{aligned}\lambda B - \lambda A r_f - \lambda A r_f + \lambda C r_f^2 &= 2R - 2r_f \\ \Rightarrow \lambda &= \frac{2R - 2r_f}{B - 2Ar_f + Cr_f^2}.\end{aligned}$$

Substituting this back into our original equation for α we have:-

$$\alpha = \frac{2R - 2r_f}{B - 2Ar_f + Cr_f^2}\Sigma^{-1}(\mu - \mathbf{1}r_f).$$

These approaches all can be used to delineate the *efficient frontier*, they do not however arrive

at a single optimal portfolio. One interpretation of this is that the investor knows what level of return they are seeking and therefore they can adopt a portfolio, to meet this return, at the lowest level of risk. This presupposes that the investor is not capable of making decisions as to what trade offs between risk and return they are prepared to take.

6.1.2 Developments

One approach to developing the mean variance framework of portfolio selection came from Sharpe (1970), he defined an investors indifference curve as

$$V = \beta + \lambda E$$

where β indicates the horizontal intercept and λ indicates the slope of the line. Under Sharpe (1970) model, "The objective of any given investor is to selected the best portfolio, in other words to find the feasible portfolio lying on the the most desirable *indifference curve*." The investor aims to find the indifference curve which is tangential to the efficient frontier and with the smallest intercept, i.e to minimise β. Hence the objective is to:-

$$\min \beta = -\lambda E + V \qquad \text{subject to} \qquad \alpha' \mathbf{1} = 1$$

Forming the Lagrangian and writing the equation in full the aim is to minimise L where:-

$$L = -\lambda \alpha' \mu + \alpha' \Sigma \alpha + \gamma(1 - \alpha' \mathbf{1})$$

$$\Rightarrow \frac{\partial L}{\partial \alpha} = -\lambda \mu + 2\Sigma\alpha - \gamma \mathbf{1} = 0$$

$$\Rightarrow 2\Sigma\alpha = \gamma \mathbf{1} + \lambda \mu \tag{6.5}$$

$$\Rightarrow \alpha = \frac{\gamma}{2}\Sigma^{-1}\mathbf{1} + \frac{\lambda}{2}\Sigma^{-1}\mu. \tag{6.6}$$

Since $\alpha'\mathbf{1} = 1$ we have

$$1 = \frac{\gamma}{2}\mathbf{1}'\Sigma^{-1}\mathbf{1} + \frac{\lambda}{2}\mathbf{1}'\Sigma^{-1}\mu$$
$$\Rightarrow \gamma = \frac{2 - \lambda\mathbf{1}'\Sigma^{-1}\mu}{\mathbf{1}'\Sigma^{-1}\mathbf{1}}.$$

Substituting back into our equation for α, (6.6)

$$\alpha = \frac{1}{2}\left\{\frac{2 - \lambda\mathbf{1}'\Sigma^{-1}\mu}{\mathbf{1}'\Sigma^{-1}\mathbf{1}}\right\}\mathbf{1}'\Sigma^{-1}\mathbf{1} + \frac{\lambda}{2}\mathbf{1}'\Sigma^{-1}\mu$$

Hence for a given value of λ the best portfolio for an investor can be calculated. This of course assumes we have values for μ and Σ. In the work of both Markowitz (1959) and Sharpe (1970) these are based on historic point estimates. The λ parameter reflects the investors attitude towards risk. The higher the value for λ the more the investor is interested in returns on the portfolio over the variance or risk of those returns. As $\lambda \to \infty$ then the investor is completely risk neutral not caring about the uncertainty of the returns. As $\lambda \to 0$ then the investor is completely risk averse only wishing to minimise variance and hence risk.

6.2 Utility Based Approaches

The methods outlined in Section 6.1 rely upon estimates for the expectation and variance of the portfolio returns and these are indeed critical to effective portfolio selection. Novomestky (1997) states "a consistent forecast of the predictive distribution of an international assets returns enables an investor to construct a portfolio which is mean variance optimal." While Frost and Savarino (1986) state that "when portfolio optimisation is implemented using historical characteristics of security returns, estimation error can degrade the desirable properties of the investment portfolio that is selected."

To overcome the problems with using historic estimates various approaches have been suggested. Novomestky (1997) uses a Bayesian approach to dynamic seemingly unrelated regression as a means of forecasting the one-step ahead, conditional distribution of asset returns. While Frost and Savarino (1986), Broad and Sutcliffe (1994) and Jorion (1986) all use Bayes-

Stein methodologies to make inference about the one-step forecast returns distribution. Young and Lenk (1998) use hierarchical Bayes methodologies to the same end.

Adopting the Bayesian paradigm naturally leads us to consider investor utility maximisation as method of portfolio selection. As discussed in Section 1.5 a rational investor would always wish to maximise their utility on an investment. An investment can be considered to be a gamble with an uncertain monetary reward and hence utility theory is applicable to investment decisions.

If an investor has exponential utility for money, $u(x) = 1 - e^{-\lambda x}$, where λ is parameter representing the investors level of risk aversion then the investor's utility for an uncertain monetary reward $X \sim N(a, b)$ is given by

$$\begin{aligned}
\mathrm{E}(u(X)) &= \int u(x)f(x)dx \\
&= \int_{-\infty}^{\infty}(1 - e^{-\lambda x})(\frac{1}{\sqrt{2\pi b}}e^{(\frac{-1}{2b}(x-a)^2)})dx \\
&= \int_{-\infty}^{\infty}\frac{1}{\sqrt{2\pi b}}e^{(\frac{-1}{2b}(x-a)^2)} - \frac{1}{\sqrt{2\pi b}}\int_{-\infty}^{\infty}e^{-\lambda x}e^{(\frac{-1}{2b}(x-a)^2)}dx \\
&= 1 - \frac{1}{\sqrt{2\pi b}}\int_{-\infty}^{\infty}e^{-\lambda x}e^{(\frac{-1}{2b}(x-a)^2)}dx \\
&= 1 - \frac{1}{\sqrt{2\pi b}}\int_{-\infty}^{\infty}e^{\frac{-1}{2b}(2b\lambda x)}e^{\frac{-1}{2b}(x-a)^2}dx \\
&= 1 - \frac{1}{\sqrt{2\pi b}}\int_{-\infty}^{\infty}e^{\frac{-1}{2b}(2b\lambda x+(x-a)^2)}dx.
\end{aligned}$$

We can rewrite the exponential term as

$$2b\lambda x + (x-a)^2 = (x-(a-\lambda b))^2 + 2\lambda b(a - \frac{\lambda b}{2}).$$

Putting this back into the integral:-

$$\begin{aligned}
\mathrm{E}(u(X)) &= 1 - \frac{1}{\sqrt{2\pi b}}\int_{-\infty}^{\infty}e^{\frac{-1}{2b}(x-(a-\lambda b))^2}e^{\frac{-1}{2b}2\lambda b(a-\frac{\lambda b}{2})}dx \\
&= 1 - e^{-\lambda(a-\frac{\lambda b}{2})}\frac{1}{\sqrt{2\pi b}}\int_{-\infty}^{\infty}e^{\frac{-1}{2b}(x-(a-\lambda b))^2}dx.
\end{aligned}$$

Recognising that the integrand is a normal density function and therefore integrates to one we

obtain

$$\mathrm{E}(u(X)) = 1 - e^{-\lambda(a-\frac{\lambda b}{2})}.$$

Hence the utility of an uncertain monetary reward is a function of the two parameters of the distribution from which that reward comes and the investors risk aversion parameter.
As in Section 6.1.1 we consider portfolio returns to be jointly Gaussian; $R \sim N(\mu, \Sigma)$. Then the return on an initial investment M in a portfolio α is

$$M\alpha' R \sim N(M\alpha'\mu, M^2\alpha'\Sigma\alpha).$$

This is obviously of the form seen in the example above and hence the investor's utility assuming an exponential utility for money as above this is maximised by maximising the power term of the exponential. Hence the investor's utility is maximised by solving the quadratic program:-

$$\max \alpha'\mu - \frac{1}{2}\eta\alpha'\Sigma\alpha, \qquad \text{subject to} \qquad \alpha'\mathbf{1} = 1$$

where $\eta = M\lambda$. Forming the Lagrangian

$$L = \alpha'\mu - \frac{1}{2}\eta\alpha'\Sigma\alpha + \gamma(1 - \alpha'\mathbf{1})$$

and taking partial derivatives

$$\frac{\partial L}{\partial \alpha} = \mu - \eta\Sigma\alpha - \gamma\mathbf{1} = 0$$

we obtain

$$\alpha = \eta^{-1}\Sigma^{-1}\mu - \gamma\eta^{-1}\Sigma^{-1}\mathbf{1}.$$

Using the constraint

$$1 = \alpha'\mathbf{1} = \eta^{-1}\mathbf{1}'\Sigma^{-1}\mu - \gamma\eta^{-1}\mathbf{1}'\Sigma^{-1}\mathbf{1}$$
$$\Rightarrow \gamma = \frac{-1}{\eta}\left(\frac{1 - \eta^{-1}\mathbf{1}'\Sigma^{-1}\mu}{\mathbf{1}'\Sigma^{-1}\mathbf{1}}\right).$$

Substituting back into our original equation for α we get

$$\alpha = \eta^{-1}\Sigma^{-1}\mu + \frac{1 - \eta^{-1}\mathbf{1}'\Sigma^{-1}\mu}{\mathbf{1}'\Sigma^{-1}\mathbf{1}}\Sigma^{-1}\mathbf{1}. \tag{6.7}$$

Thus for a given value of λ it is possible to calculate an investor's optimal portfolio. This option was not available in the traditional Markowitz approach where we only calculated the *efficient frontier* and did not really consider the individual investors preferences. Leaving the decision on choosing the right portfolio to the investor who would be presented with a range of options from the *efficient frontier*. Equation (6.7) is of course an unconstrained maximisation, with extreme portfolio positions as possible solutions, which are unlikely to be taken in reality, we will look at methodologies for practical constrained maximisation in Section 6.4.3.

There is a fundamental problem with the solution provided by (6.7), it still relies upon one step ahead forecasts for the μ and Σ parameters, that is there is no dynamic to the model for price evolution. This was recognised by Quintana and West (1987) who developed a mode for price evolution which is applied to the portfolio selection problem by Quintana (1992).

The MCMC methodologies presented in Chapter 3, Chapter 4 and Chapter 5 allow a radically different approach to the portfolio selection problem.

6.3 New Approaches

The methodologies presented in Chapter 3, Chapter 4 and Chapter 5 allow for direct simulation from the full joint posterior distributions of each series under consideration for the portfolio and hence by calculation from the returns distribution. These methodologies integrate over all model uncertainty.

The advantage of this new approach is that rather than relying upon summary statistics of the predicted values of the series we have forecast distributions giving a fuller picture of the pre-

dictive returns.

To use these forecast distributions the first step is to calculate the returns, which are the proportional gain/loss of the k step ahead forecast compared to the last known data point. This gives us a collection of simulated return vectors $\{R^{(i)}\}, 1 \leq i \leq N$, where N is the number of iterations of the MCMC scheme, generated by the MCMC scheme, after an appropriate burn-in period is discarded.

One simple option would be to calculate the summary statistics from these simulated returns and use these in the standard Markowitz utility maximisation scheme (6.7). This however would result in sub-optimal portfolios as the distributional information is lost in this process.

A preferable option is to use all the information provided by the returns distribution. If we assume a particular investment strategy, α it is trivial to compute the utility for any one particular return vector, $R^{(i)}$, assuming the investors utility for money to be of the form $u(x) = 1 - e^{-\lambda x}$ as previously

$$u(R^{(i)}) = 1 - e^{-\eta \alpha' R^{(i)}}$$

where $\eta = M\lambda$ as previously. We have the collection of such return vectors $R^{(i)}$, from the MCMC run and we can easily calculate the utility of all of these, for any particular α. In Section 1.5 we stated that the utility of a gamble is the expected utility of that gamble, i.e.

$$u(G) = \mathrm{E}(u(G)).$$

Now in this case the gamble is the investment strategy α and the expected utility of that gamble for a set of return vectors $\{R^{(i)}\}, 1 \leq i \leq N$ is given by:

$$\lim_{N \to \infty} \frac{\sum_{i=1}^{N} \left\{ 1 - e^{-\eta \alpha' R^{(i)}} \right\}}{N} = \mathrm{E}(u(\alpha)) \tag{6.8}$$

Hence the utility of a particular investment strategy α is the sample mean of the utility for a collection of returns vectors, $R^{(i)}$. Given (6.8) and the $R^{(i)}$'s from an MCMC scheme output we can evaluate the utility for a particular investment strategy α. We now require a method whereby we maximise this with respect to α to find the investor's optimal portfolio.

6.4 Maximisation Techniques

In this section we will look at the practical considerations of maximising (6.8) in both an unconstrained and a constrained manner.

6.4.1 Unconstrained Maximisation

As discussed in Section 6.3 it is possible to use the summary statistics calculated from the forecast returns from the MCMC scheme output and evaluate equation (6.7) to find the portfolio which maximises the investor's utility, given their risk preference. This calculation is trivial to program in **R** and it is possible to investigate the investors response to risk by varying the value of λ. This can be compared with the classical Markowitz portfolio based on historic one-step returns as calculated from the data. The advantage of the use of the MCMC forecast distributions is that it contains a model dynamic and hence more than one step ahead returns are able to be calculated.

We have already stated in Section 6.3 that only considering the summary statistic method leads to sub-optimal portfolios and that it is preferable to maximise equation (6.8) with respect to α. Such unconstrained optimisations are however only of limited use, in both practical terms and due to regulatory requirements the extreme short positions which these tend to generate can not be implemented by real investors. What is required in some form of constrained optimisation.

6.4.2 Examples

To illustrate the unconstrained optimisation let us consider our data from the locally constant example first introduced in the MCMC context in Section 3.2.2. We can for example, sample from the 50 step ahead forecast distribution at the same time as sampling the model parameters, the returns relative to the last known data point can easily be calculated.

We can calculate the returns of the historic data and its summary statistics. These can then be used in (6.7) to find the optimum portfolio depending upon η. Figure 6.1 shows the solution to this equation for varying η's, what we see is a fairly static portfolio with the investment decision being little affected by the risk preference of the investor.

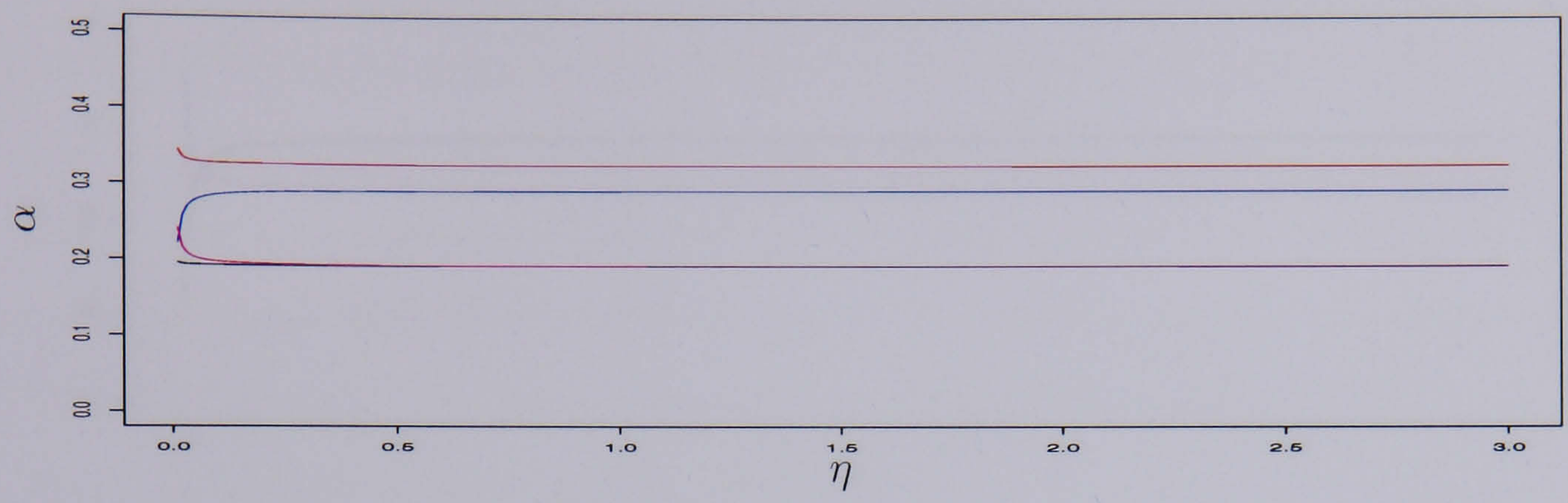


Figure 6.1: Plots of portfolio weights α for varying levels of risk preference η.

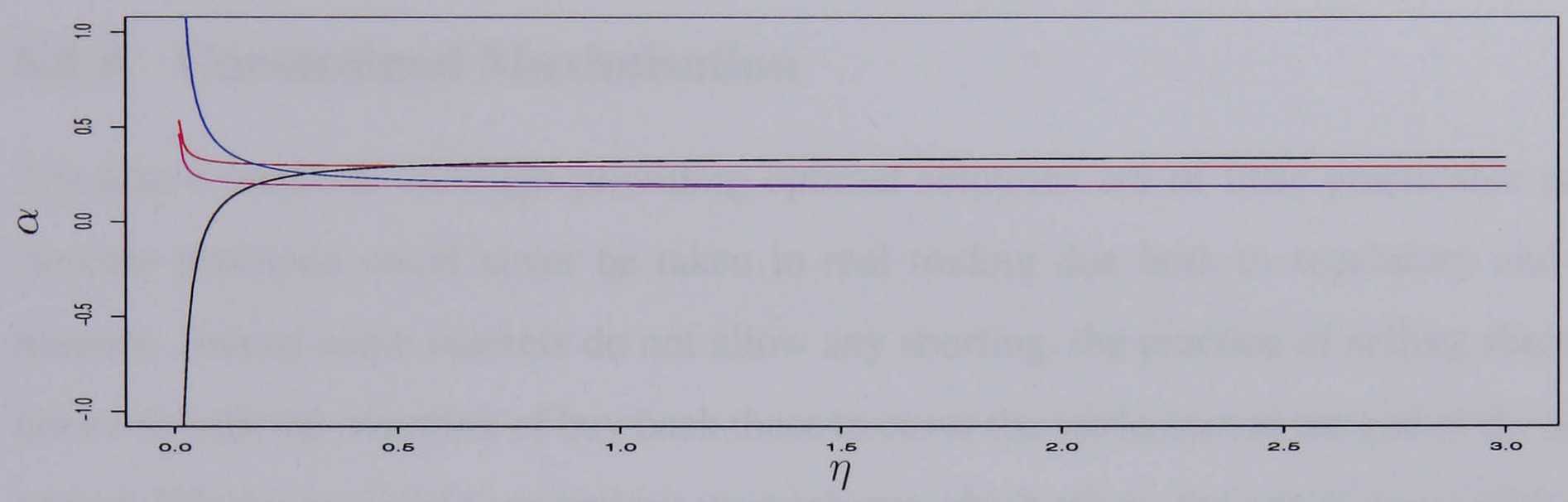


Figure 6.2: Plots of portfolio weights α for varying levels of risk preference η for a 10 step ahead forecast.

As stated above one limitation to the Markowitz approach is that it only considers the historic one step returns, the output from the forecast from the MCMC schemes allows us to consider time points further into the future.

Again using the results from the MCMC example in Section 3.2.2,Figure 6.2 and Figure 6.3 show the solution for (6.7) using the summary statistics of the 10 and 50 step ahead returns distributions, for varying values of η. This shows that the portfolio to be held by the investor changes over time, with less extreme positions for a risk neutral investor being taken for the 50 step ahead forecast as opposed to the 10 step ahead forecast. Both these graphs show very different portfolios than that suggested by the traditional Markowitz scheme, Figure 6.1. They also contain some extreme short positions for the risk neutral investor, such short positions might not be achievable in real trading.

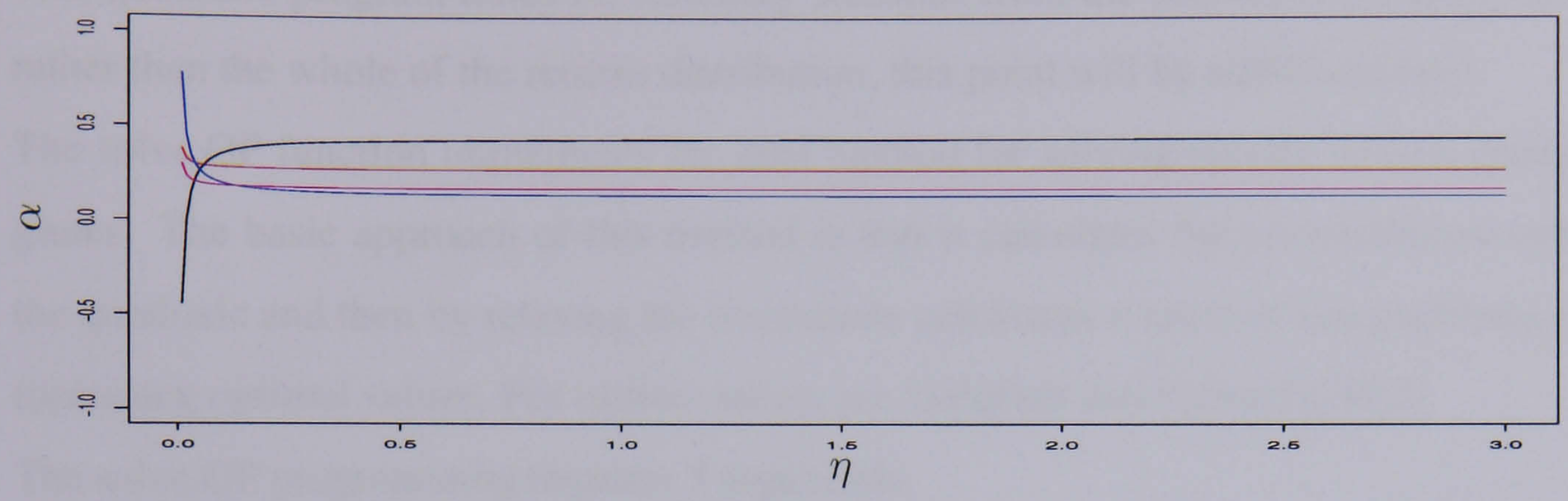


Figure 6.3: Plots of portfolio weights α for varying levels of risk preference η for a 50 step ahead forecast.

6.4.3 Constrained Maximisation

The above methods although providing optimal solutions are of little practicable use. Such extreme positions could never be taken in real trading due both to regulatory and practical reasons. Indeed some markets do not allow any shorting, the practice of selling shares you do not own, with the intention of buy back these to cover the settlement at the end of the settlement period. What is required is an optimisation scheme which allows the constraining of the possible solutions, either to prevent shorting all together or placing realistic limits on this behaviour. Two different methods of constrained optimisation were considered.

6.4.4 Constrained Maximisation of the Markowitz solution

What is required is a method of optimising the utility function that produces an optimal solution in a timely manner and which is also accurate. The aforementioned **R** programming language has several libraries, one of which is **quadprog**, this is a programming package designed to solve quadratic programs. Recall that the unconstrained optimisation in section Section 6.1.1 is a quadratic, to which we can add a further constraint, preventing negative weights and hence the solution becomes the solution to the following quadratic program:

$$\max \alpha'\mu - \frac{1}{2}\eta\alpha'\Sigma\alpha, \qquad \text{subject to} \qquad \alpha'\mathbf{1} = 1 \qquad \text{and} \qquad \alpha \geq 0 \tag{6.9}$$

where $\eta = M\lambda$ and the second constraint prevents α taking a value less than zero, i.e. that there are no short sales. This can be modified so as to prevent short sales in excess of set limits.

This quadratic program relies on summary statistics from the predictive posterior distribution rather than the whole of the returns distribution, this point will be addressed later.

The solve.QP function implements the dual method for solving strictly convex quadratic programs. The basic approach of this method is that it calculates the unconstrained optimum of the quadratic and then by relaxing the constraints calculates a series of sub problems each with increasing optimal values. For further details see Goldfarb and Indnani (1982).

The solve.QP programming requires 5 arguments.

- Dmat which is the matrix in the quadratic
- dvec which is the vector in the first element of the quadratic
- Amat which is the matrix of the constraints
- bvec which is the vector of the constraints
- meq which specifies the number of equality constraints.

In our example this corresponds to

- Dmat $= \lambda\Sigma$
- dvec $= \mu$
- Amat if Σ is $n \times n$ then Amat is $(n+1) \times n$ where the first row is a vector of 1's and then the $n \times n$ matrix remaining is an identity matrix
- bvec is a vector of length $(n+1)$ the first element is a one corresponding to the sum and the remainder are zeros corresponding to the equality constraints
- meq =1, i.e.the $\alpha'\mathbf{1} = 1$ constraint

This produces the constrained optimal solution to the quadratic (6.9), for a particular value of η this can easily be incorporated into a loop which calculates the optimum for varying values of η to allow investigation of the portfolio for varying risk preference levels.

6.4.5 Stochastic Simulation

The second approach to maximising an investor's utility from a portfolio is based on stochastic simulation. This method is based on calculating the utility using equation (6.8) for a large number of potential portfolios and keeping that which gives the highest utility. When generating the potential portfolios we can impose any restrictions we require, for example preventing or limiting short positions. This method has the advantage over that described in Section 6.4.4 in that it does not rely on summary statistics but is based on the whole series of simulated future values. To clarify this let us consider the maximisation algorithm.

Algorithm

1. Set an initial value for $\eta = M\lambda$ and an initial set of portfolio weights, α.

2. Calculate the utility of this portfolio using equation (6.8) and store this α as the current optimum.

3. Take the current optimum and add a small amount of stochastic noise to each of the first $(p-1)$ elements of the vector, i.e. $\alpha_i = \alpha_i + \epsilon_i$ where $\epsilon_i \sim N(0, \sigma^2)$. This noise is generated from independent $N(0, \sigma^2)$ distributions, the value of σ^2 is set to some small value. Calculate the p^{th} element from $1 - \sum_{i=1}^{(p-1)}(\alpha_i + \epsilon_i)$.

4. Check to see that all elements of the proposed vector are legitimate, for example if we require a portfolio with no shorting each element must fall within the simplex. Repeat step 3 until a legitimate portfolio is proposed.

5. Calculate the utility using equation (6.8) for the proposed portfolio and compare with the current optimum, keep the α which gives the greatest utility.

6. Repeat steps 2 to 5 a large number of times.

7. Move to next value of η of interest and repeat steps 2 to 6.

8. Repeat for all values of η of interest.

This method can be computationally intensive, however when compared to the MCMC schemes required to produce the simulated returns values this is insignificant.

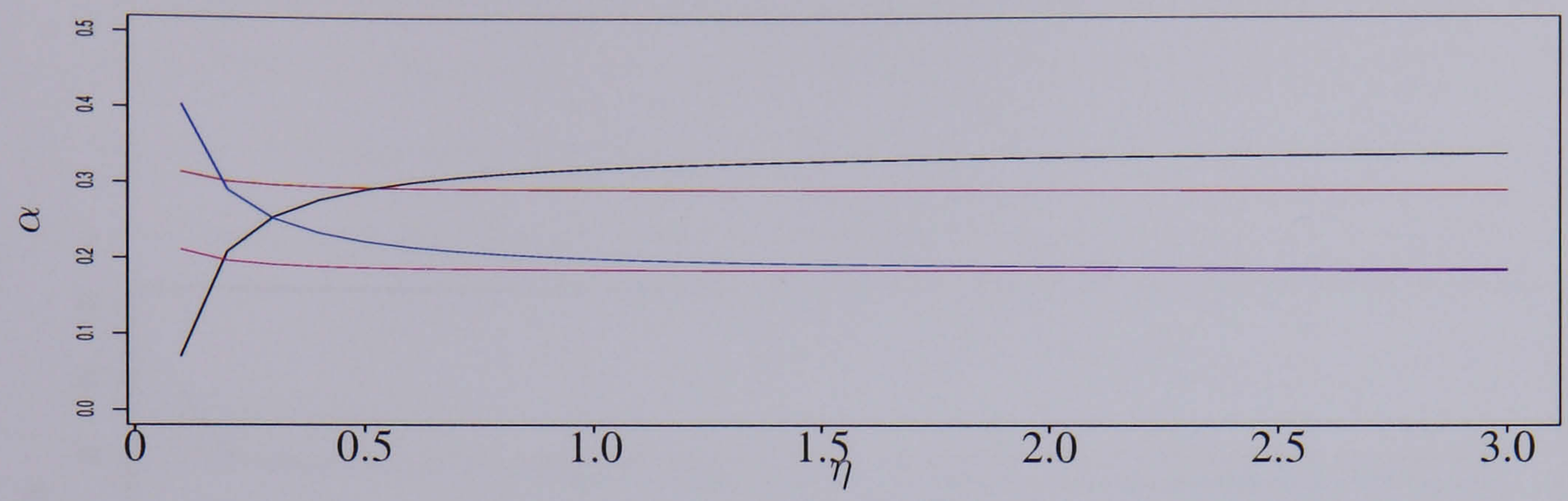


Figure 6.4: Plots of portfolio weights α for varying levels of risk preference η for a 10 step ahead forecast, by solving the constrained quadratic program.

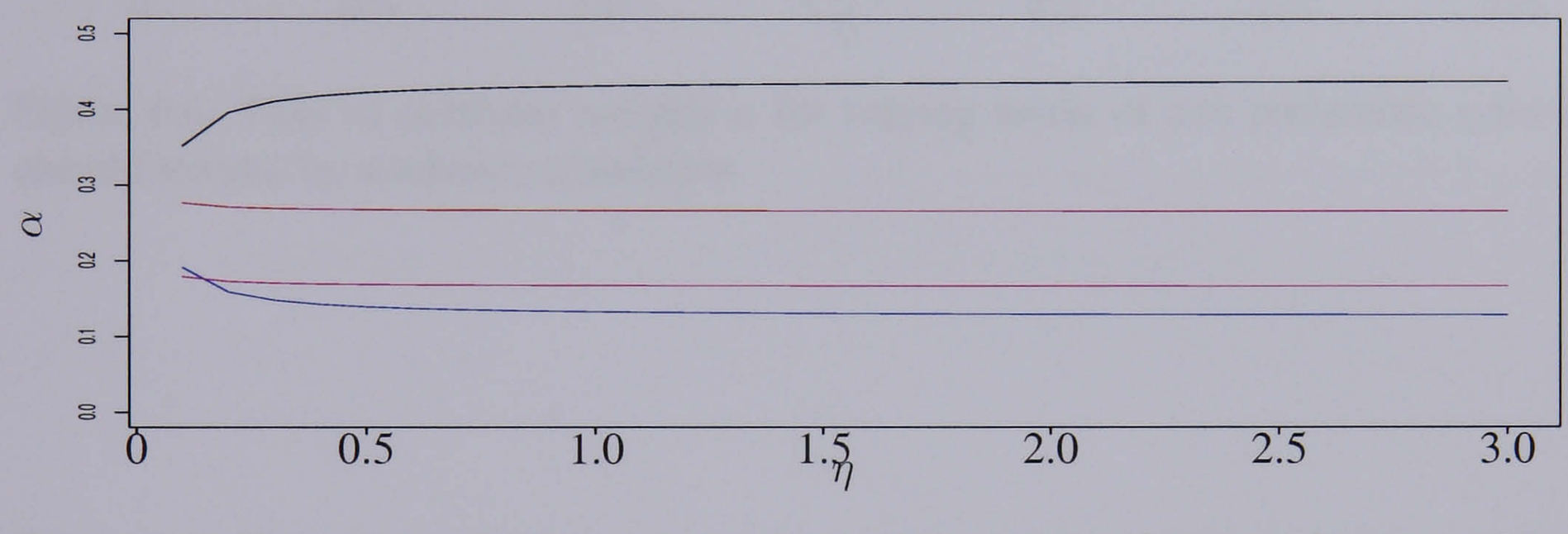


Figure 6.5: Plots of portfolio weights α for varying levels of risk preference η for a 50 step ahead forecast, by solving the constrained quadratic program.

6.4.6 Examples

As discussed above there are two ways of constraining the maximisation problem. In Section 6.4.4 we looked at constraining the Markowitz quadratic program. Using the summary statistics for the 10 and 50 step ahead forecasts from Section 3.2.2 we can run program the algorithm from Section 6.4.4 in **R**. Figure 6.4 and Figure 6.5 show the results of running this program based on the summary statistics for the two forecast distributions while preventing shorting. These two plots show radically different portfolios from those when shorting was allowed as well as the change in portfolio with time horizon. The change in portfolio with time horizon is to be expected, as we look further into the future we can be less certain of our predictions and hence the portfolio should reflect this.

If we use the full forecast distribution and again limit to prevent shorting as described in Section 6.4.5 using the **sather** program *stochoptim.sa* a copy of which can be found in Appendix A

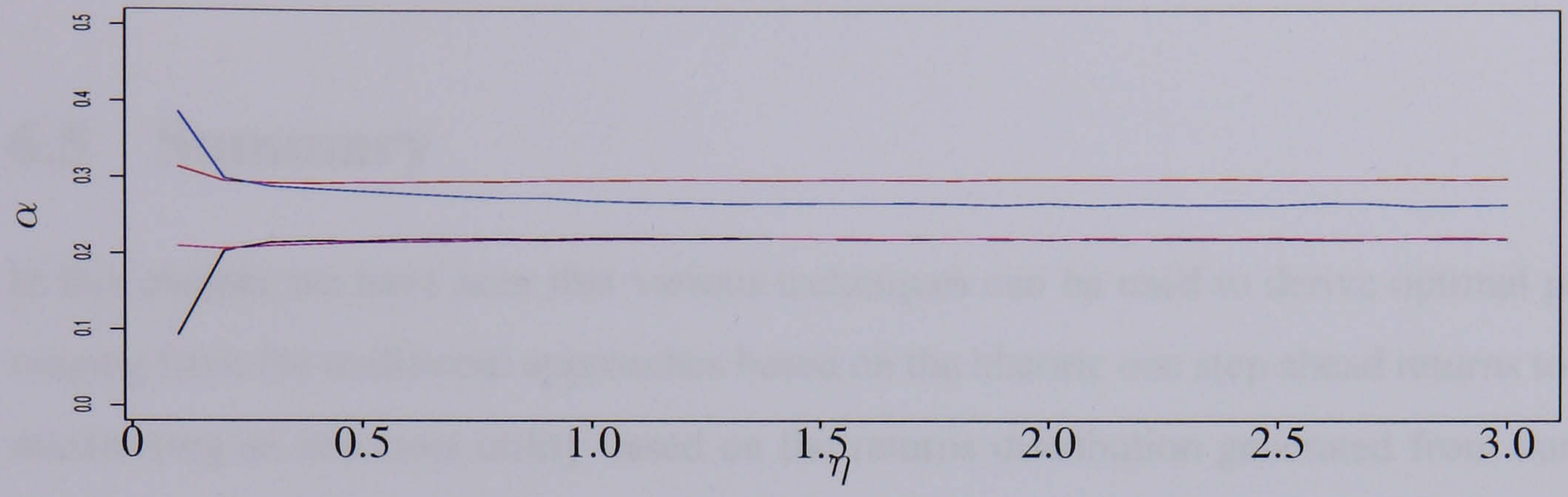


Figure 6.6: Plots of portfolio weights α for varying levels of risk preference η for a 10 step ahead forecast, by stochastic simulation.

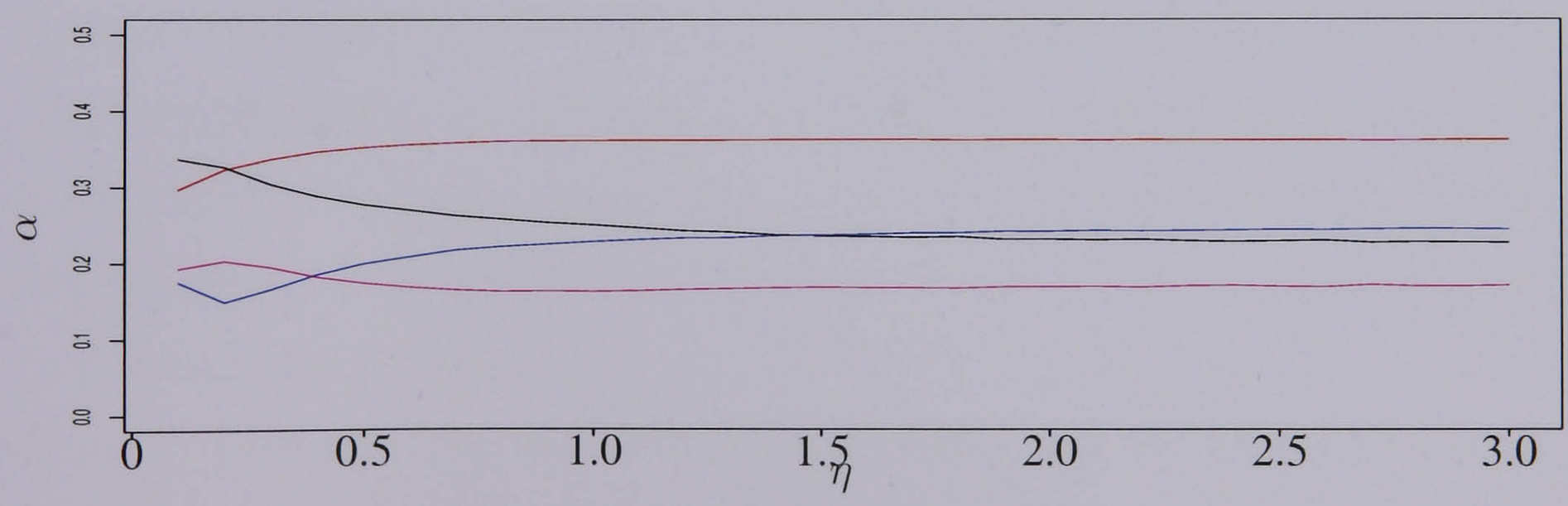


Figure 6.7: Plots of portfolio weights α for varying levels of risk preference η for a 50 step ahead forecast, by stochastic simulation.

we get a different portfolio allocation to that from using only the summary statistics. Figure 6.6 and Figure 6.7 show the results of using this stochastic optimisation techniques. What we see are radically different portfolios for the both the 10 and 50 step ahead forecasts when using the whole of the returns distribution to calculate the utility rather than just the summary statistics.

6.5 Summary

In this chapter we have seen that various techniques can be used to derive optimal portfolios, ranging from the traditional approaches based on the historic one step ahead returns to methods maximising an investors utility based on the returns distribution generated from our MCMC schemes. These portfolios can be radically different. We have not included any measure of portfolio performance and it might be an interesting exercise to compare the future performance of portfolios chosen and draw concisions as to which method produced the "best" portfolios.

Chapter 7

Case Studies

In the previous chapters of this thesis we have looked at methods whereby we can make inference about the future values of log share prices or the returns on a series of shares and use these predictions in the portfolio selection process. In Chapter 3 and Chapter 4 we have illustrated models and methods of making inference on these models by means of examples using simulated data. In this chapter we will apply our models and inference techniques to some real data.

7.1 The Data

The data that we will use in this chapter consists of 6 shares from the S and P 500 representing a diversified portfolio. The shares are American Express, Boeing Corporation, Disney, Federal Express, I.B.M and Morgan Stanley. These were chosen as they are well established companies with complete trading histories over the period of interest with no unusual activities such as share splits or rights issues.

Figure 7.1 shows the time series plots of these shares. The data are the closing prices for the 1009 consecutive trading days between the 2^{nd} January 1997 and the 29^{th} December 2000. We will actually be applying our models to the log share price which is shown in Figure 7.2. We chose the log share price as it is the log share price which exhibits linear Gaussian dynamics as discussed in Chapter 2 and Chapter 3.

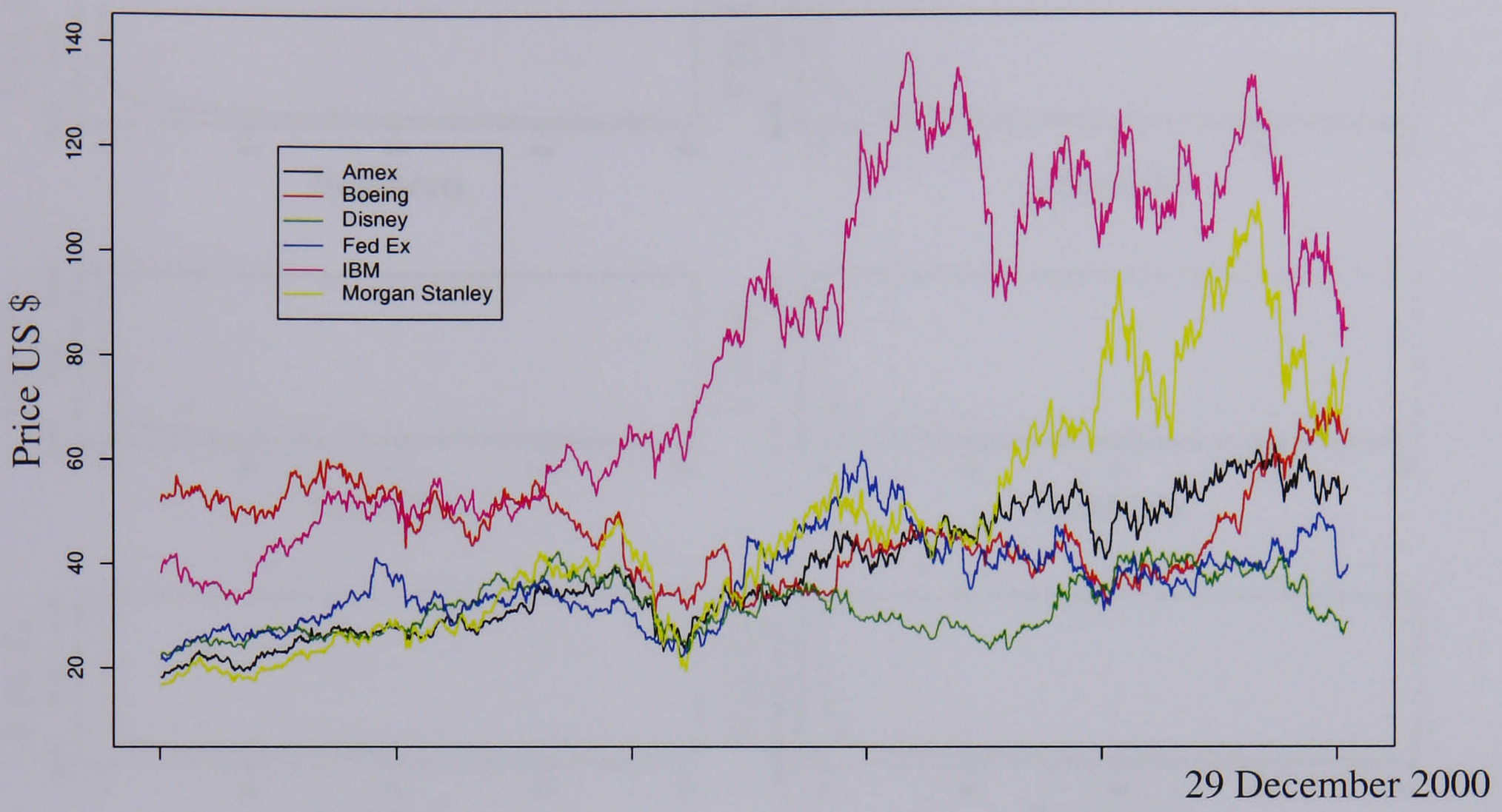


Figure 7.1: Times series plots of the 6 shares.

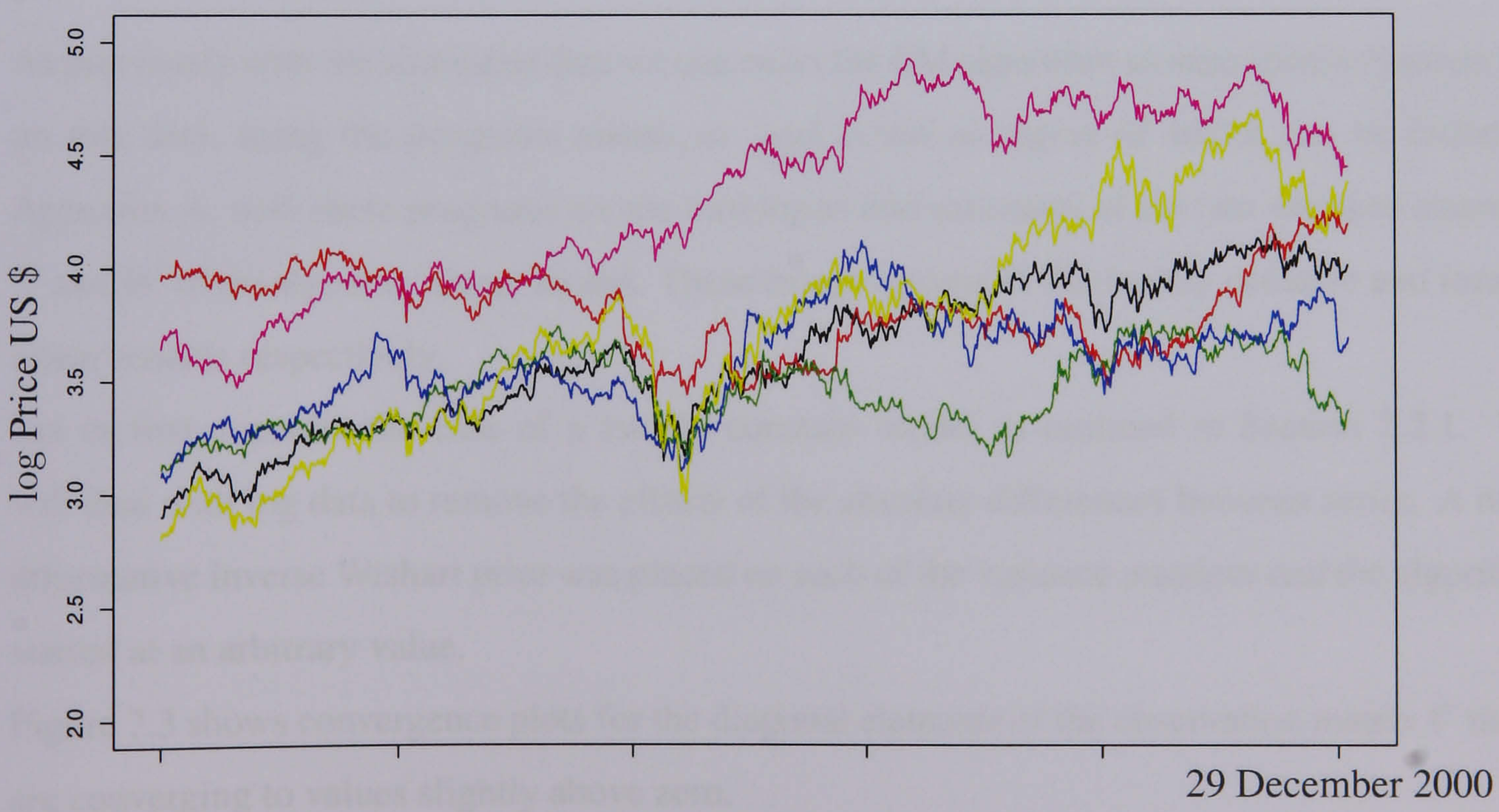


Figure 7.2: Times series plots of the 6 shares.

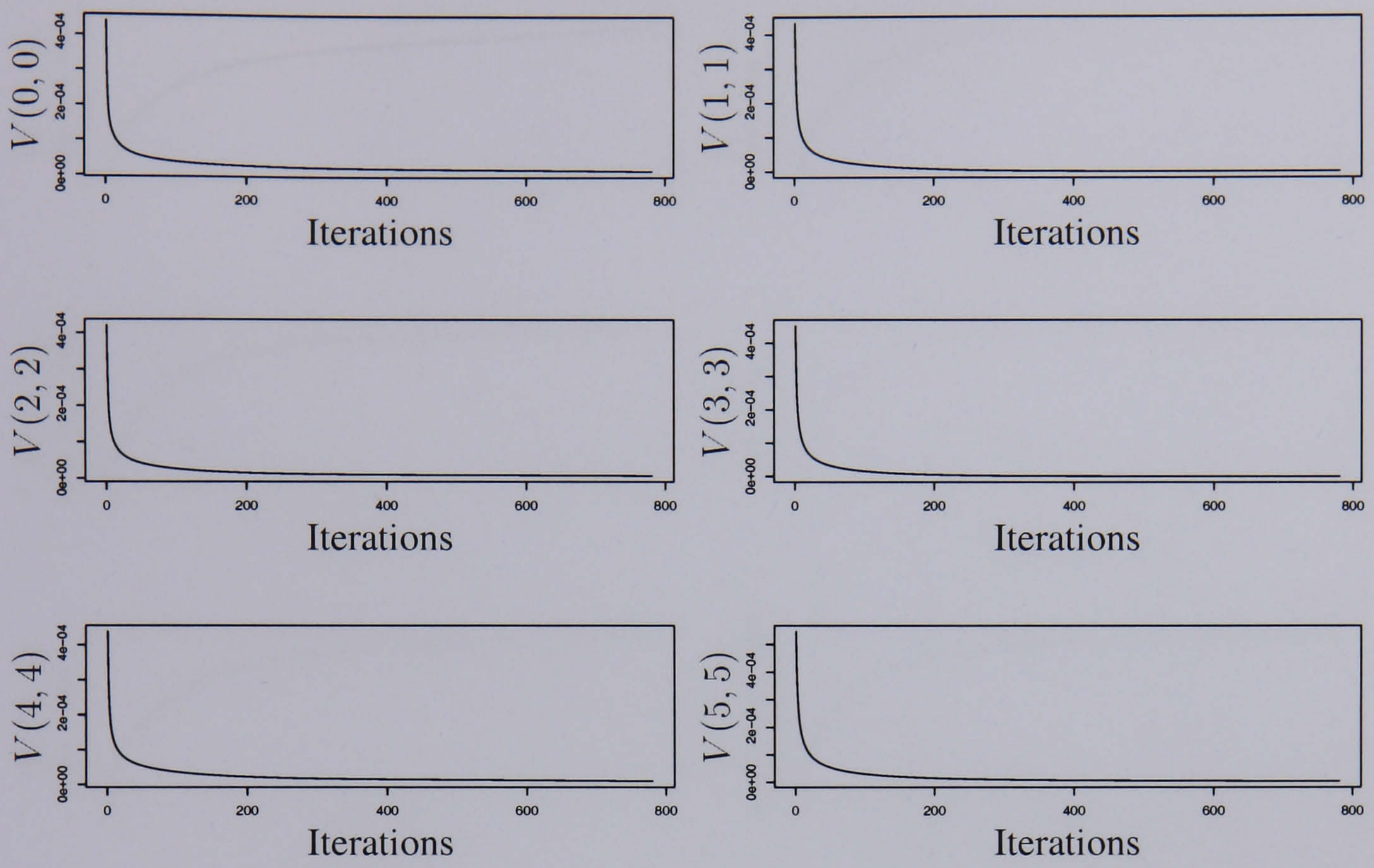


Figure 7.3: Convergence plots for selected elements of the observation variance matrix of the multivariate locally constant EM Algorithm, on the real data.

7.2 EM Algorithm

As previously with the simulated data we can enact the EM algorithm as described in Section 3.1 on this data, using the programs *memlc.sa* and *memll.sa* copies of which can be found in Appendix A, with these programs we are looking to find estimates of the two variance matrices V and W of the dynamic linear model. These two programs fit the locally constant and locally linear models respectively.

Let us first consider the case of a locally constant model as outlined in Section 2.2.1. We will deal with log data to remove the effects of the absolute differences between series. A non-informative Inverse Wishart prior was placed on each of the variance matrices and the algorithm started at an arbitrary value.

Figure 7.3 shows convergence plots for the diagonal elements of the observation matrix V these are converging to values slightly above zero.

Figure 7.4 shows convergence plots for the diagonal elements of the state evolution matrix W these are converging to values in the range 0.00052 to 0.001, slightly higher than for the

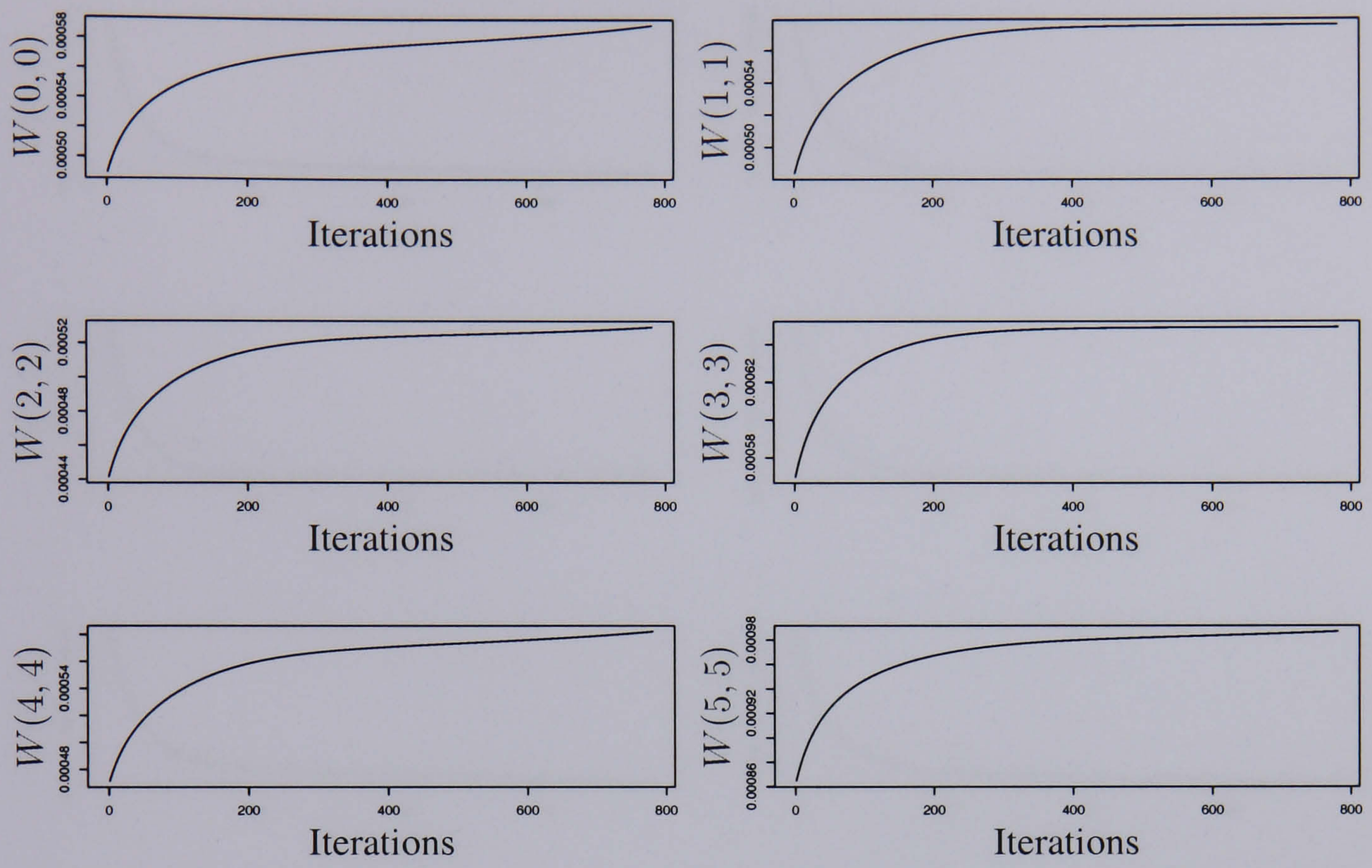


Figure 7.4: Convergence plots for selected elements of the state variance matrix of the multivariate locally constant EM Algorithm, on the real data.

observation variance values. This means that in this case the state has a greater variance than the observation variance.

We can also run the Expectation Maximisation algorithm on this data assuming a locally linear dynamic linear model structure as outlined in Section 2.2.2. Once again non-informative inverse Wishart priors were set and the algorithm started at some arbitrary values.

Figure 7.5, Figure 7.6 and Figure 7.7 show the convergence plots for selected elements of the variance matrices. This show similar results as with the locally constant model, with the elements of V converging at values close to zero. The values for W appear to be converging to values slightly lower than for the locally constant model, with the difference perhaps being explained by the trend component whose variance Z appears to be converging to very small values. It is interesting to note that the algorithm took longer to converge for the locally linear model than for the locally constant model.

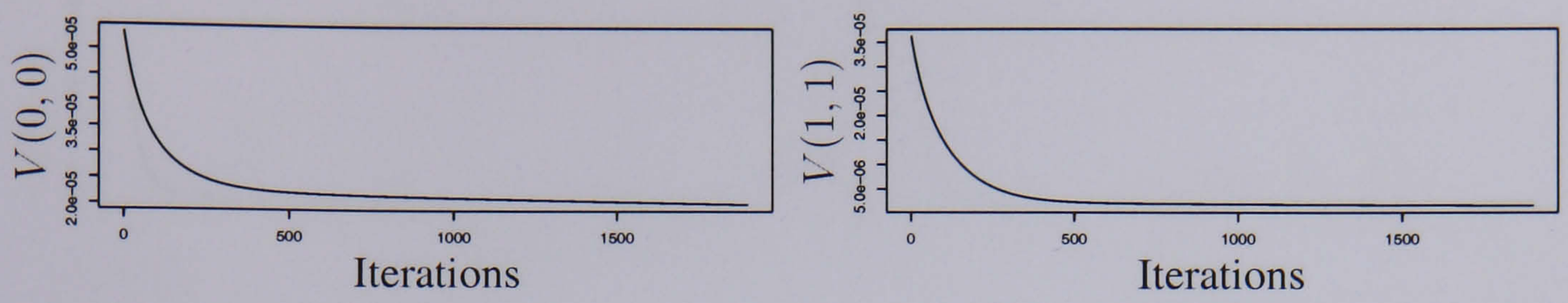


Figure 7.5: Convergence plots for selected elements of the observation variance matrix of the multivariate locally linear EM Algorithm, on the real data.

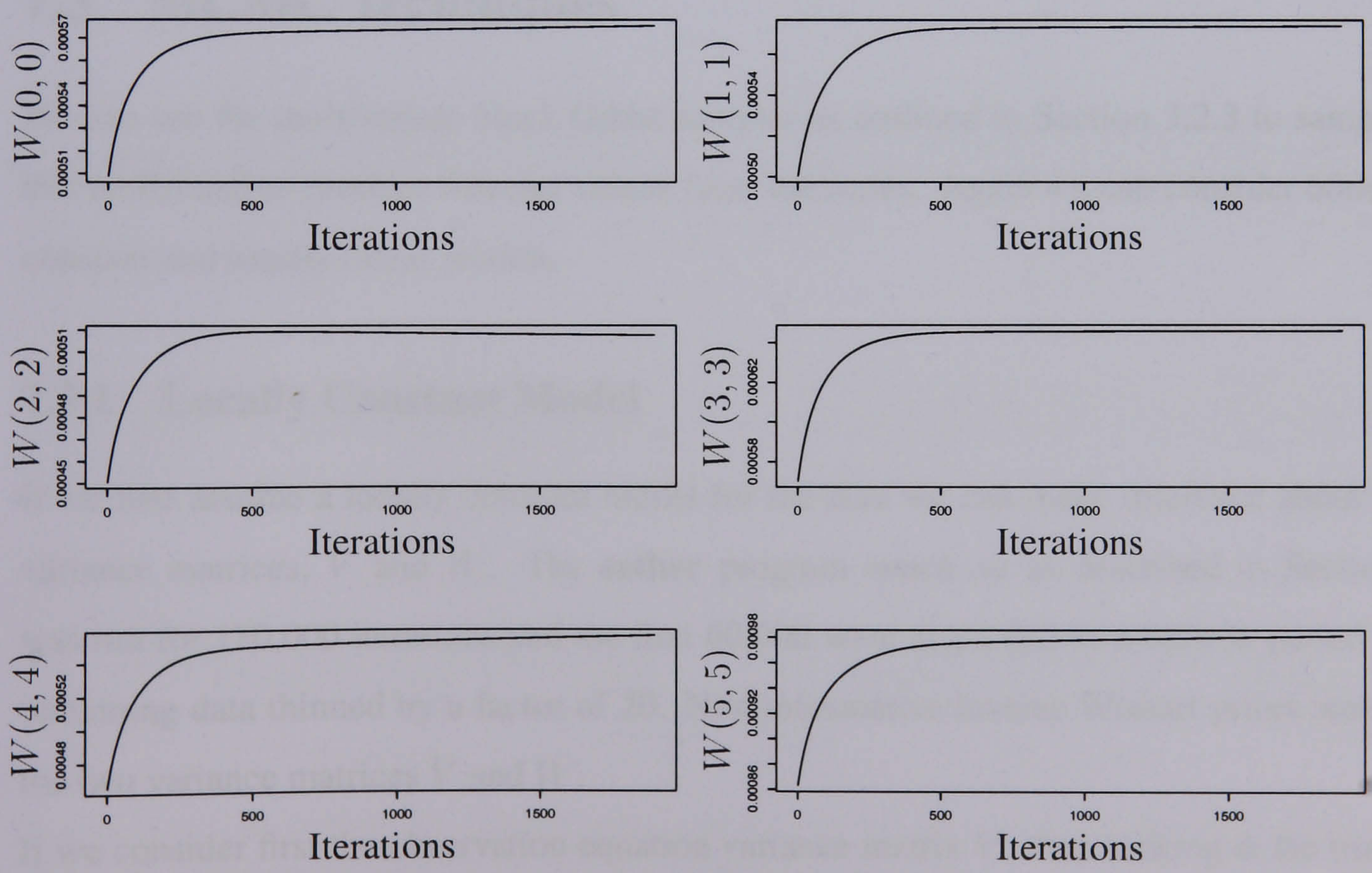


Figure 7.6: Convergence plots for selected elements of the state variance matrix of the multivariate locally linear EM Algorithm, on the real data.

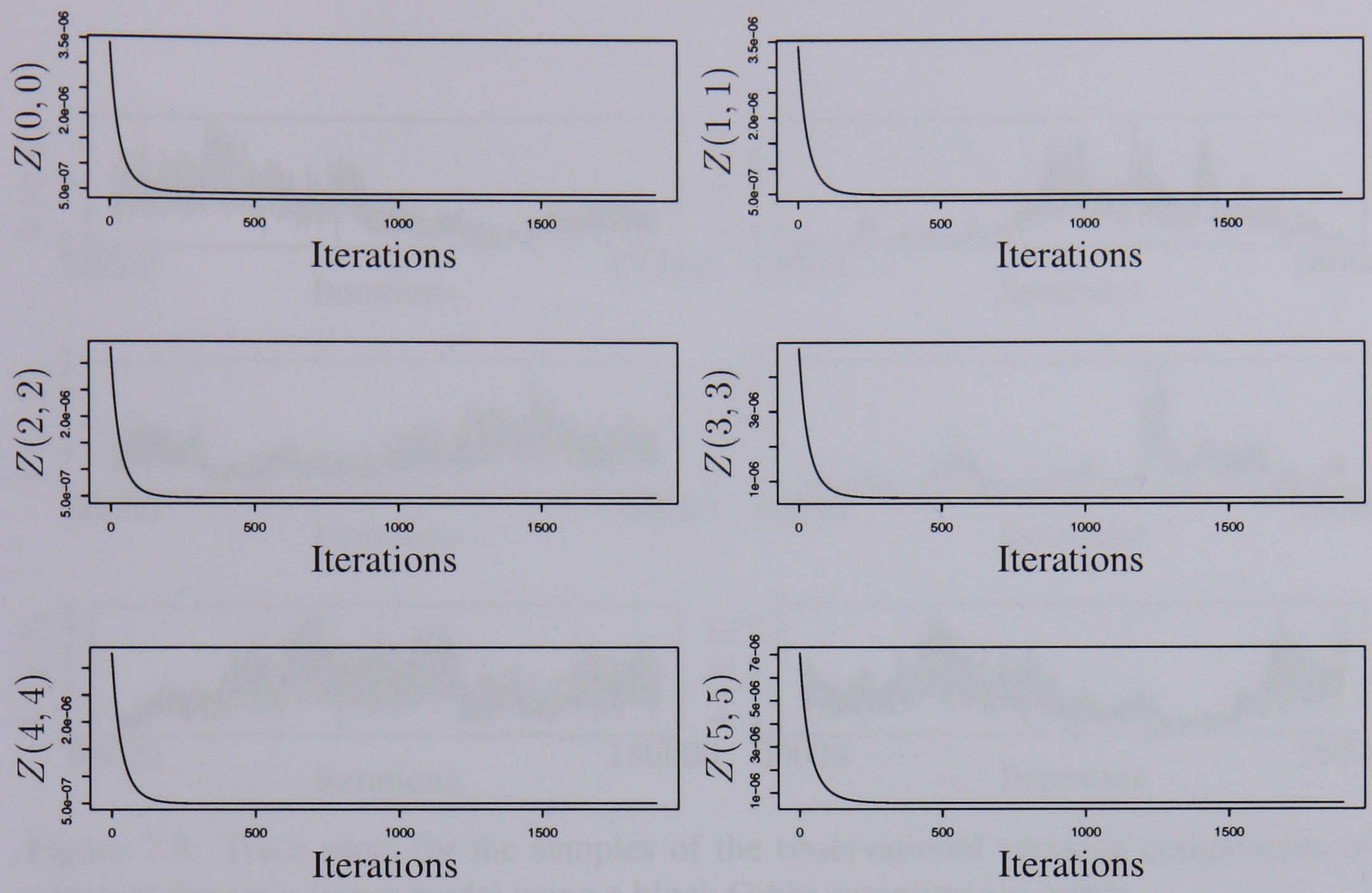


Figure 7.7: Convergence plots for selected elements of the state variance matrix of the multivariate locally linear EM Algorithm, on the real data.

7.3 MCMC Techniques

We can use the multivariate block Gibbs sampler as outlined in Section 3.2.3 to sample from this model and to produce forecast values from the series. Again we can consider both locally constant and locally linear models.

7.3.1 Locally Constant Model

If we first assume a locally constant model for the data we can make inference about the two variance matrices, V and W. The **sather** program *mmclc.sa* as described in Section 3.2.3 was run for 180,000 iterations and the first 60,000 were discarded as a burn-in period and the remaining data thinned by a factor of 20. Non-informative Inverse Wishart priors were set on the two variance matrices V and W.

If we consider first the observation equation variance matrix V, then looking at the trace plots Figure 7.8 these suggest that sampler has not converged and this is supported by the autocorrela-

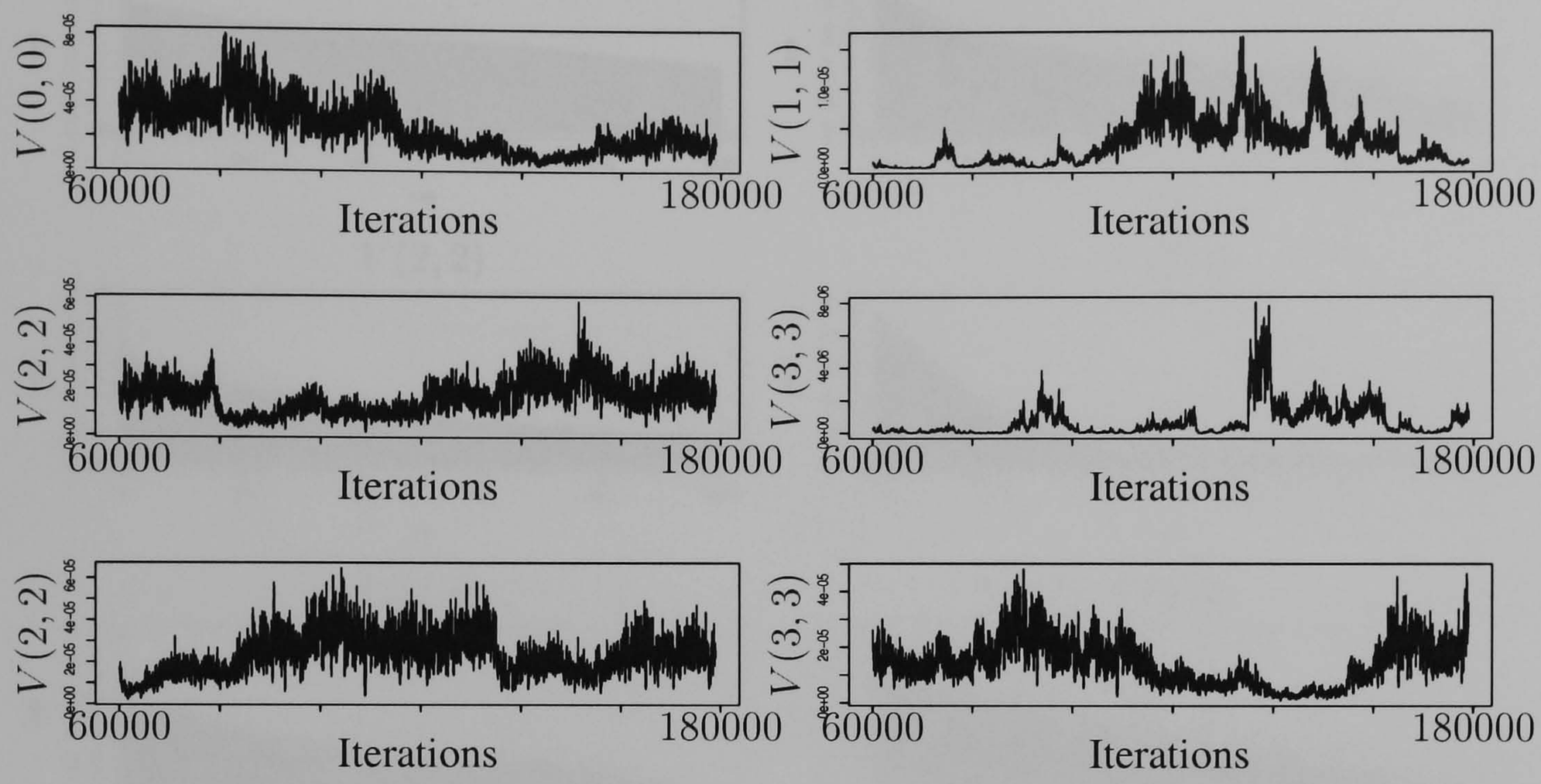


Figure 7.8: Trace plots for the samples of the observational variance components of locally constant dynamic linear model using a block Gibbs sampling algorithm

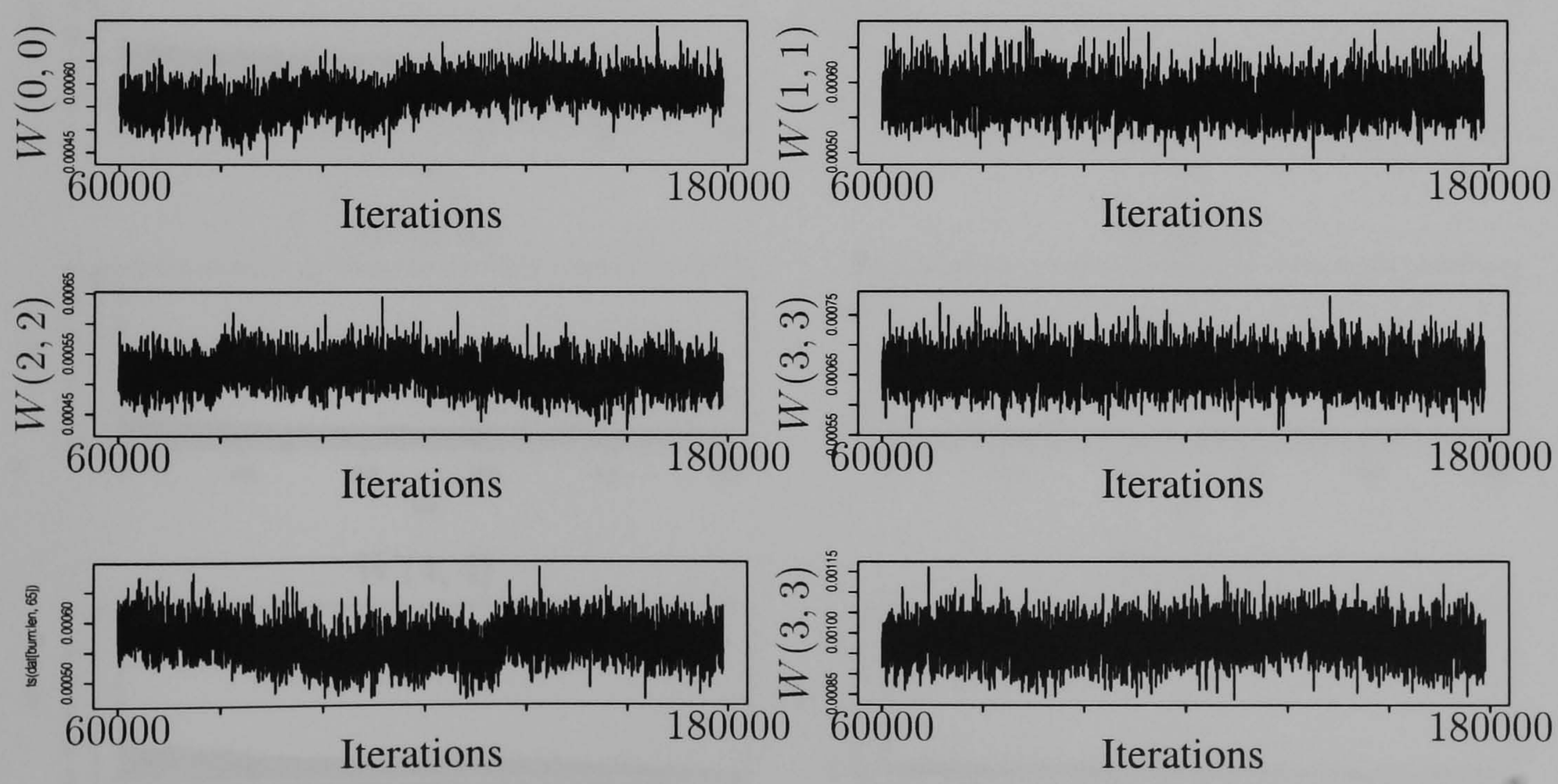


Figure 7.9: Trace plots for the samples of the state variance components of locally constant dynamic linear model using a block Gibbs sampling algorithm

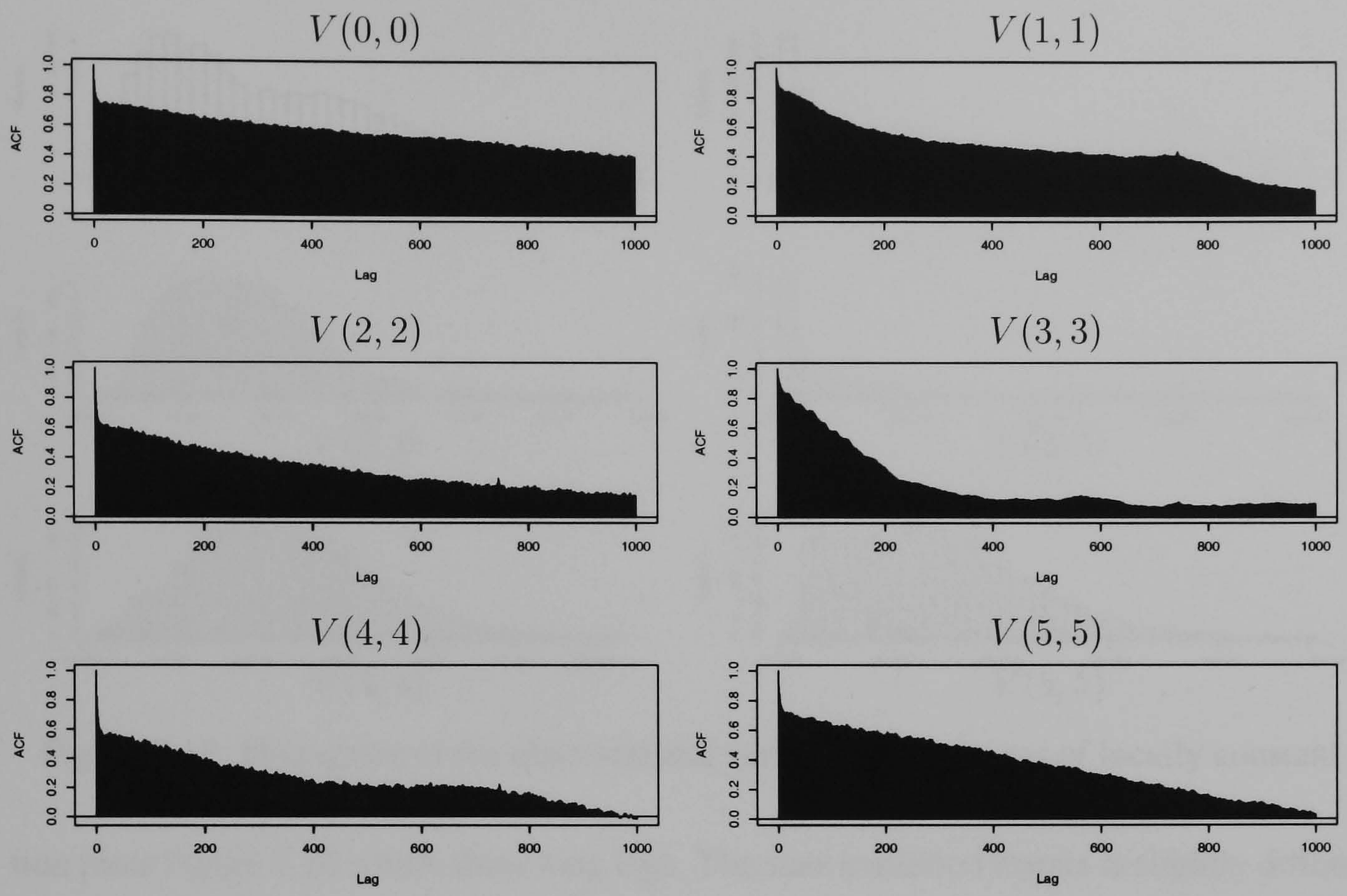


Figure 7.10: Autocorrelation plots for selected elements of the observation variance matrix V.

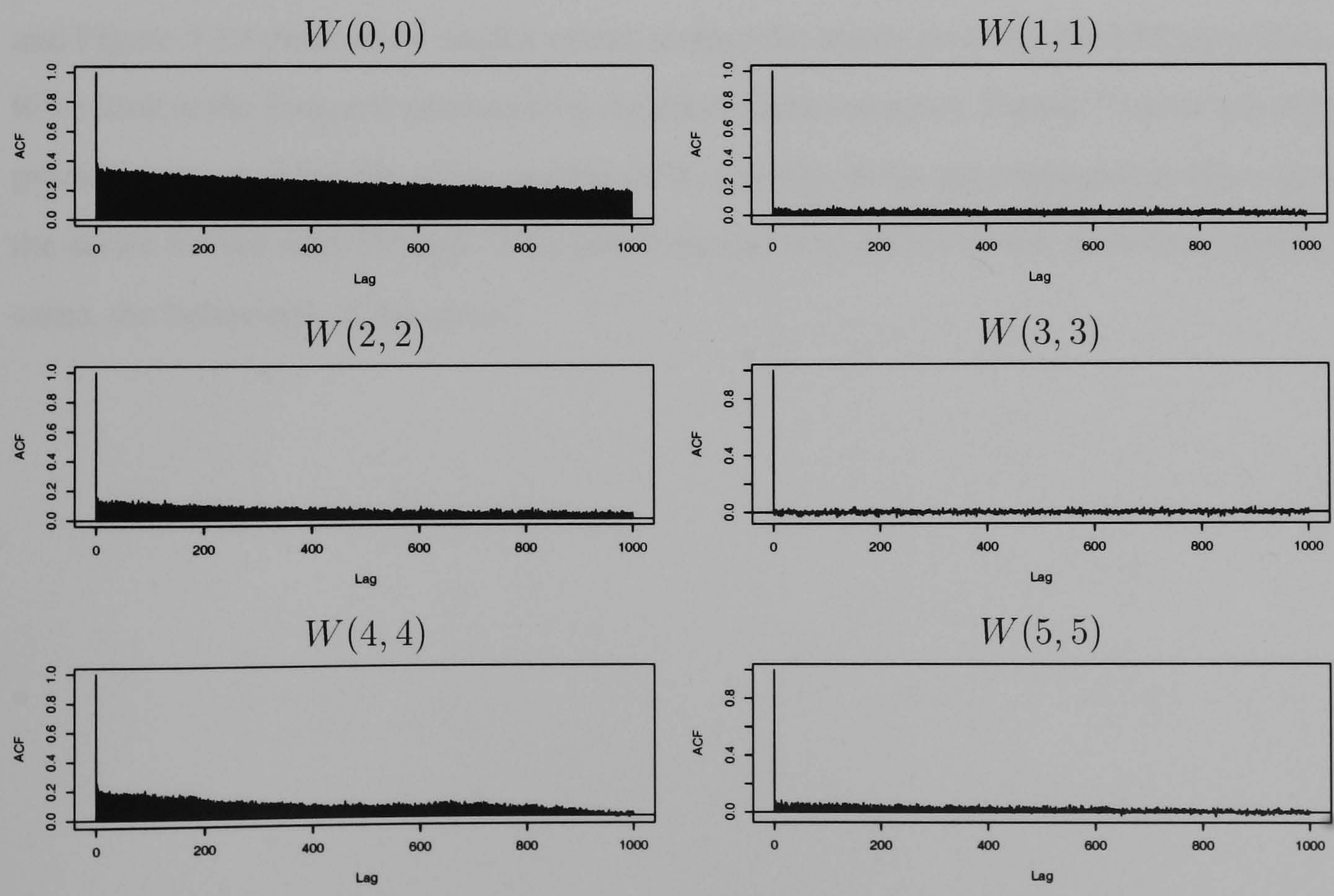


Figure 7.11: Autocorrelation plots for selected elements of the state evolution variance matrix W.

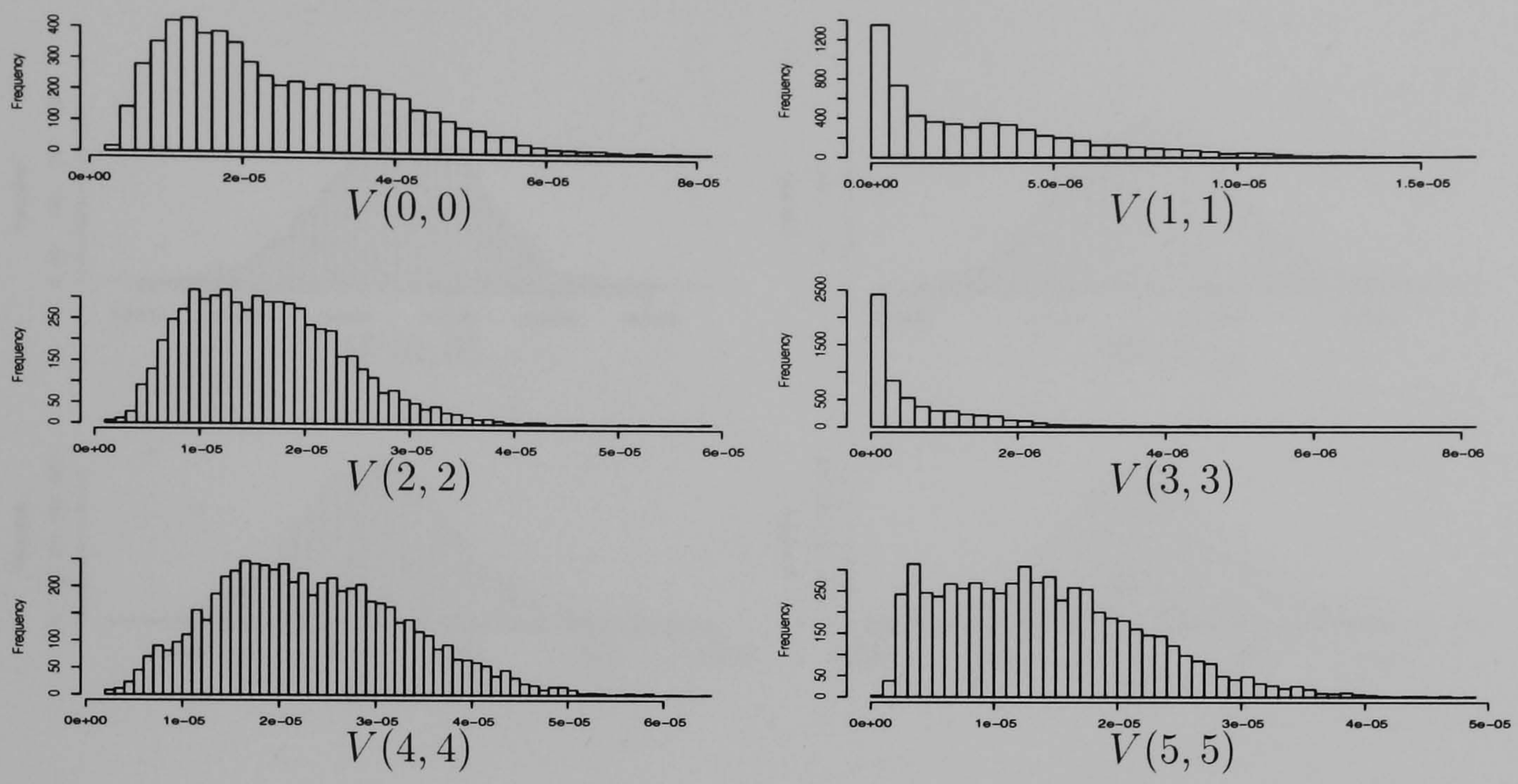


Figure 7.12: Histogram of the observational variance components of locally constant DLM

tion plots Figure 7.10 which show long lags. The state evolution matrix is slightly different with both the trace plots Figure 7.9 and the autocorrelation plots Figure 7.11 suggesting convergence of the sampler. If we look at histograms of the sampled values for both V and W, Figure 7.12 and Figure 7.13 these show modal values around the levels given by the EM algorithm.

If we look at the forecasts produced by the block Gibbs sampler, Figure 7.14 we can see the 95% predictive interval for the series and the 50% quantile, these are compared to the true values of the series for the next 50 days. This plot illustrates the ability of the model to capture, in most cases, the behaviour of the series.

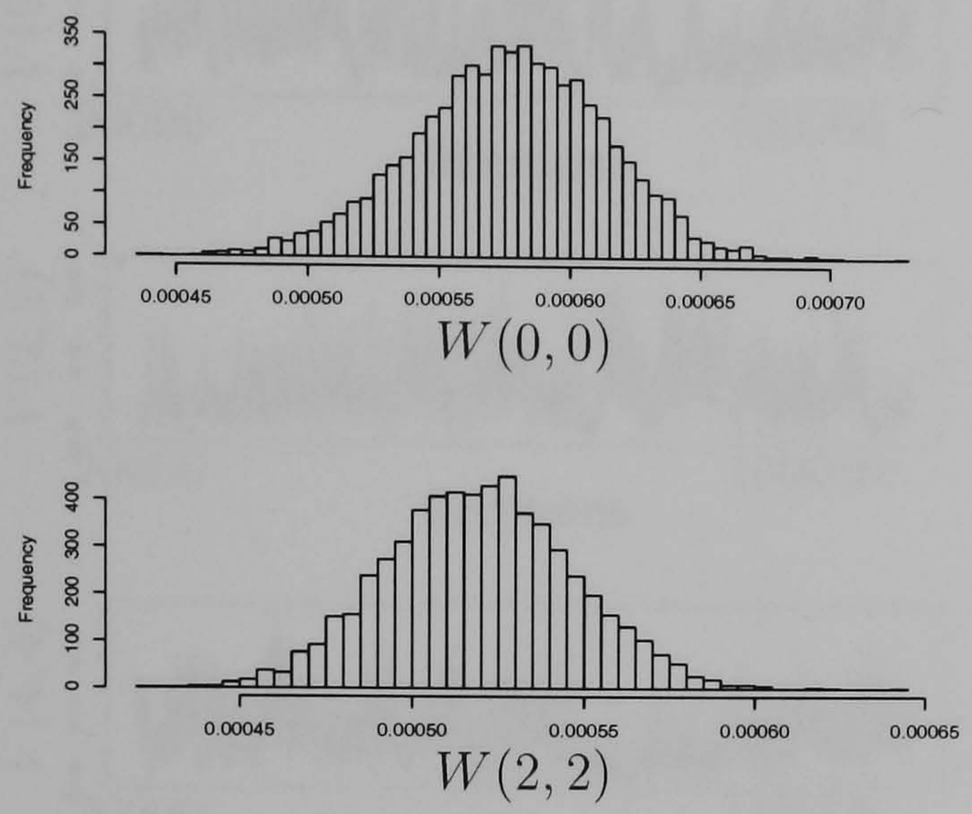


Figure 7.13: Histogram of the state evolution variance components of locally constant DLM

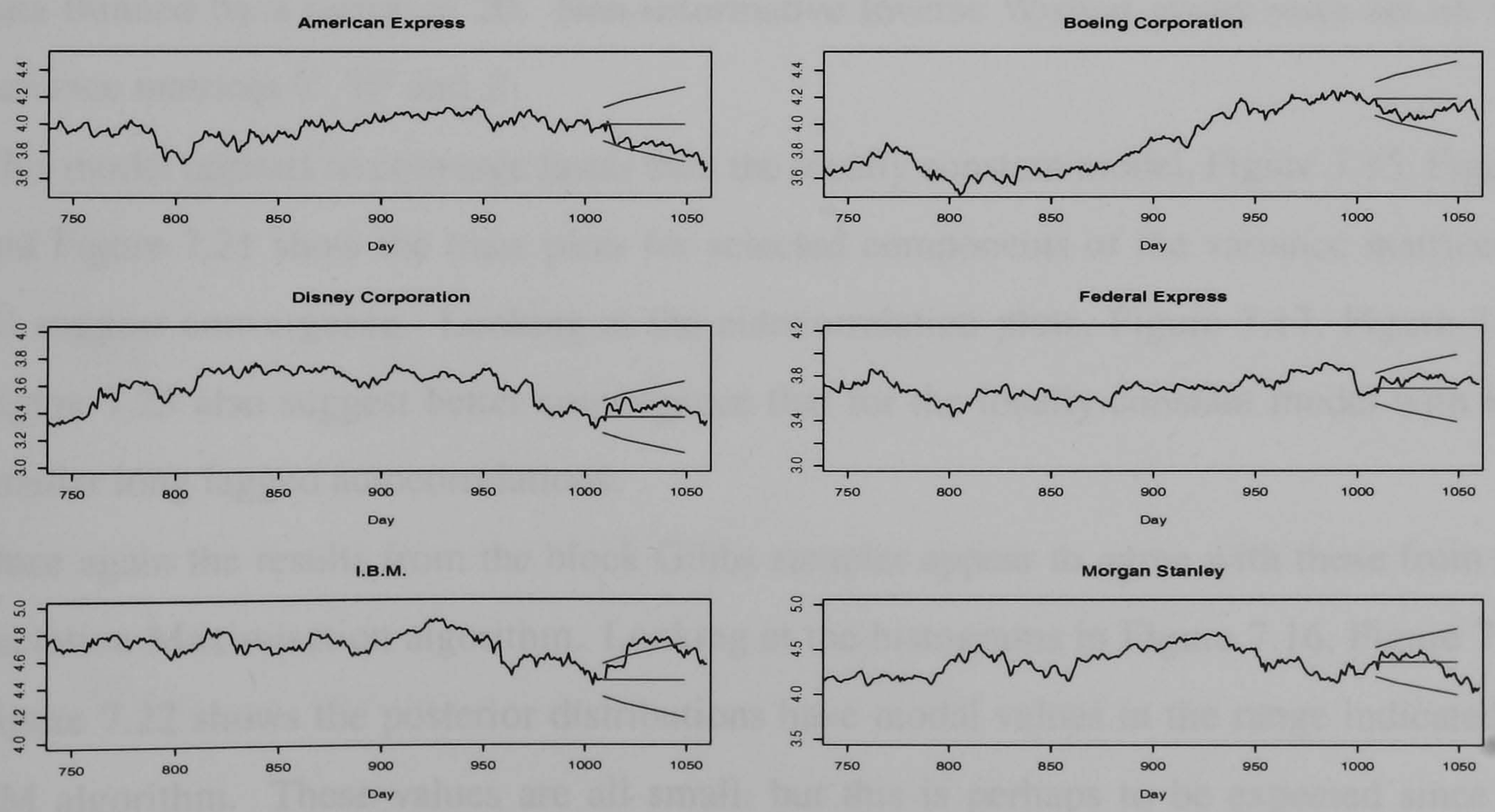


Figure 7.14: Predictive intervals for the 50 step ahead forecasts

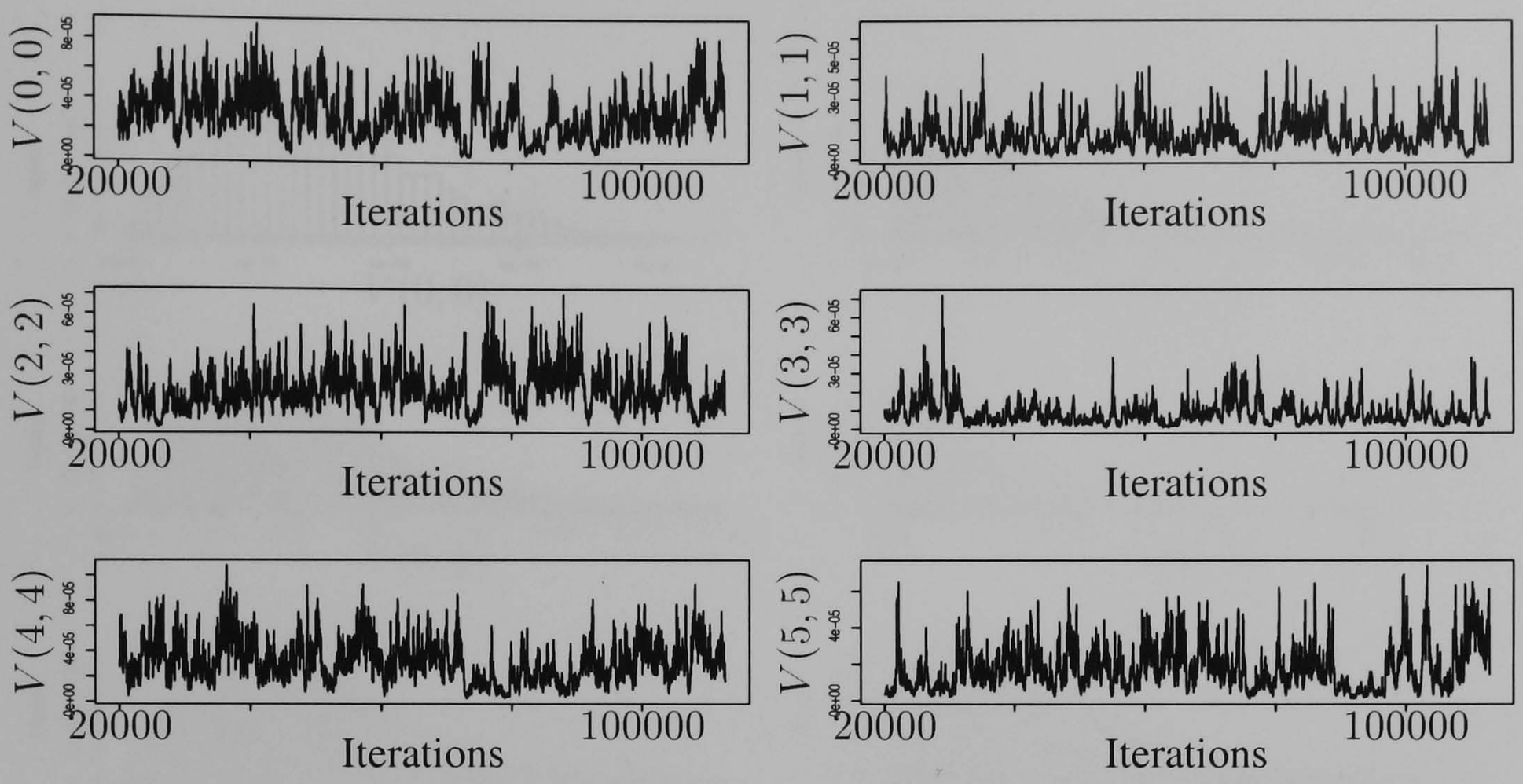


Figure 7.15: Trace plots for the samples of the observational variance components of locally linear dynamic linear model using a block Gibbs sampling algorithm

7.3.2 Locally Linear Model

If we now assume a locally linear model for the data we can make inference about the variance matrices, V W and Z. The **sather** program *mmcll.sa* as described in Section 3.2.3 was run for 120,000 iterations and the first 20,000 were discarded as a burn-in period and the remaining data thinned by a factor of 20. Non-informative Inverse Wishart priors were set on the thre variance matrices V, W and Z.

This model appears to converge faster than the locally constant model, Figure 7.15, Figure 7.18 and Figure 7.21 show the trace plots for selected components of the variance matrices, these all suggest convergence. Looking at the autocorrelation plots, Figure 7.17, Figure 7.20 and Figure 7.23 also suggest better convergence that for the locally constant model with less and smaller long lagged autocorrelations.

Once again the results from the block Gibbs sampler appear to agree with those from the Expectation Maximisation algorithm. Looking at the histograms in Figure 7.16, Figure 7.19 and Figure 7.22 shows the posterior distributions have modal values in the range indicated by the EM algorithm. These values are all small, but this is perhaps to be expected since we are working on logged data.

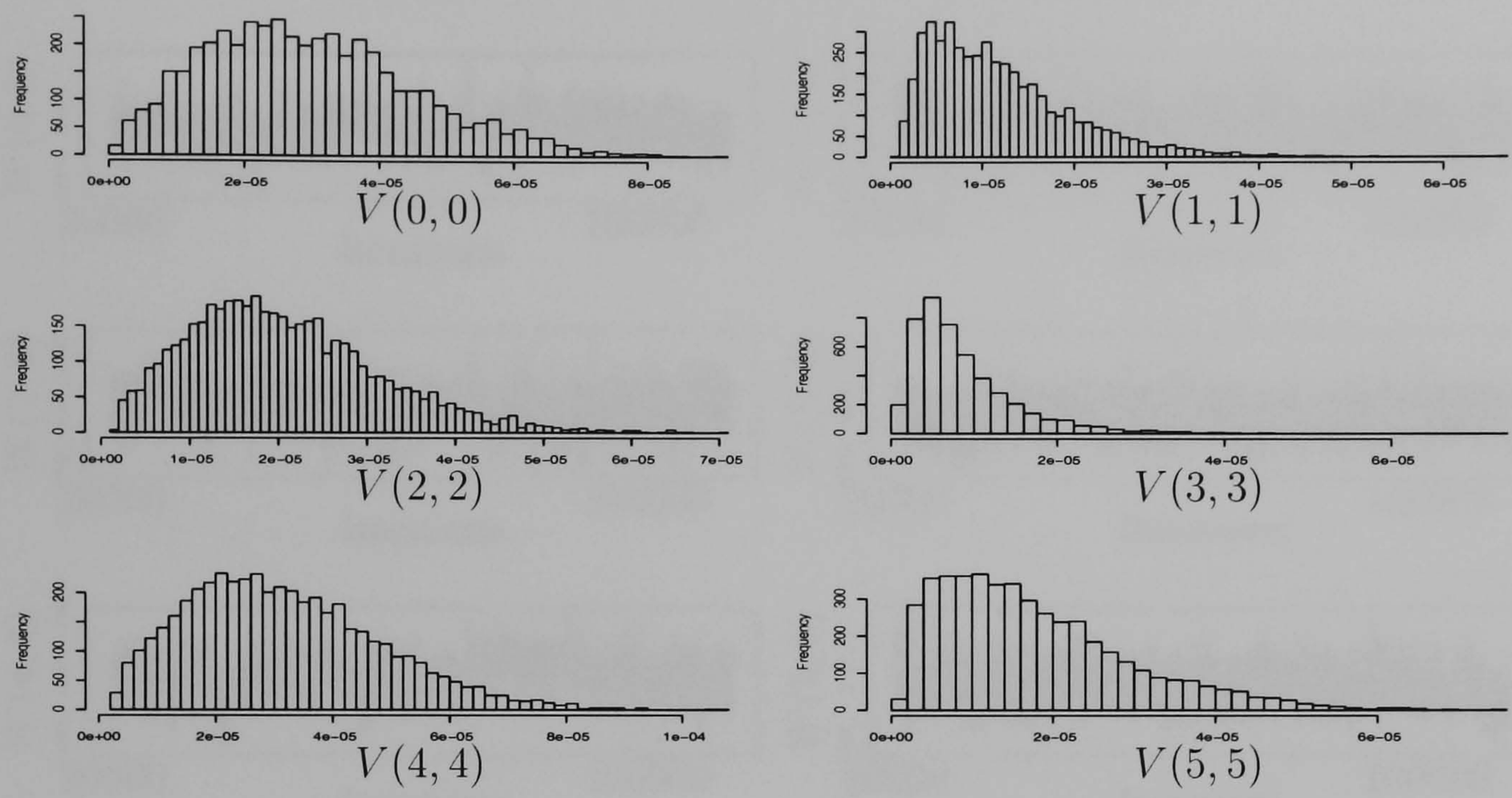


Figure 7.16: Histogram of the observational variance components of locally linear DLM

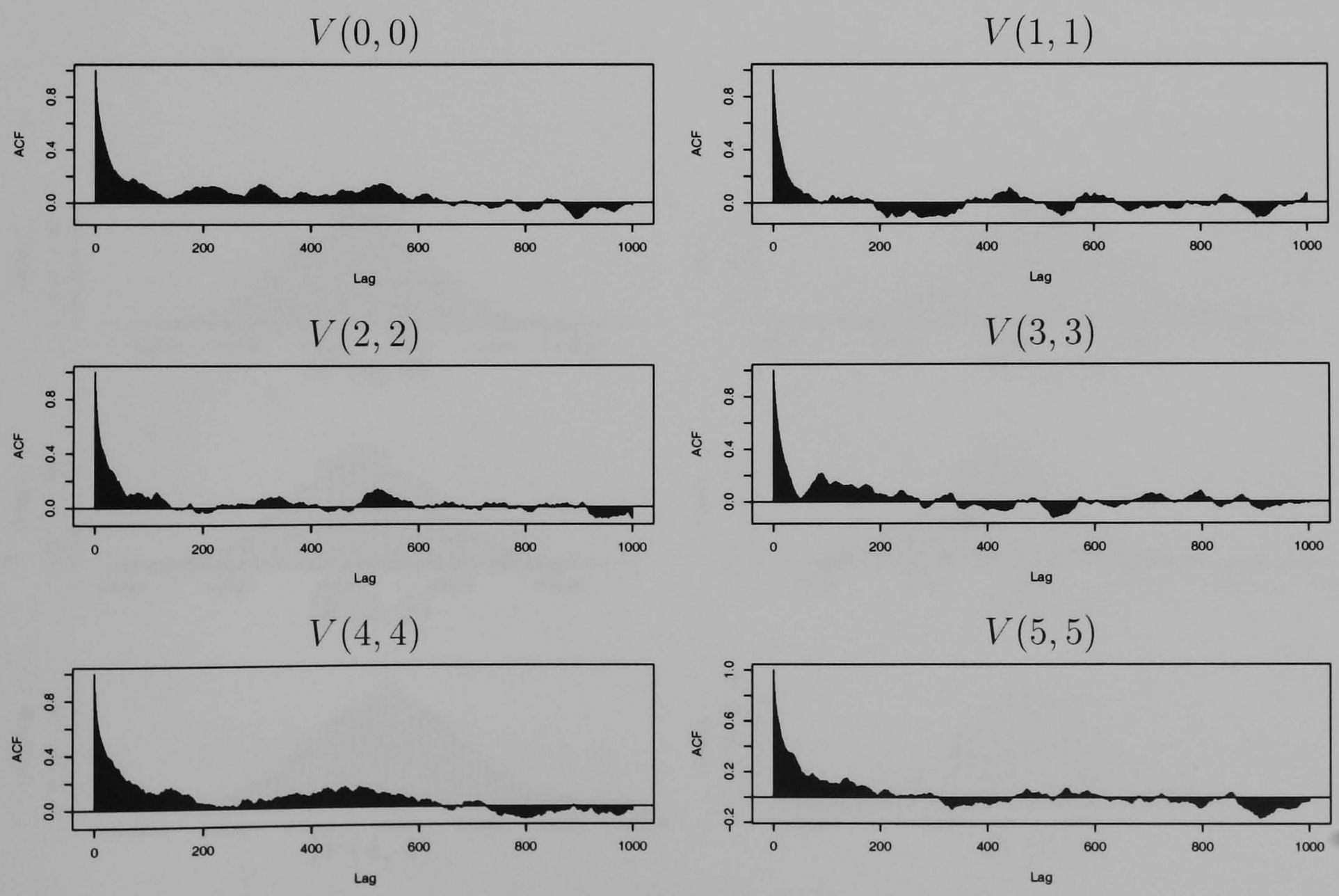


Figure 7.17: Autocorrelation plots for selected elements of the observation variance matrix V.

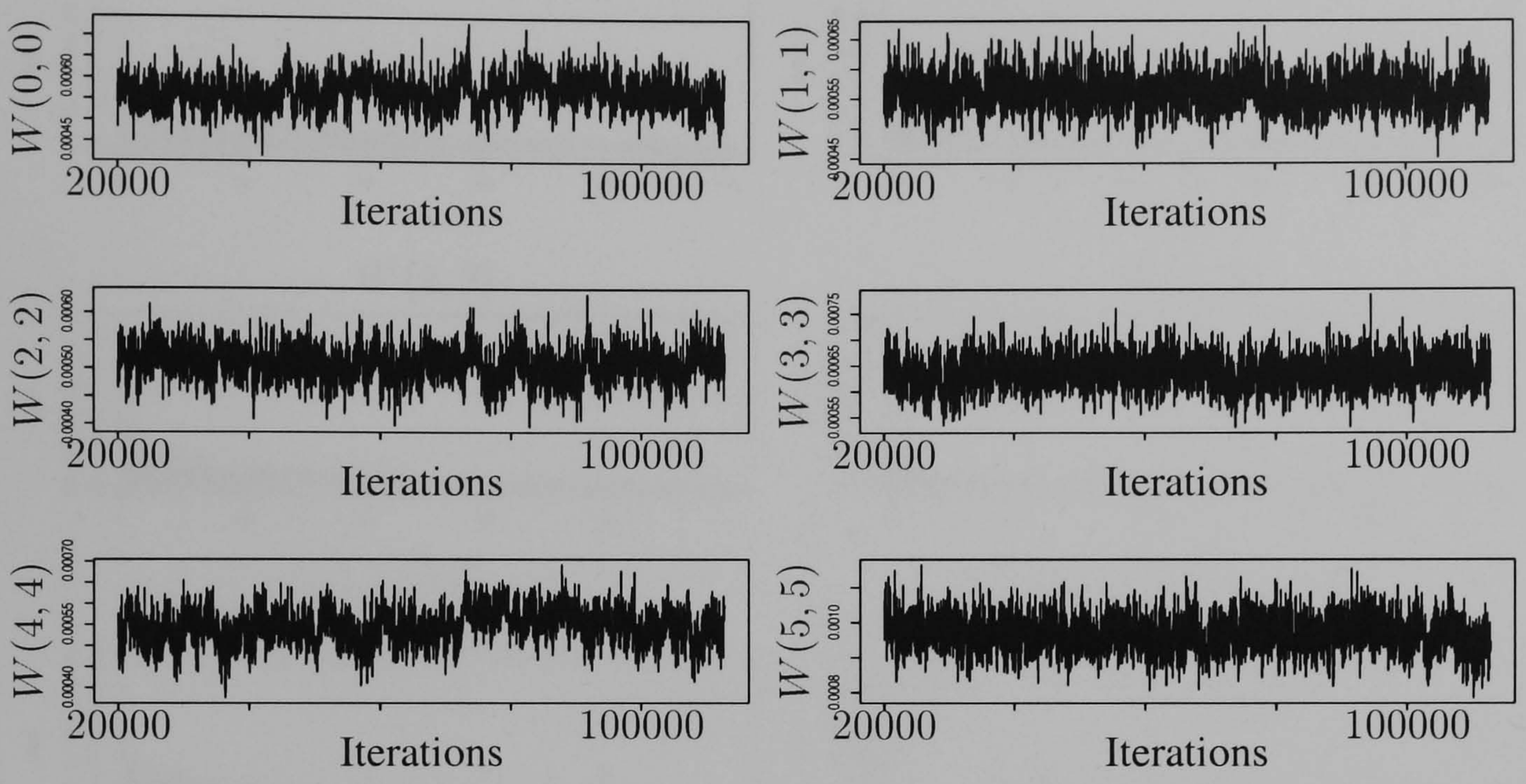


Figure 7.18: Trace plots for the samples of the state variance components of locally linear dynamic linear model using a block Gibbs sampling algorithm

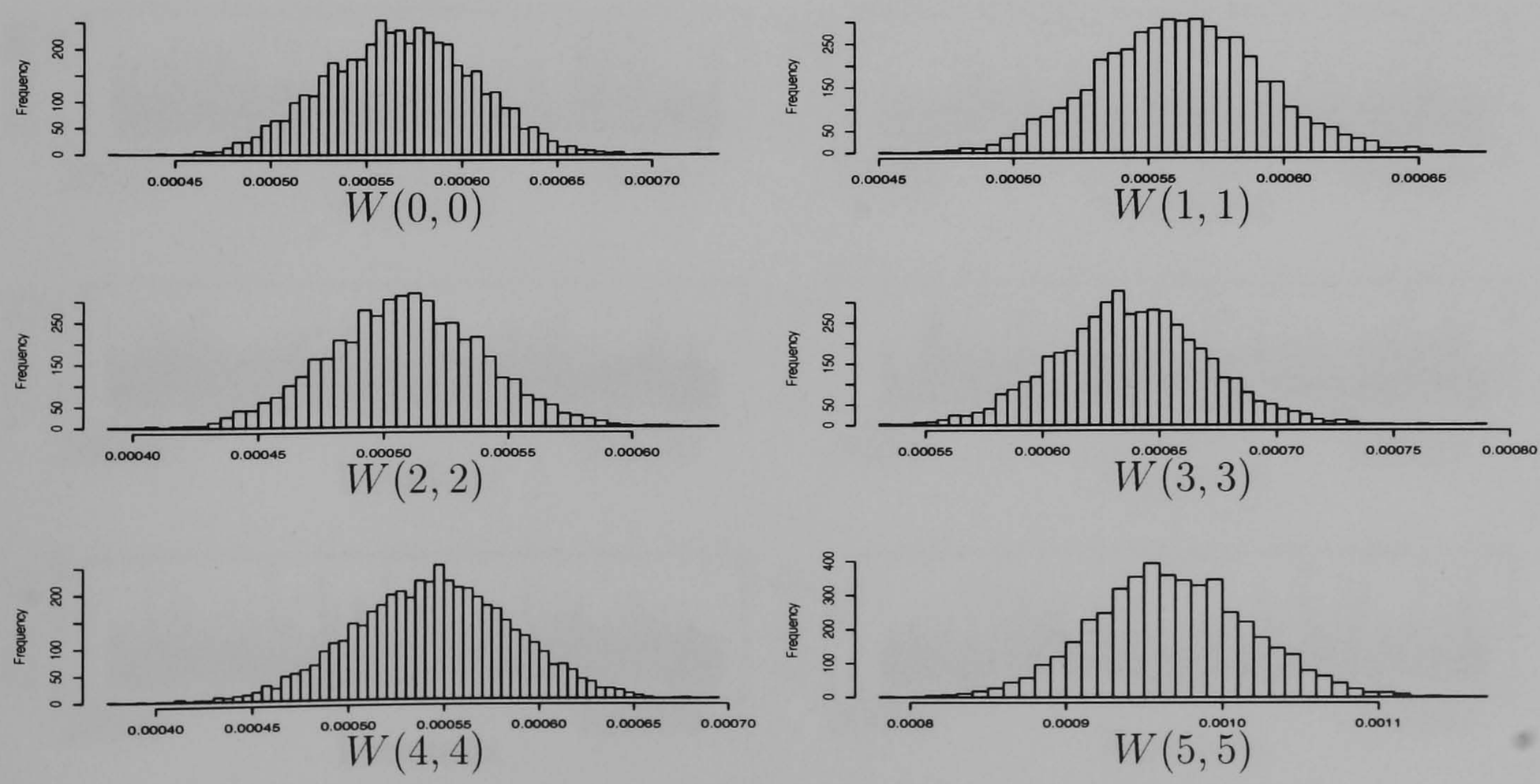


Figure 7.19: Histogram of the state variance components of locally linear DLM

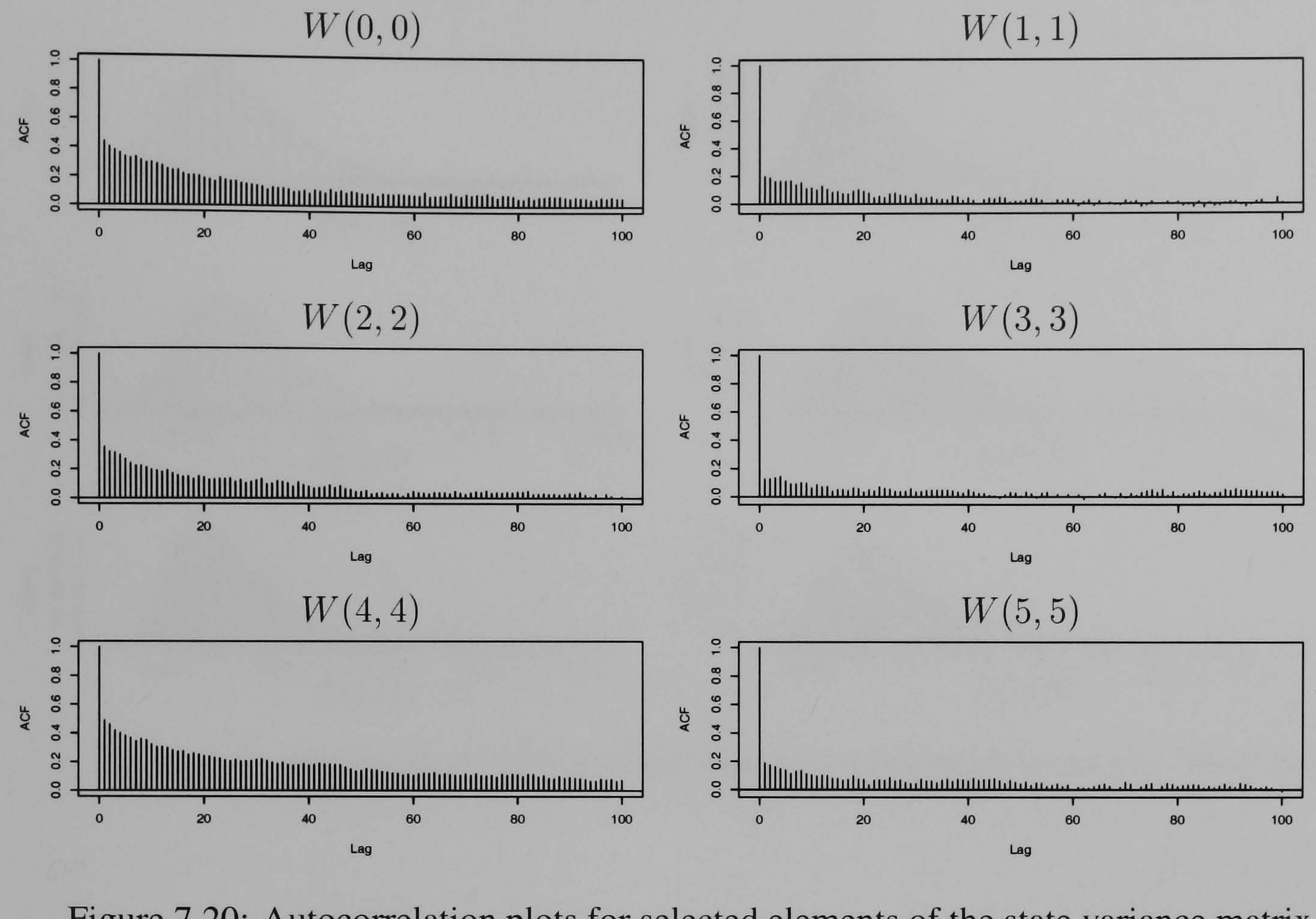


Figure 7.20: Autocorrelation plots for selected elements of the state variance matrix W.

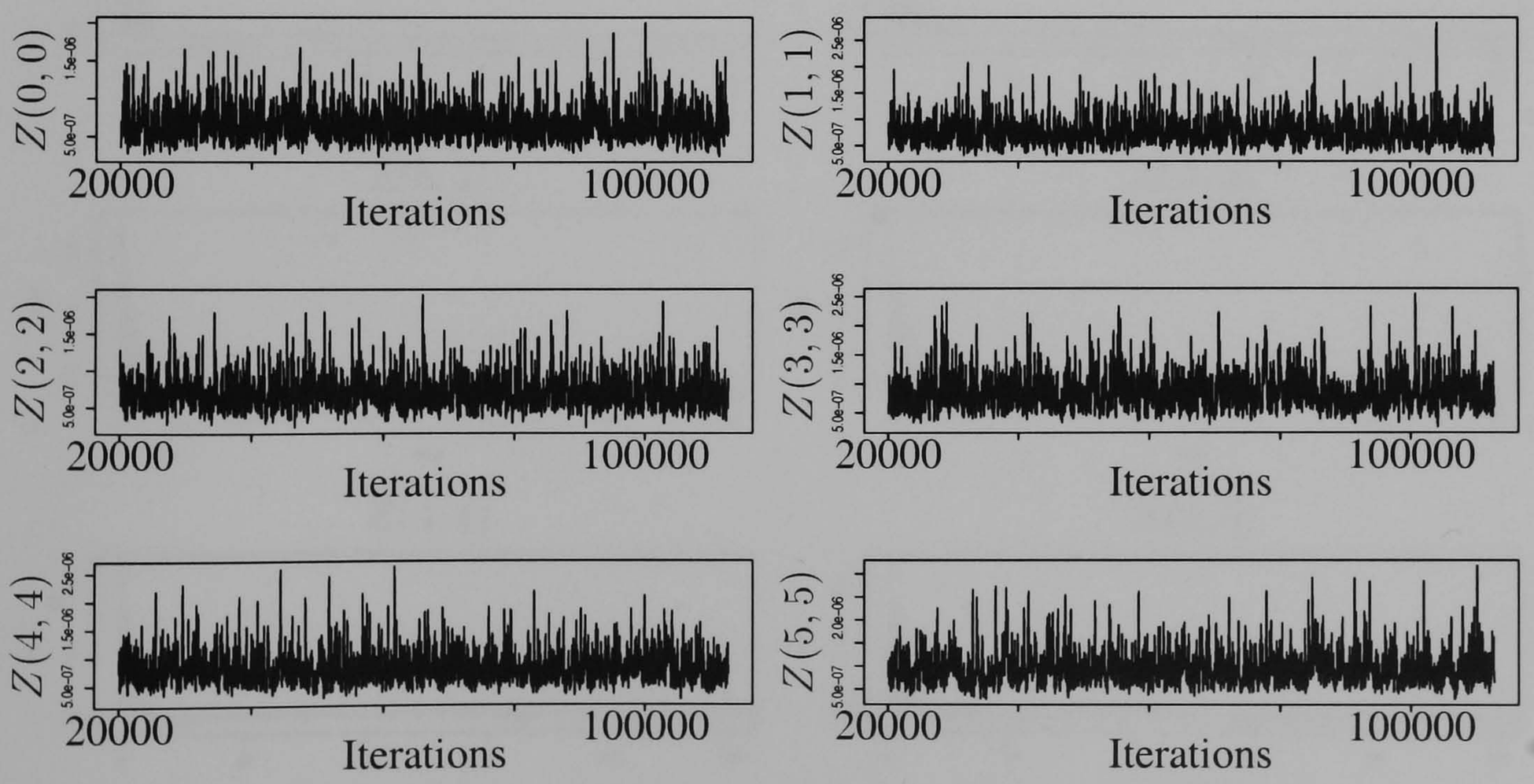


Figure 7.21: Trace plots for selected samples of the state variance components of locally linear dynamic linear model using a block Gibbs sampling algorithm

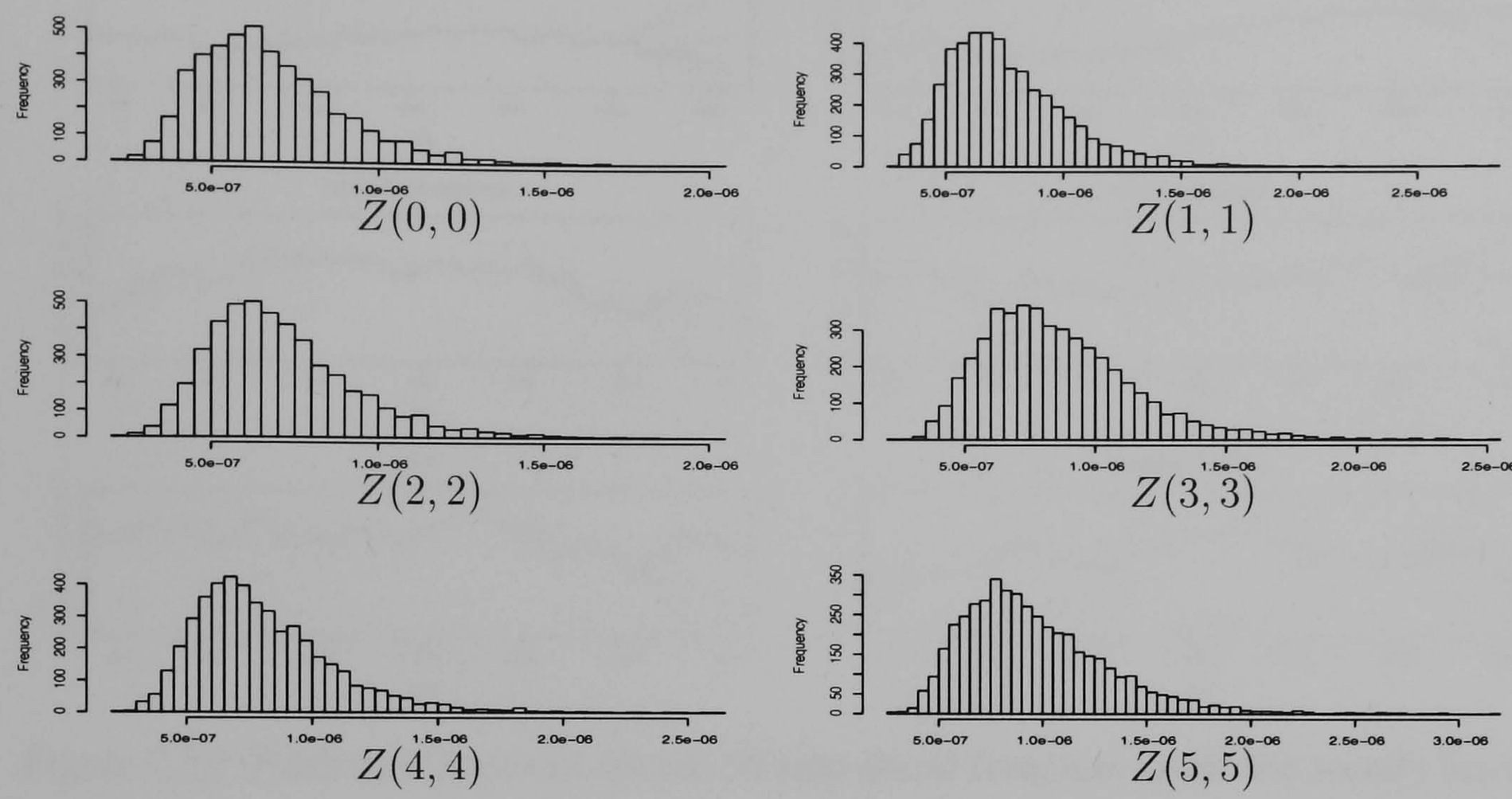


Figure 7.22: Histogram of selected state variance components of locally linear DLM

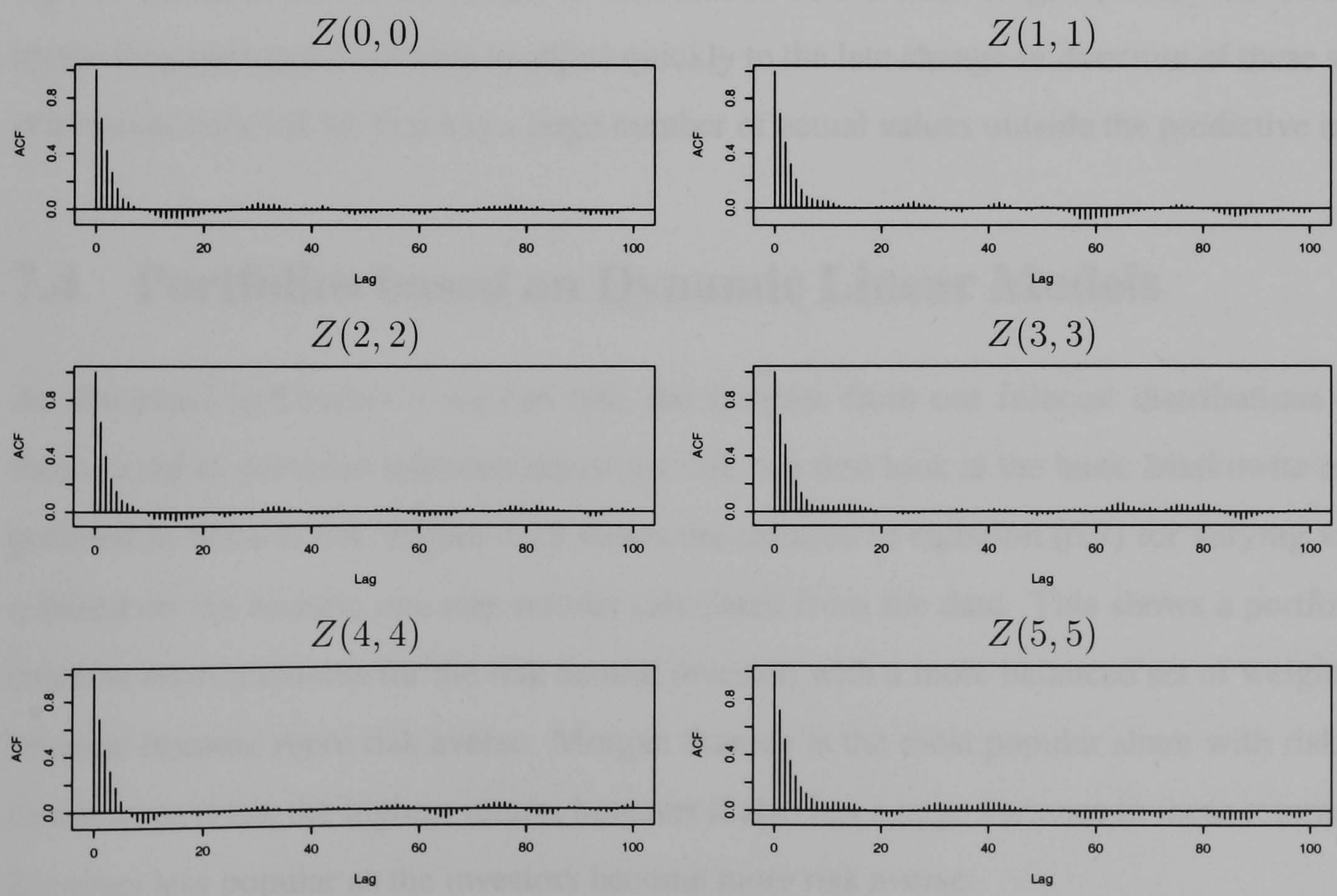


Figure 7.23: Autocorrelation plots for selected elements of the state variance matrix Z.

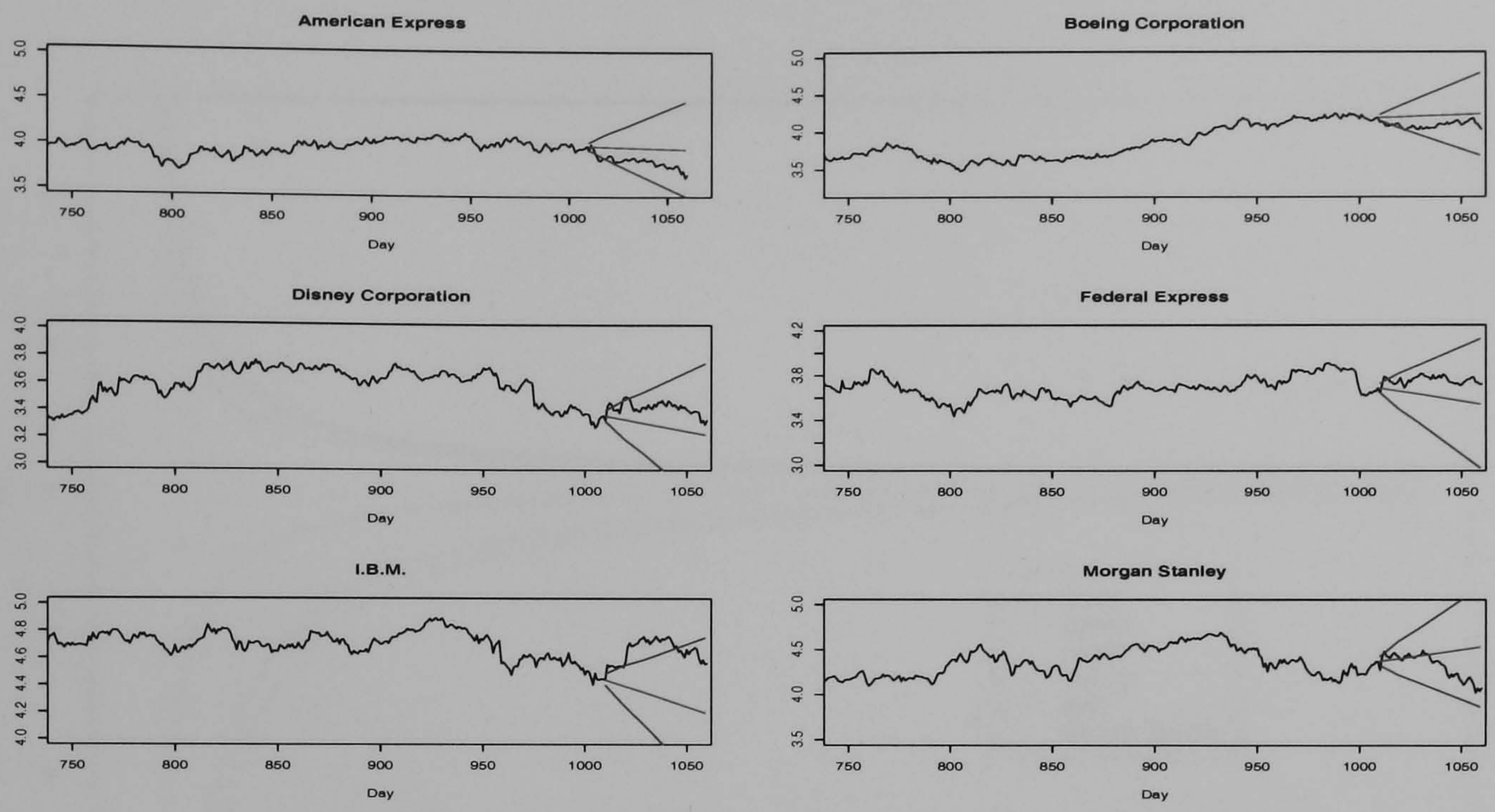


Figure 7.24: Predictive intervals for the 50 step ahead forecasts under the locally linear model

We can also consider the predictive abilities of this model, Figure 7.24 shows the 95% predictive interval as well as the 50% quantile for the up to 50 step ahead forecasts. These again generally capture the behaviour of the series. In the cases of I.B.M. and Morgan Stanley the model picks up the long term trend but fails to adjust quickly to the late change in direction of these series. It is however only I.B.M. that has a large number of actual values outside the predictive interval.

7.4 Portfolios based on Dynamic Linear Models

As discussed in Chapter 6 we can take the samples from our forecast distributions and use these to aid in portfolio selection decisions. We can first look at the basic Markowitz model as outlined in Section 6.4. Figure 7.25 shows the solution to equation (6.7) for varying values of η based on the historic one step returns calculated from the data. This shows a portfolio with extreme short positions for the risk neutral investor, with a more balanced set of weights as the investor become more risk averse. Morgan Stanley is the most popular share with risk neutral investors as it has the highest return, however it also has a large variance in these returns and so becomes less popular as the investors become more risk averse.

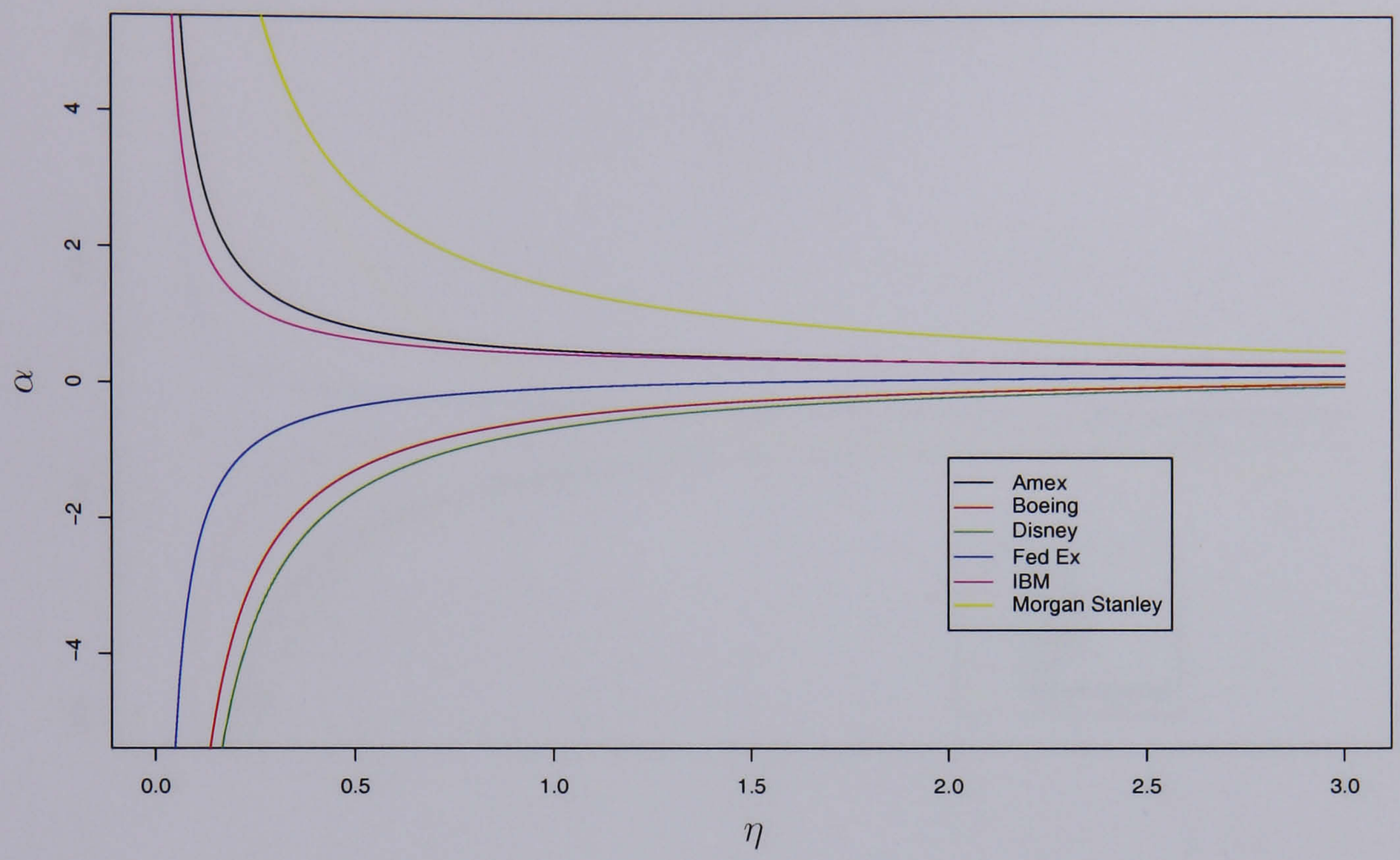


Figure 7.25: Plots of portfolio weights α for varying levels of risk preference η, using a simple Markowitz approach for unconstrained portfolios.

7.4.1 The Locally Constant Model

We can use the forecast values from our locally constant model to calculate returns for 10 days into the future. Figure 7.26 shows the suggested portfolio weights, α for a range of risk aversion parameters η for this 10 step ahead forecast, having calculated summary statistics for the returns distribution and used these in equation (6.7). This shows that although we still have some extreme portfolios being suggested for the risk neutral investor as we look ahead in time the variability of returns effects the portfolio selection decision more.

In Figure 7.27 we see the effect of using the **R** package **quadprog** to constrain the maximisation to prevent shorting, as discussed in Section 6.4.4. This restriction on shorting, results in a portfolio of one share, Morgan Stanley for the completely risk neutral investor, however this proportion declines as the investors become more risk averse. This is again because this share has the highest return but also the highest variability.

In Figure 7.28 we see the effect of constraining the portfolio by using the stochastic optimisation techniques as discussed in Section 6.4.5. Here we see a subtle difference between the two

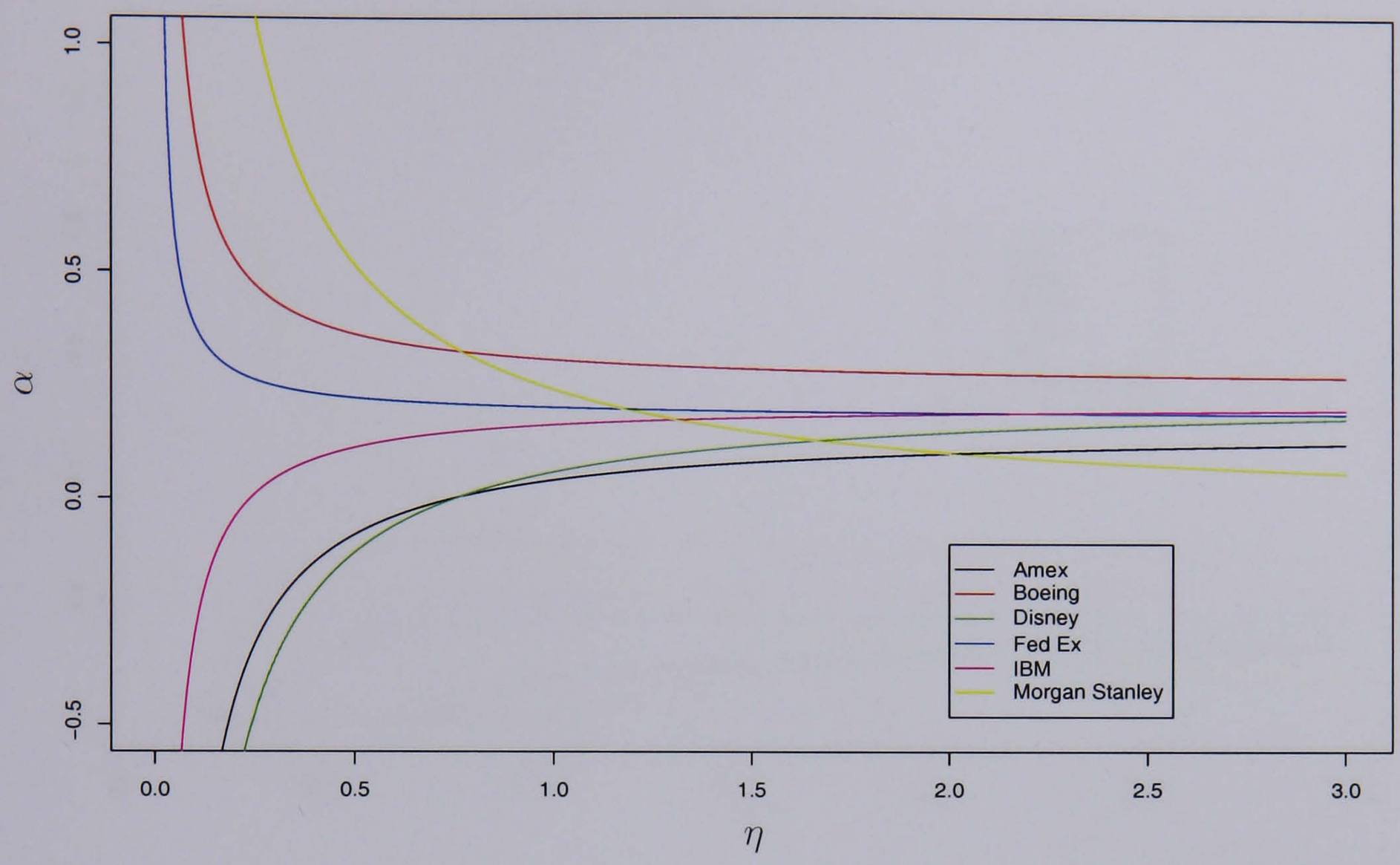


Figure 7.26: Plots of portfolio weights α for varying levels of risk preference η, based on 10 step ahead returns, using an unconstrained maximisation approach.

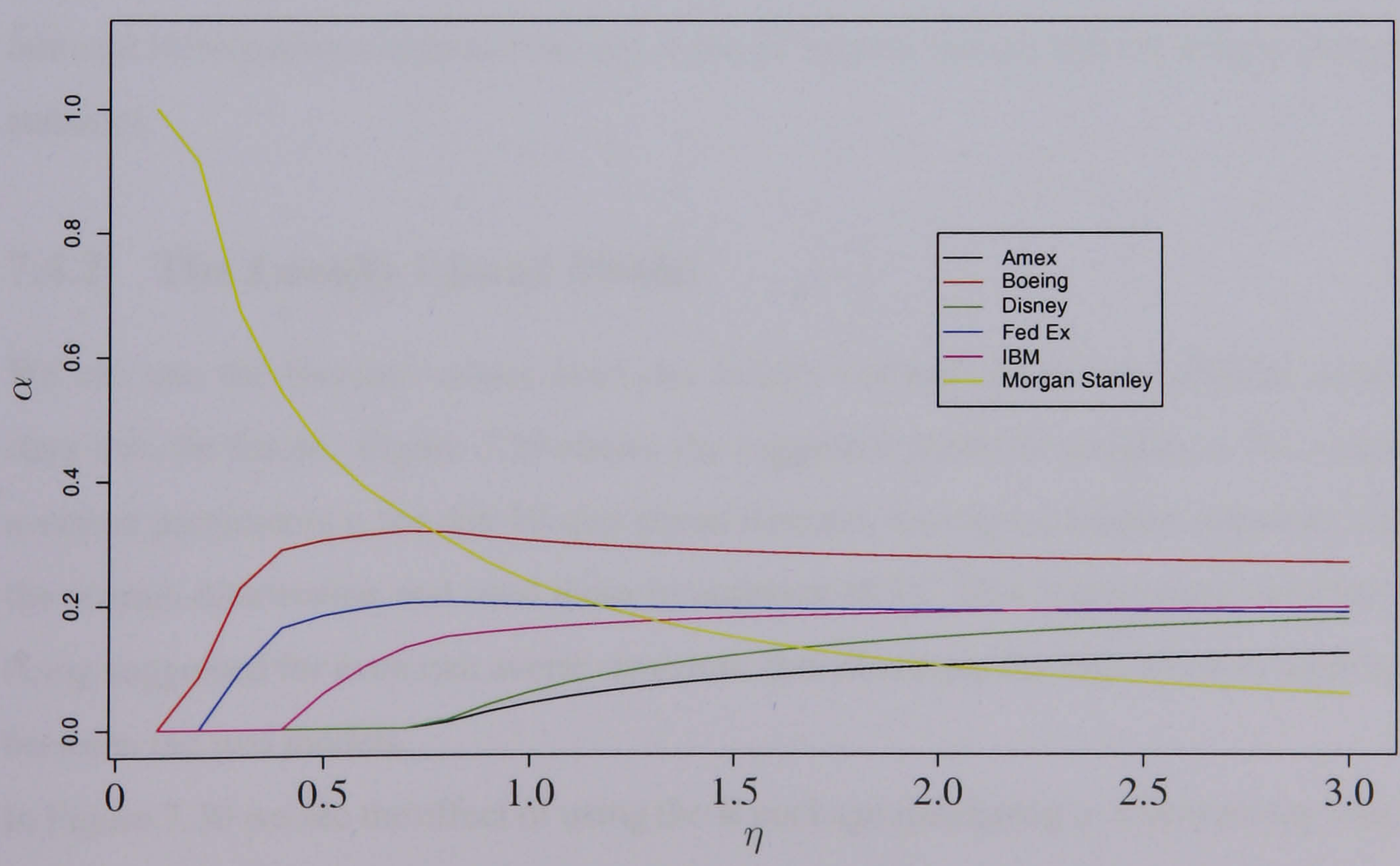


Figure 7.27: Plots of portfolio weights α for varying levels of risk preference η for the 10 step ahead forecast, by solving the constrained quadratic program.

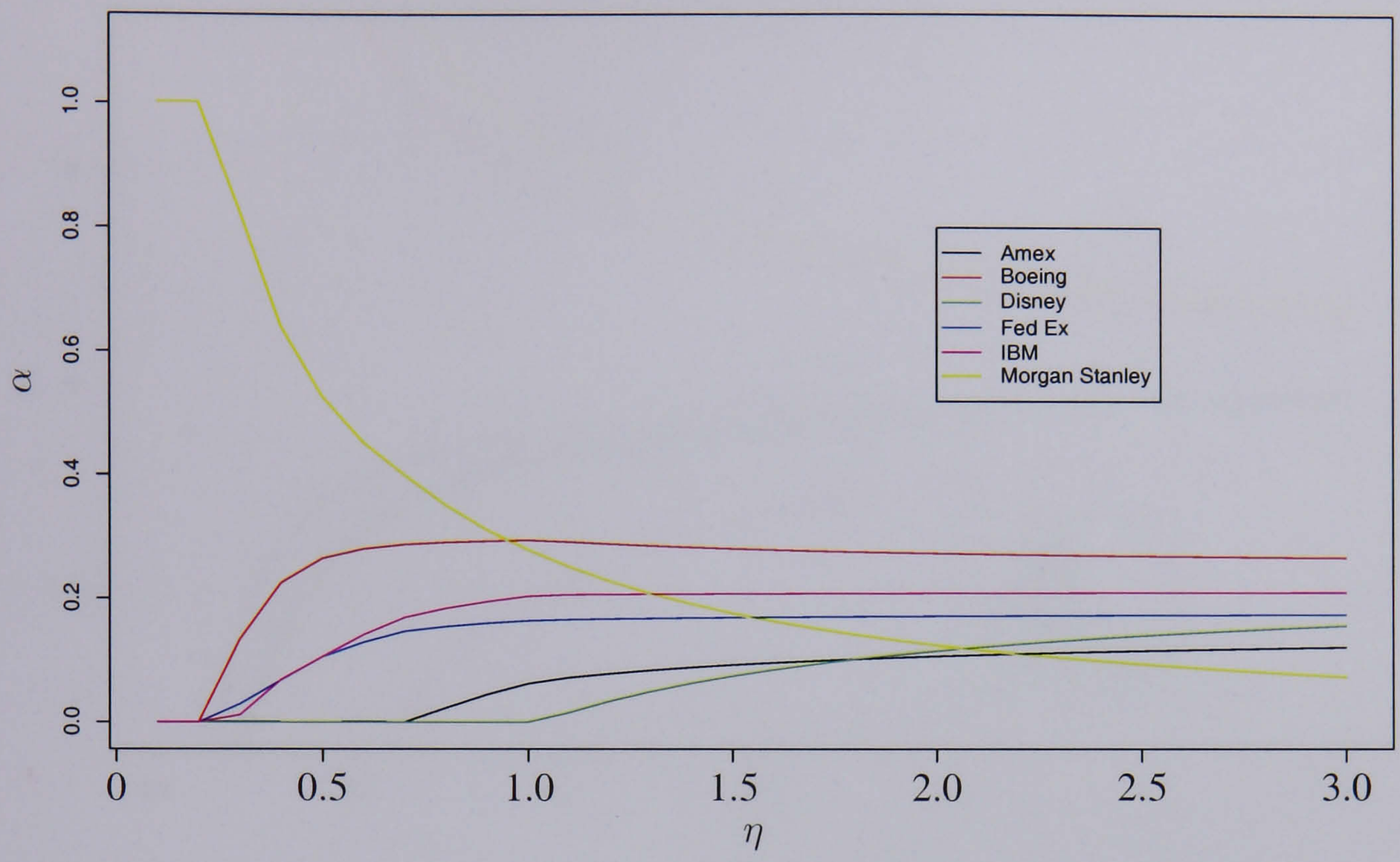


Figure 7.28: Plots of portfolio weights α for varying levels of risk preference η for the 10 step ahead forecast, by stochastic simulation, utilising the full forecast distribution.

suggested ranges of portfolios, these differences are a result of taking into account the distributional information available from the series of returns vectors and not simply using summary statistics.

7.4.2 The Locally Linear Model

We can use the forecast values from our locally constant model to calculate returns for 10 days into the future. Figure 7.29 shows the suggested portfolio weights, α for a range of risk aversion parameters η for this 10 step ahead forecast, having calculated summary statistics for the returns distribution and used these in equation (6.7). This shows some extreme portfolios being suggested for even risk averse investors, this illustrates the difference in predicted returns between the two models.

In Figure 7.30 we see the effect of using the **R** package **quadprog** to constrain the maximisation to prevent shorting, as discussed in Section 6.4.4. This again shows the extreme portfolios for even quite risk averse investors with a second share only being introduced after a the risk

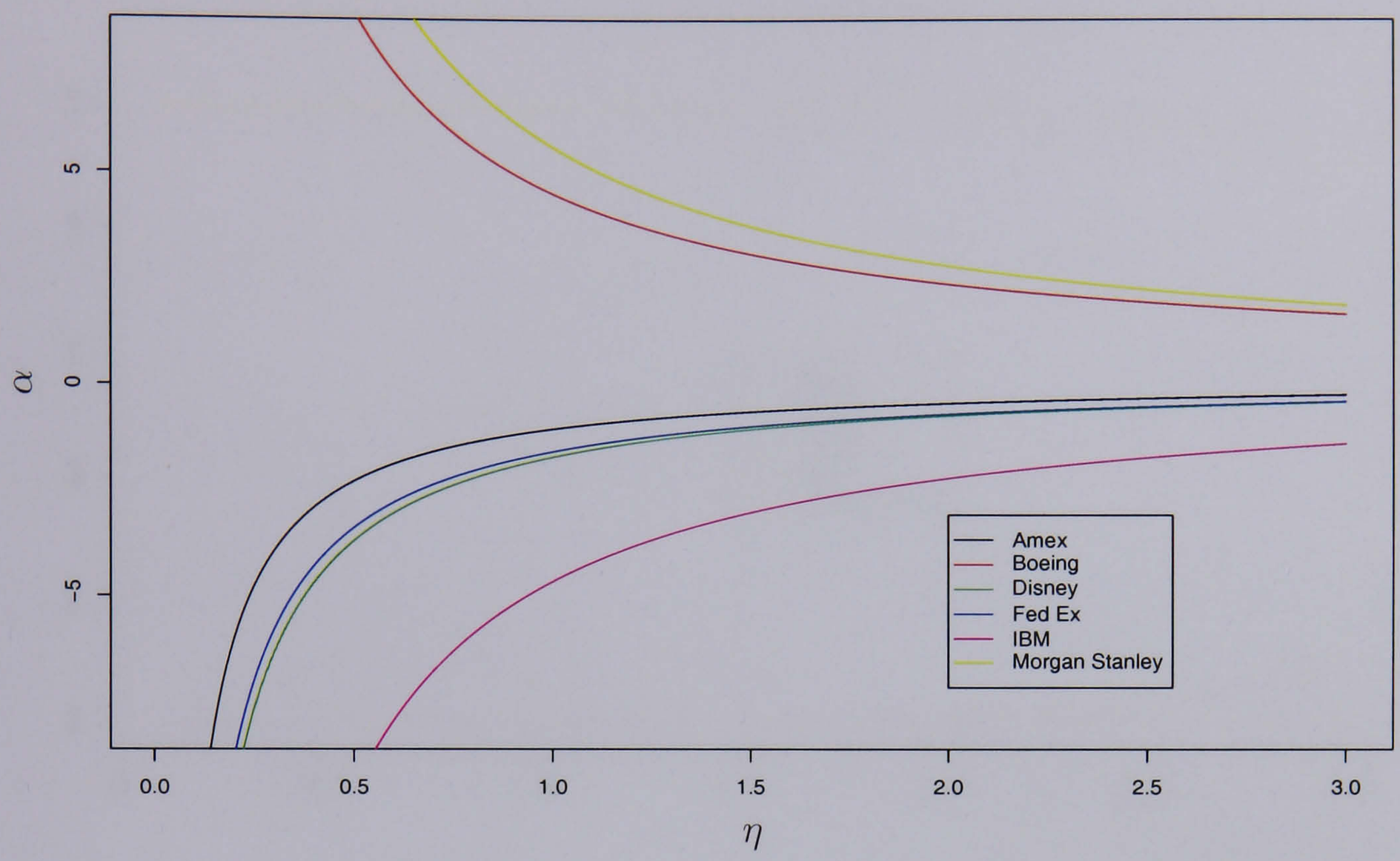


Figure 7.29: Plots of portfolio weights α for varying levels of risk preference η, based on 10 step ahead returns, using an unconstrained maximisation approach.

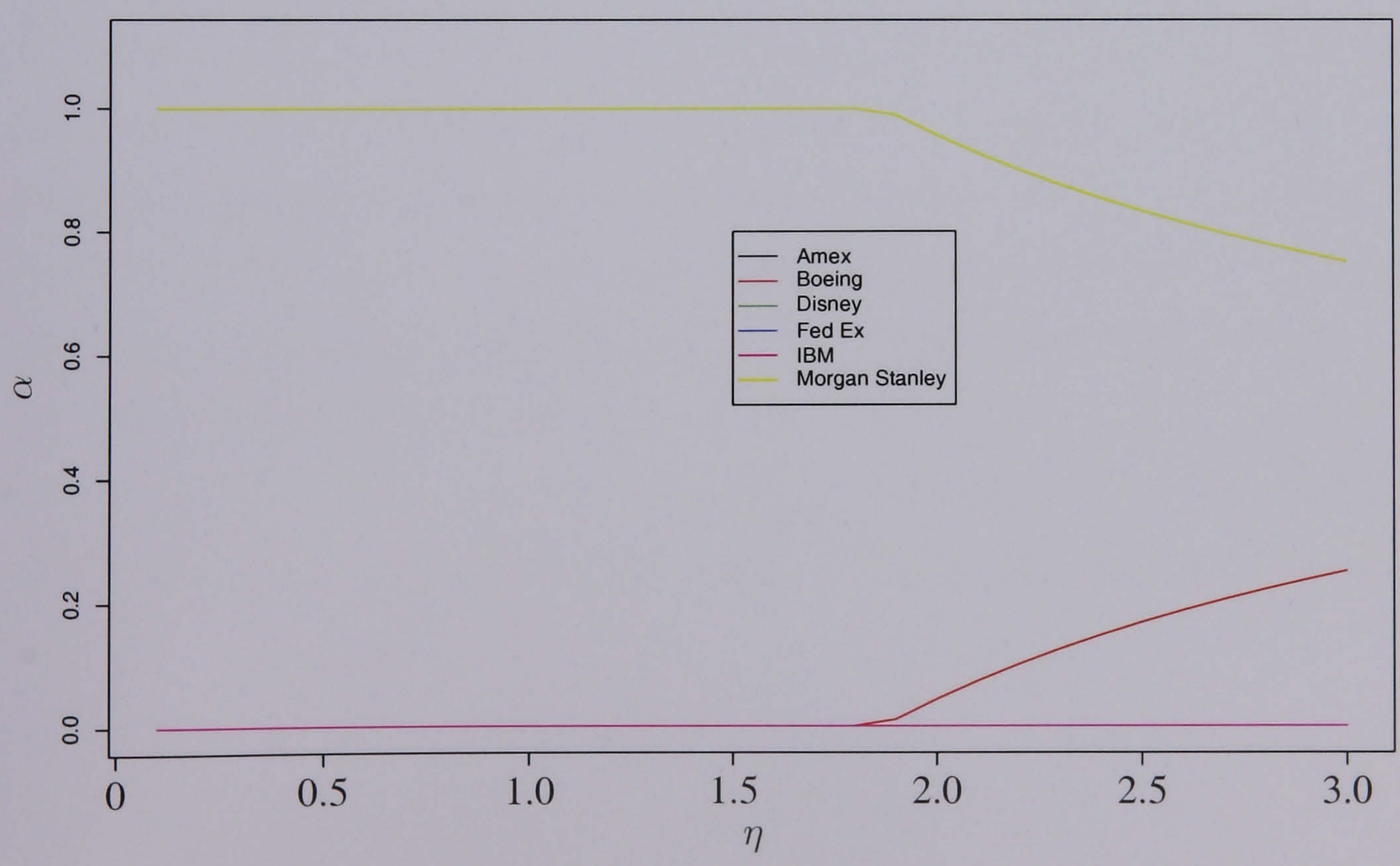


Figure 7.30: Plots of portfolio weights α for varying levels of risk preference η for the 10 step ahead forecast, by solving the constrained quadratic program.

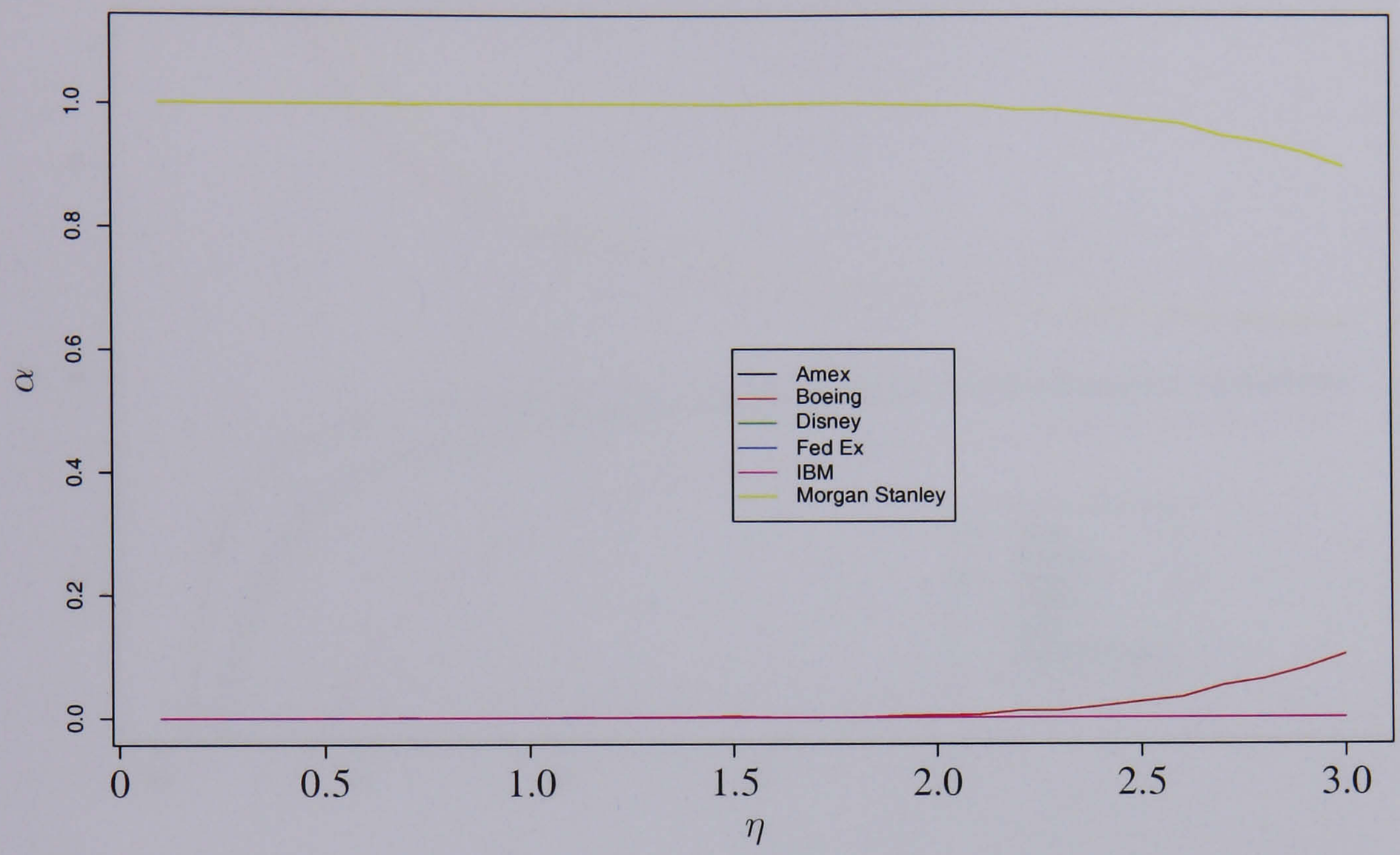


Figure 7.31: Plots of portfolio weights α for varying levels of risk preference η for the 10 step ahead forecast, by stochastic optimisation, utilising the full forecast distribution.

aversion parameter, η has exceeded about 1.8.

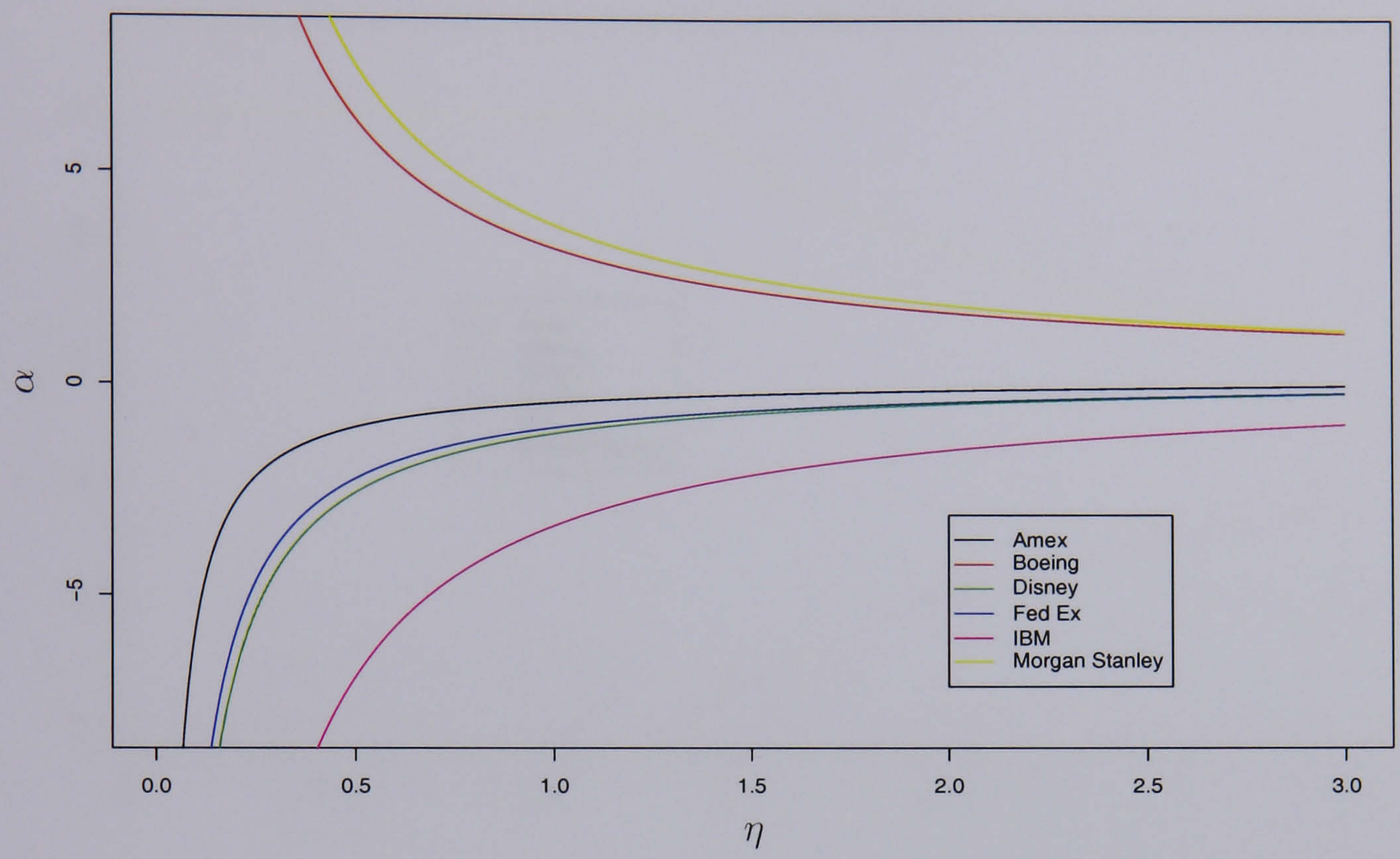


Figure 7.32: Plots of portfolio weights α for varying levels of risk preference η, based on 20 step ahead returns, using an unconstrained maximisation approach.

In Figure 7.31 we see the effect of constraining the portfolio by using the stochastic optimisation techniques as discussed in Section 6.4.5. Here we see a subtle difference between the two suggested ranges of portfolios, this is again a result of taking into account the distributional information available from the series of returns vectors and not simply using summary statistics. Figure 7.32, Figure 7.33, Figure 7.34 and Figure 7.35 further illustrate the effects of changing time horizons on the investment decision. As the investor looks further into the future then the risks of the investment as represented by the variance of the returns influence the decision process. This results in an increasing diversification of the portfolio as the investor looks to the future. Morgan Stanley and Boeing still dominate as these have the highest predicted returns.

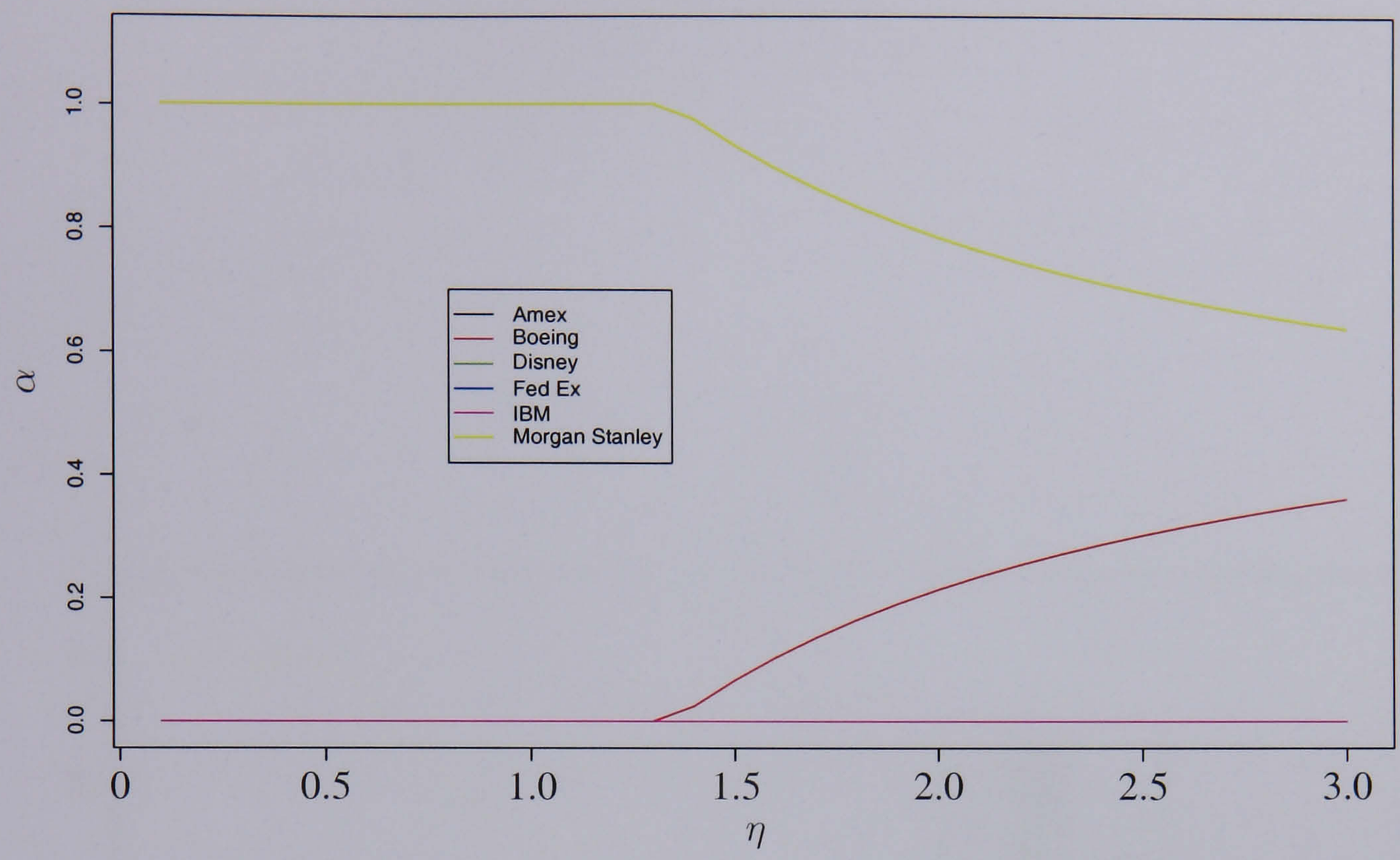


Figure 7.33: Plots of portfolio weights α for varying levels of risk preference η for the 20 step ahead forecast, by solving the constrained quadratic program.

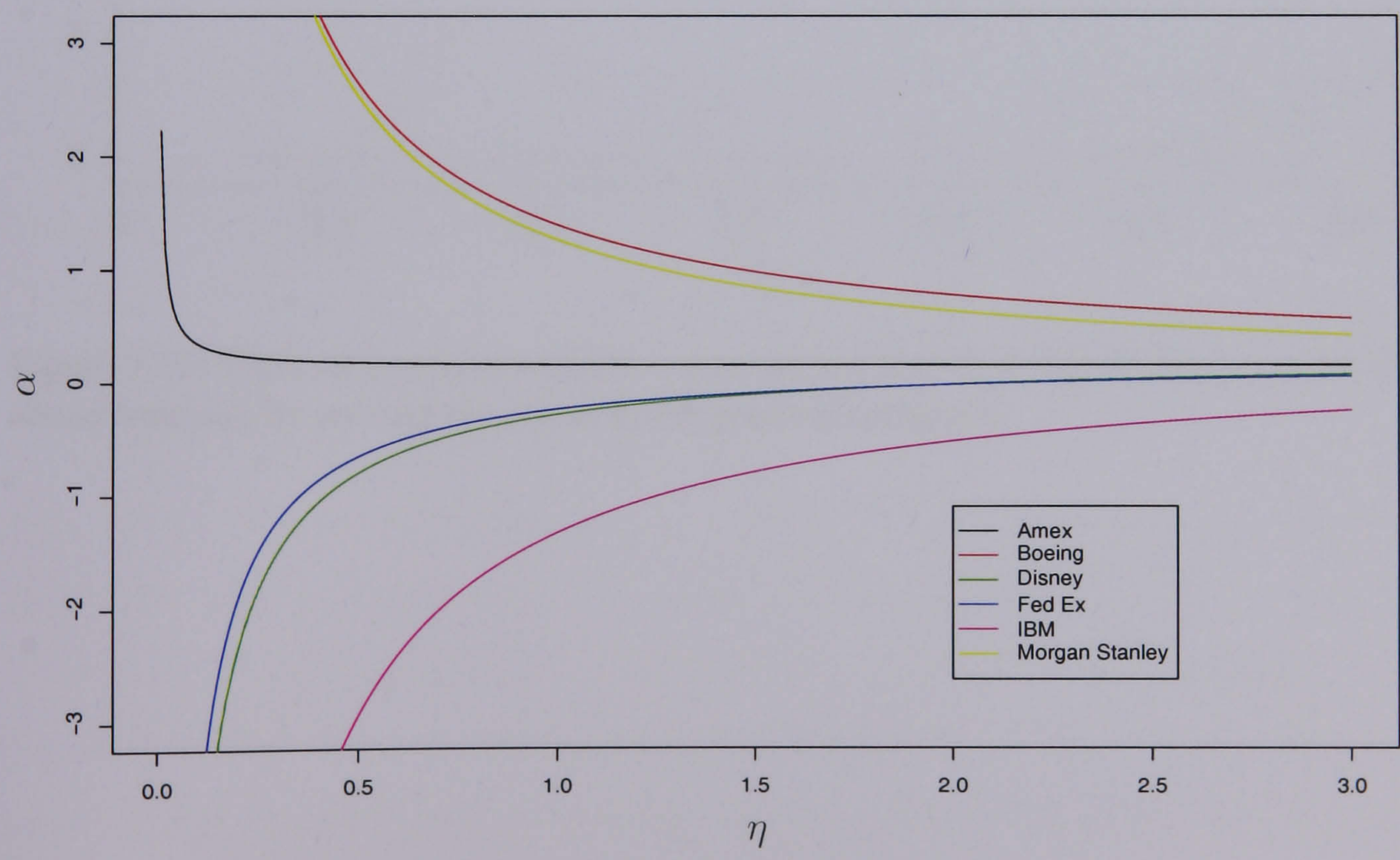


Figure 7.34: Plots of portfolio weights α for varying levels of risk preference η, based on 50 step ahead returns, using an unconstrained maximisation approach.

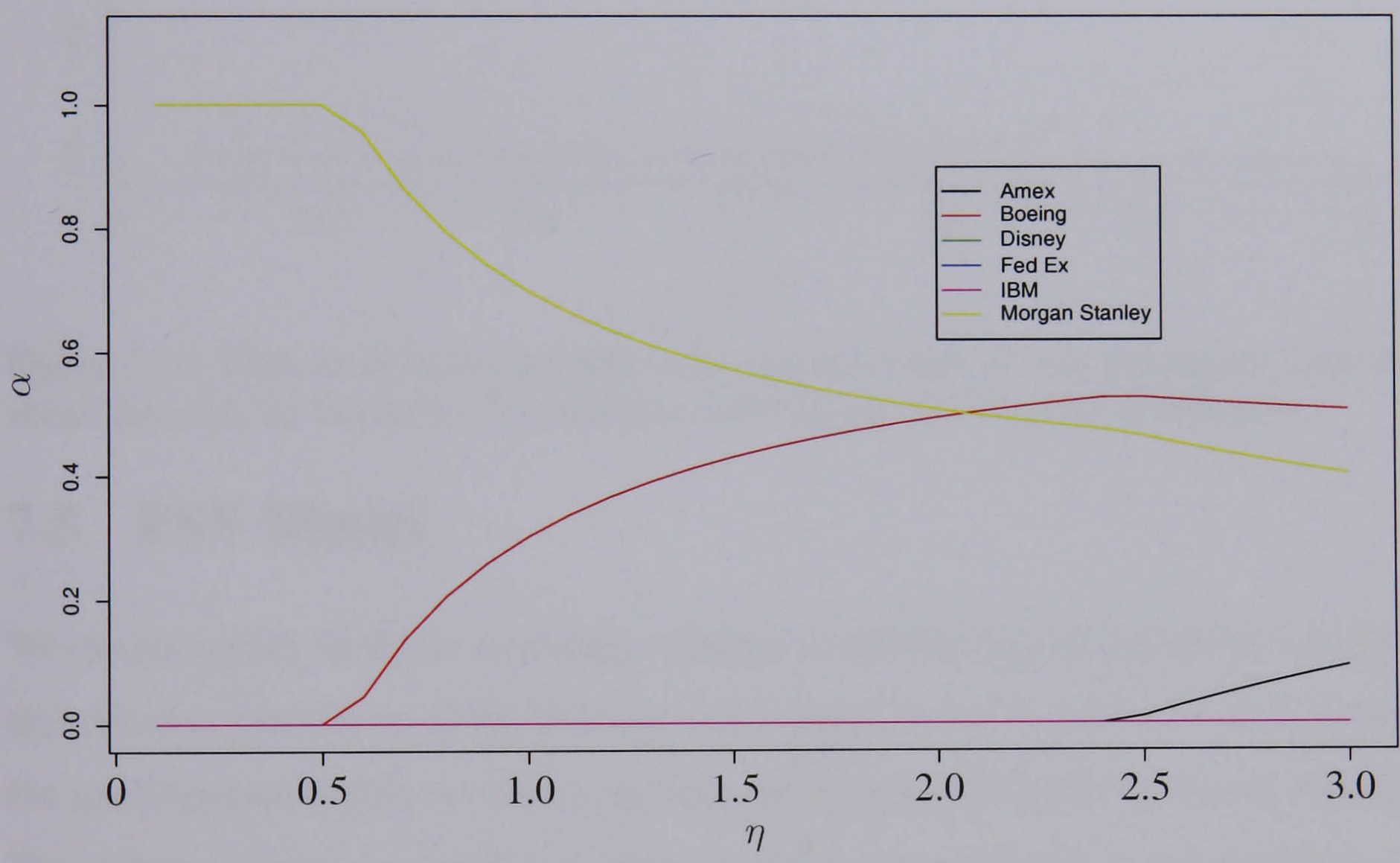


Figure 7.35: Plots of portfolio weights α for varying levels of risk preference η for the 50 step ahead forecast, by solving the constrained quadratic program.

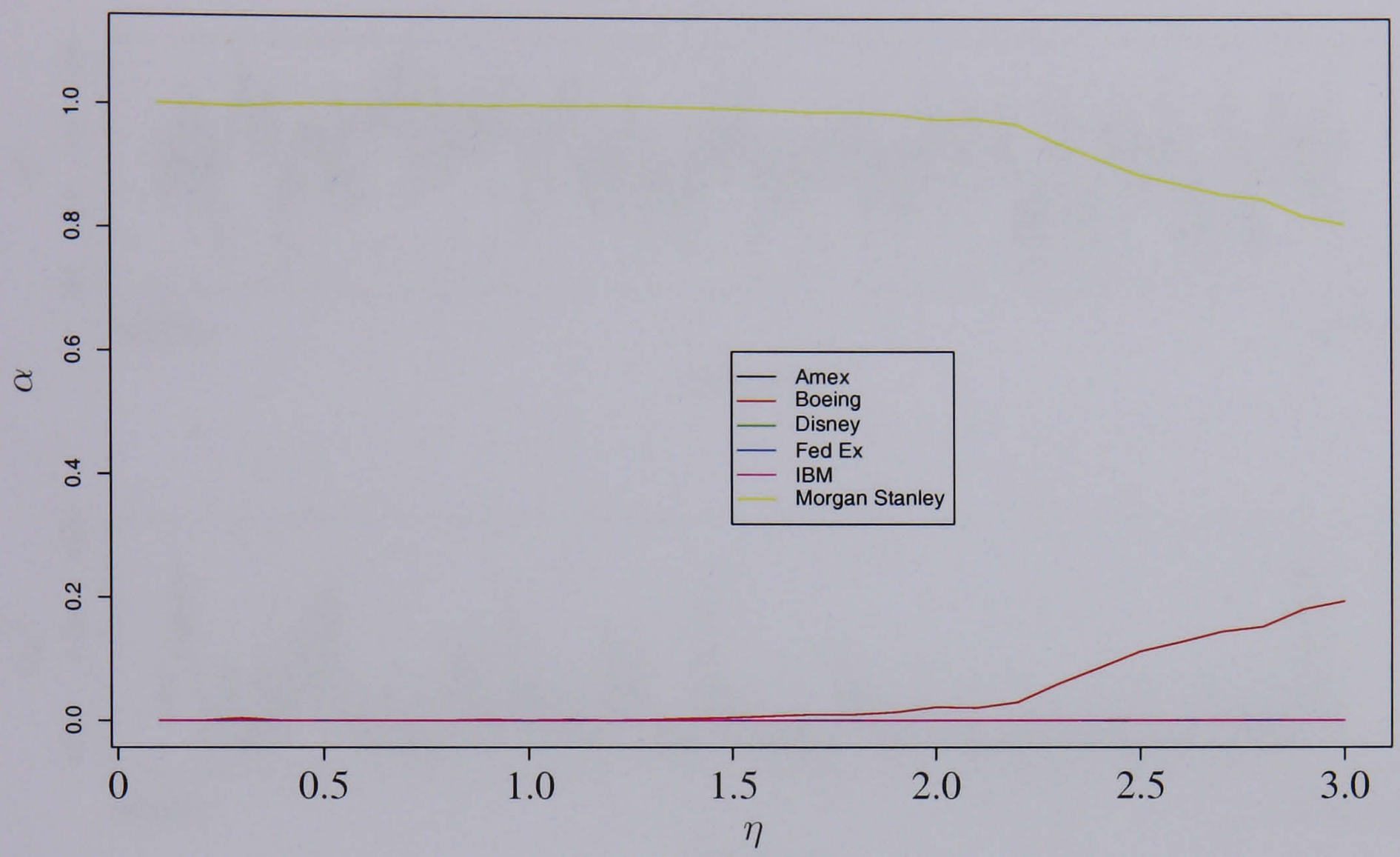


Figure 7.36: Plots of portfolio weights α for varying levels of risk preference η for the 50 step ahead forecast, by stochastic optimisation, utilising the full forecast distribution.

7.5 FSV Model

We can also apply the factor stochastic volatility model introduced in Chapter 4 to this data. We first calculate the returns of the data and then multiply these by 100, following the example of Pitt and Shephard (1999), which we can then model using the factor stochastic volatility model. The **sather** program *examplefsv.sa* which is included in Appendix A was modified to take this data and was run for 150,000 iterations, the first 30,000 were discarded as a burn in and the remaining output was thinned by a factor of 20.

Non-informative $\Gamma(1, 0.00001)$ priors were placed on each of the σ_η terms and flat $N(0, 10)$ priors were placed on each of the μ^τ's and the β parameters. A $\beta(10, 8)$ prior was placed on the ϕ^f and each of the ϕ^τ parameters to reflect the prior belief about these parameters.

Figure 7.37 shows the trace plots for ϕ^f and ${\sigma_\eta}^f$ these suggest the sampler is mixing reasonably well which is confirmed by looking at the autocorrelation plots Figure 7.38. The histograms in Figure 7.39 show the distributions of these two parameters.

Figure 7.40, Figure 7.41 and Figure 7.42 show the same things for τ^2, again suggesting conver-

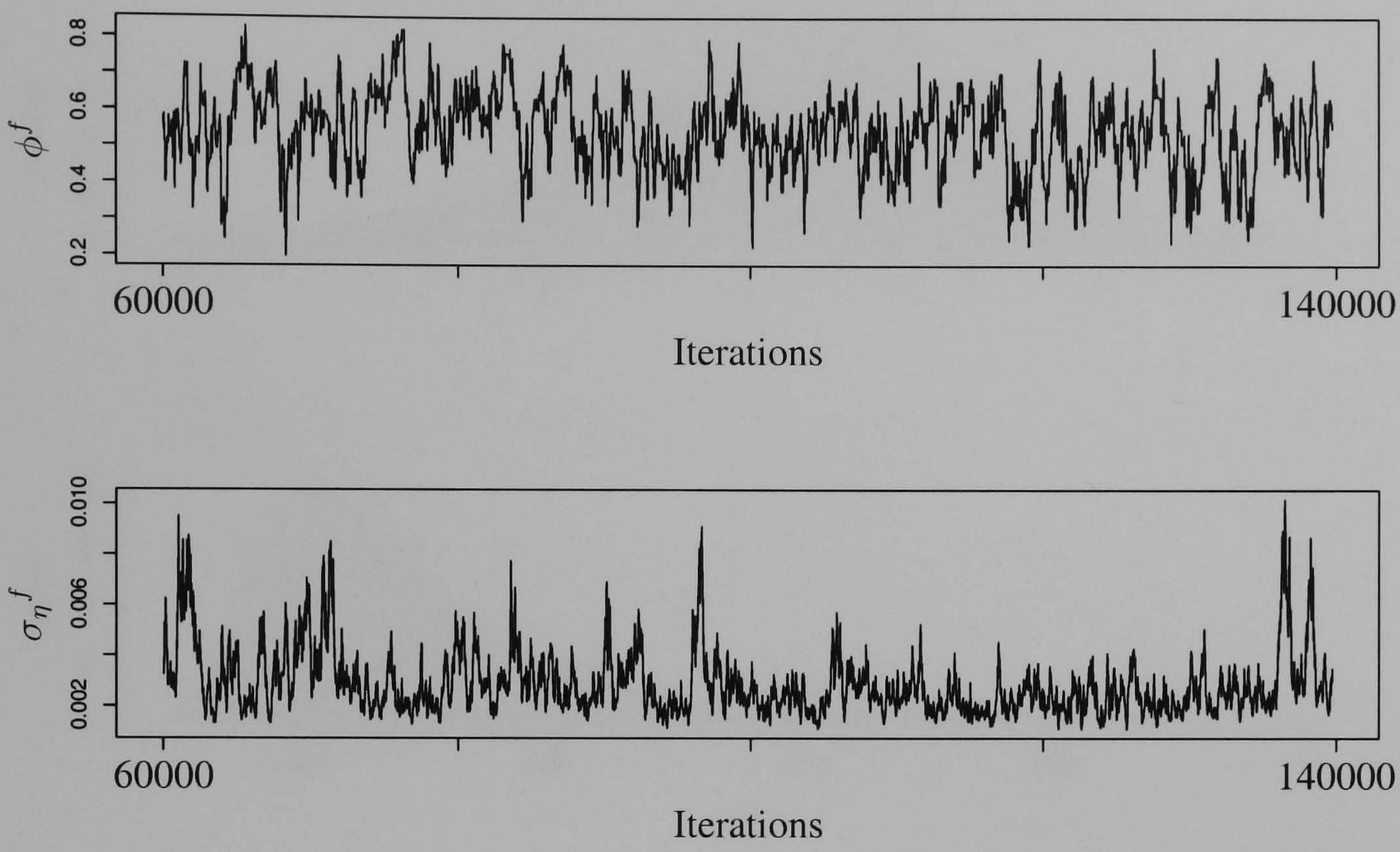


Figure 7.37: Trace plots for the two parameters of the f component of the FSV model

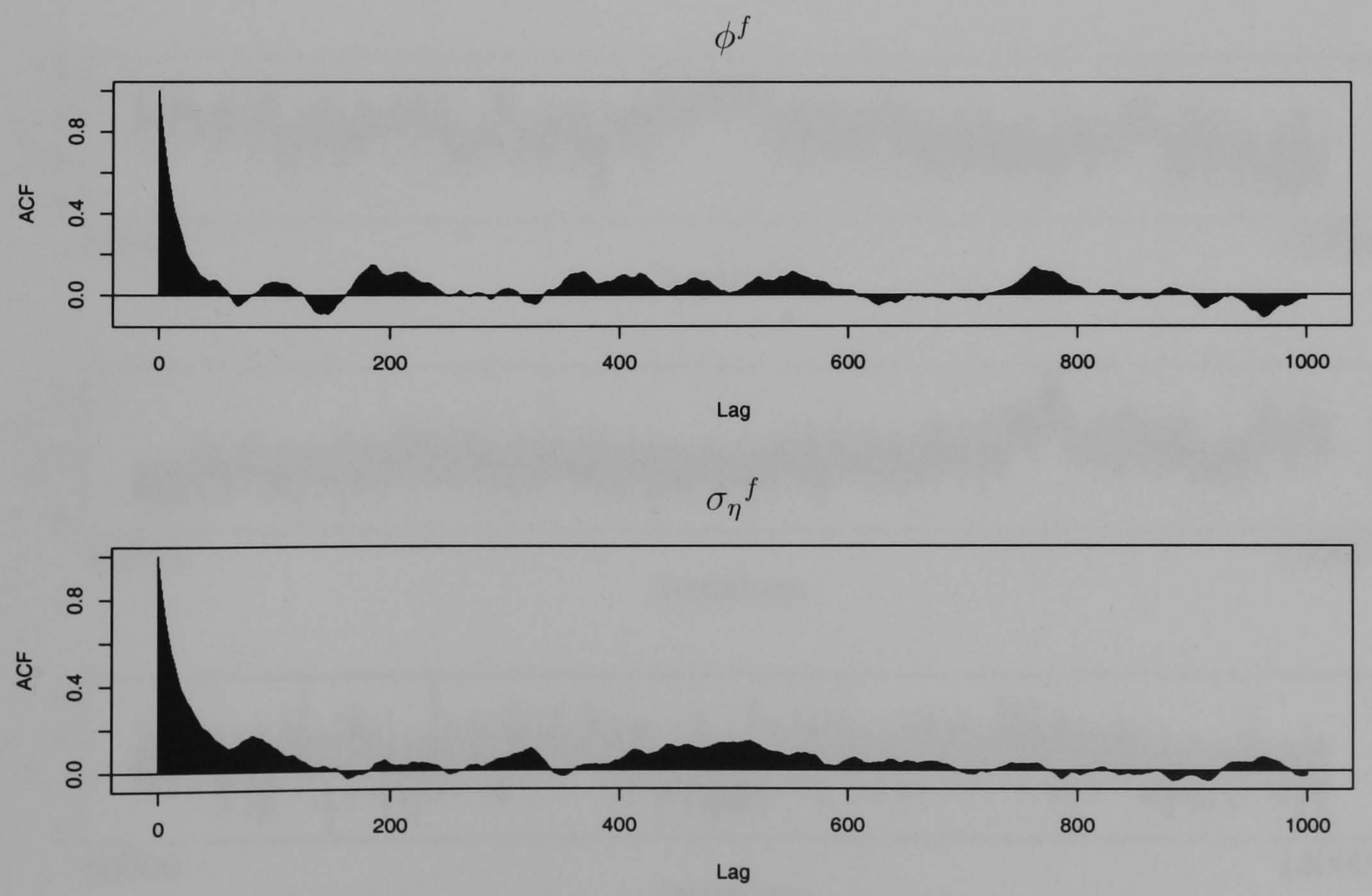


Figure 7.38: Autocorrelation plots for the two parameters of the f component of the FSV model

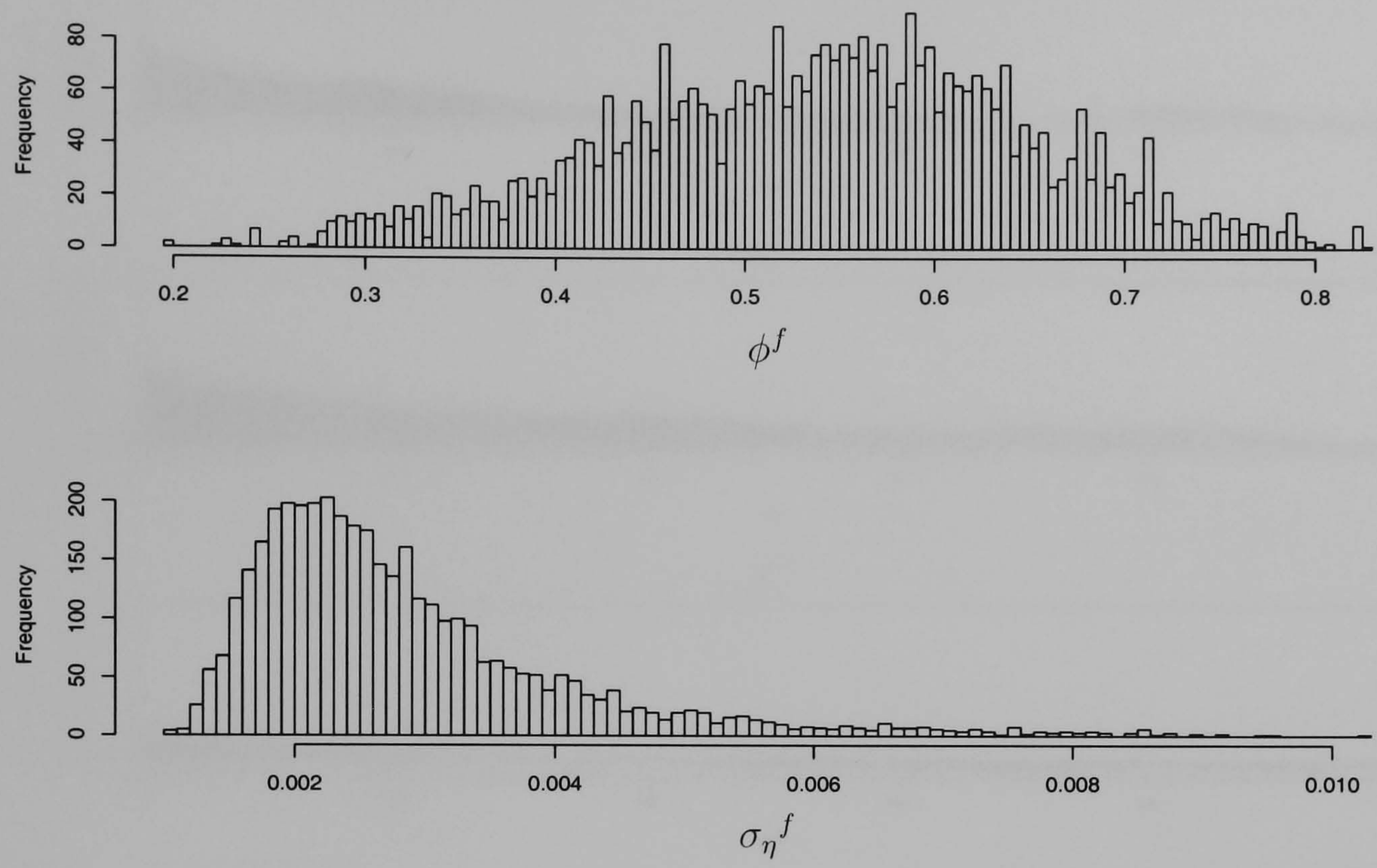


Figure 7.39: Histograms of the samples from the two parameters of the f component of the FSV model

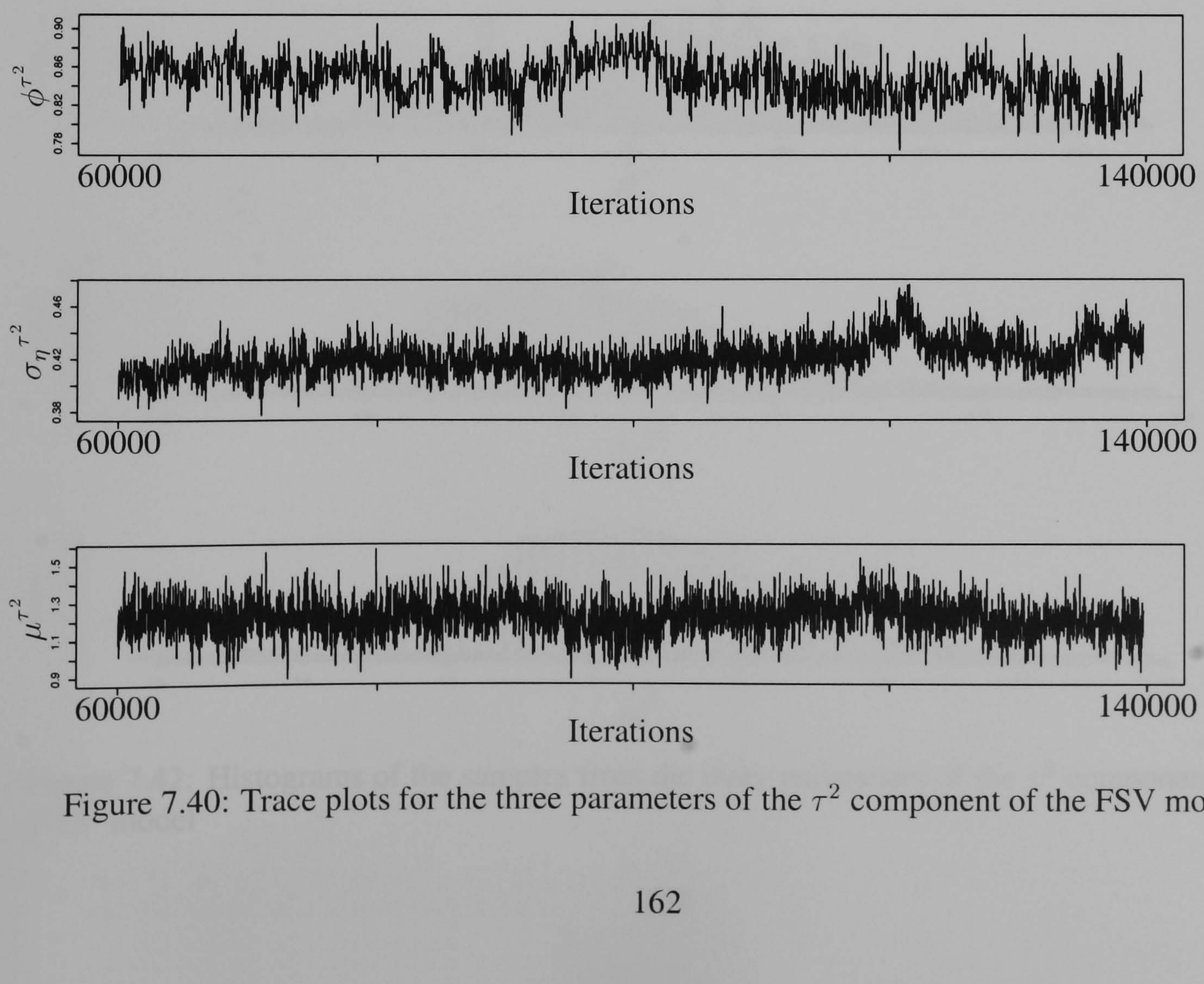


Figure 7.40: Trace plots for the three parameters of the τ^2 component of the FSV model

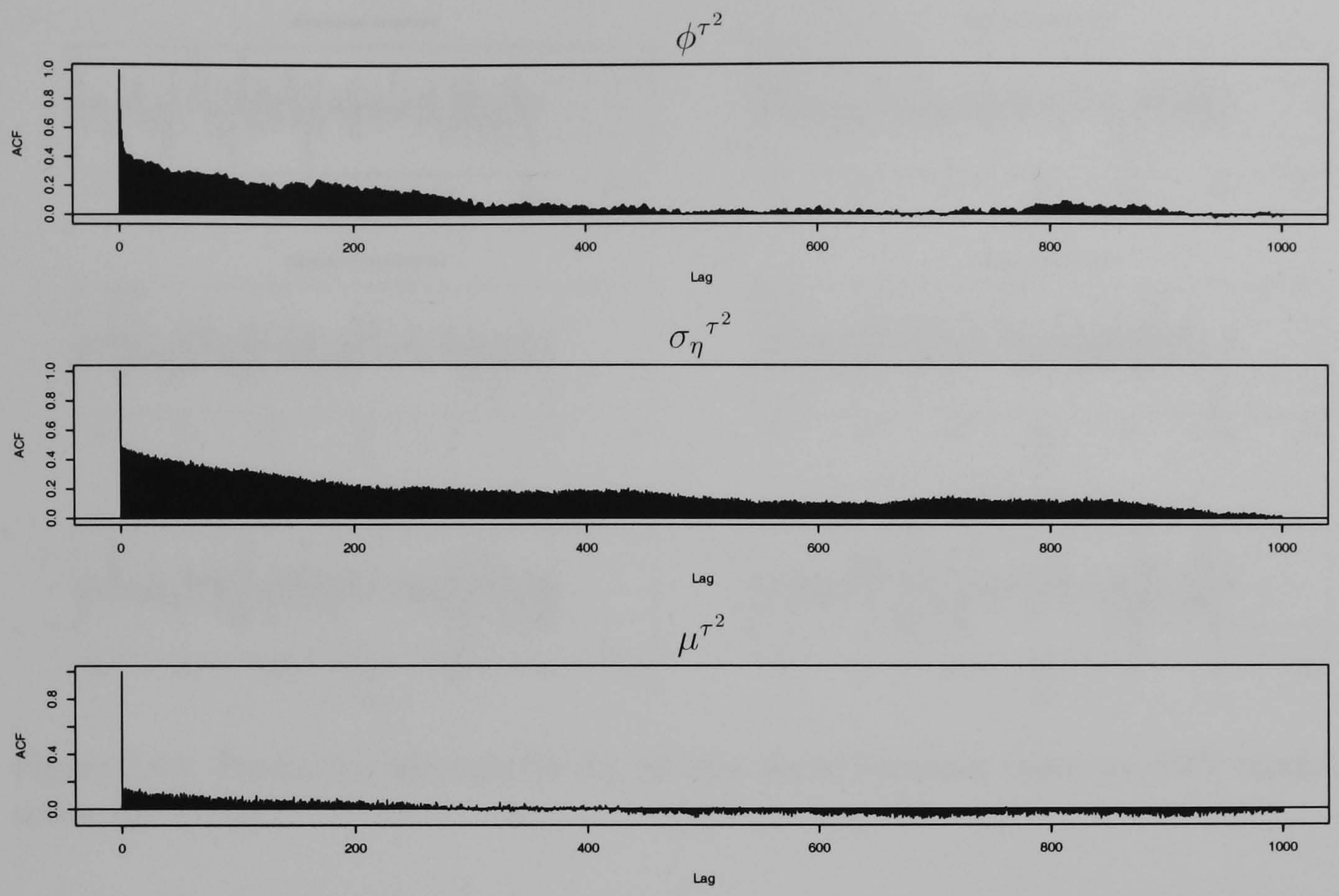


Figure 7.41: Autocorrelation plots for the two parameters of the τ^2 component of the FSV model

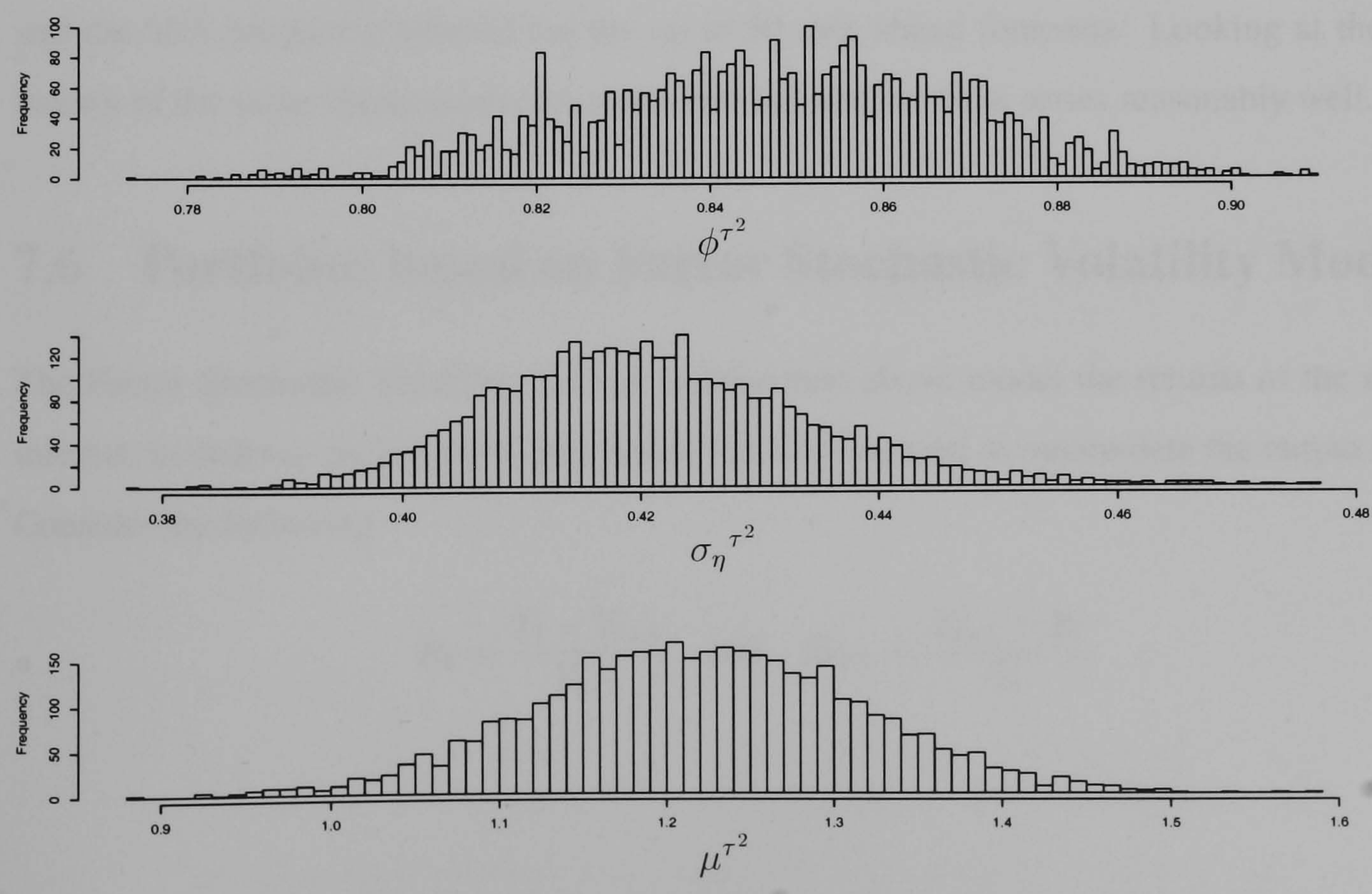


Figure 7.42: Histograms of the samples from the three parameters of the τ^2 component of the FSV model

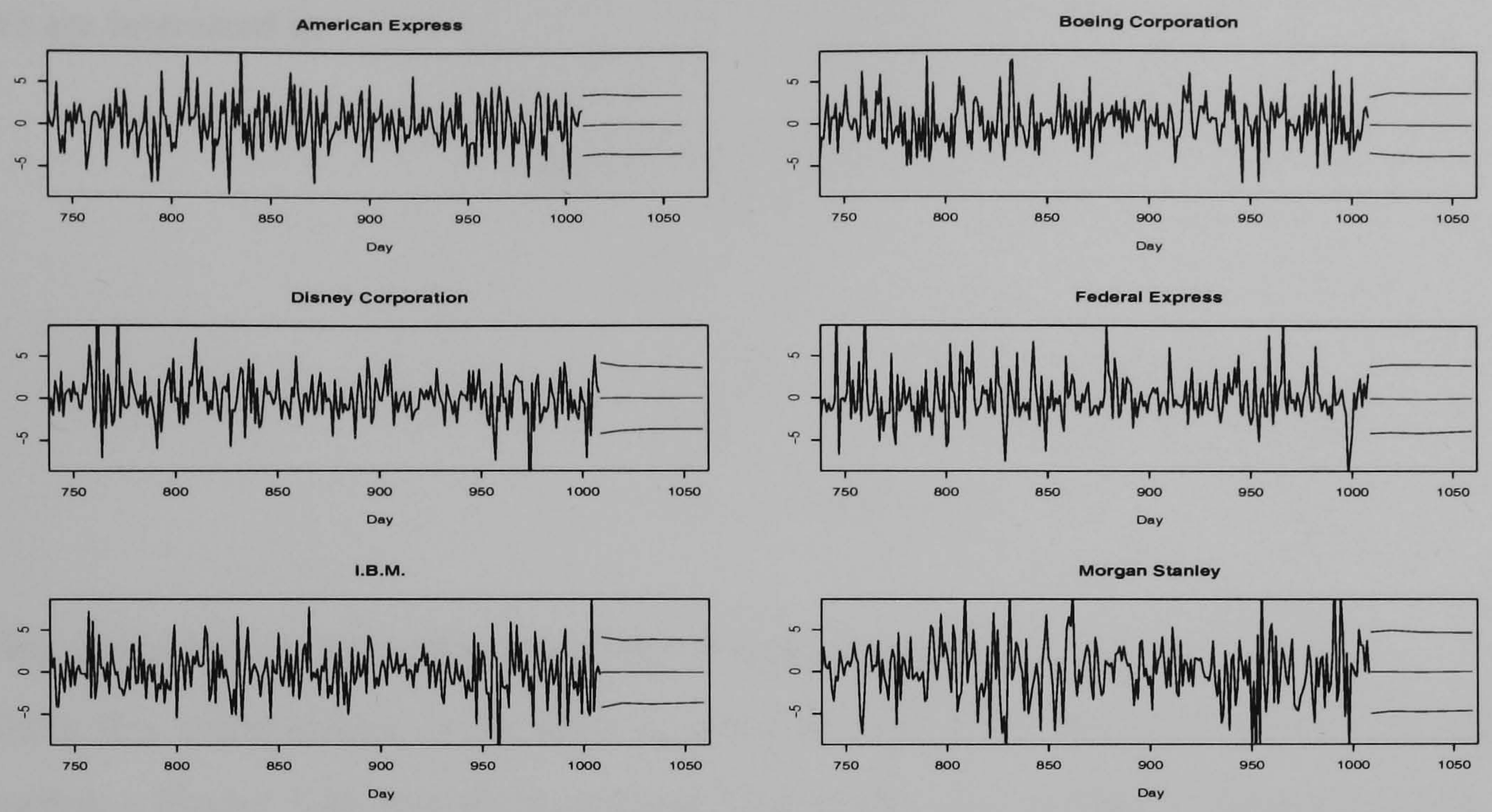


Figure 7.43: Predictive intervals for the 50 step ahead forecasts under the FSV model, for the series $R_T \times 100$.

gence to values which appear reasonable given the data and our expectations for these kind of models.

Figure 7.43 shows the last 250 actual values of our series, $100 \times R_t$, as well as the 50% quantile and the 95% predictive interval for the up to 50 step ahead forecasts. Looking at the recent history of the series these appear to capture the behaviour of the series reasonably well.

7.6 Portfolios based on Factor Stochastic Volatility Models

The Factor Stochastic Volatility models as described above model the returns of the series of interest, in order to look at the k step ahead forecast we need to manipulate the output slightly. Consider the following

$$R_t = \frac{Y_t - Y_{t-1}}{Y_{t-1}} \quad \text{and} \quad R_{t+1} = \frac{Y_{t+1} - Y_t}{Y_t}.$$

We are interested in

$$\begin{aligned} R'_t &= \frac{Y_{t+1} - Y_{t-1}}{Y_{t-1}} \\ &= \frac{Y_{t+1}}{Y_{t-1}} - 1 \\ &= \frac{Y_{t+1}Y_t}{Y_t Y_{t-1}} - 1 \\ &= (R_{t+1} + 1)(R_t + 1) - 1 \end{aligned}$$

This generalises to the k step ahead case straight forwardly.

Using this manipulation of the data we can look initially at the unconstrained 10 step ahead portfolio, Figure 7.44, this shows a range of portfolios dominated by Federal Express, Disney Corporation and I.B.M. these three dominate as they have positive mean returns. We can constrain this portfolio to prevent shorting. Figure 7.45 shows the range of portfolios generated by using the summary statistics and solving the Markowitz quadratic program, (6.7). Here we see a more diversified portfolio being recommended especially for the more risk averse investor. Using the full forecast distributional information we obtain the portfolios as seen in Figure 7.46, these varies slightly from those obtained when using only summary statistics.

Figure 7.47 shows the 50 step ahead unconstrained portfolios, these show that as we look further into the future the increased uncertainty about returns impacts on the portfolios recommended. At this point American Express has become the share with the highest mean return and hence dominates the portfolio. Figure 7.48 and Figure 7.49 show the effects of constraining the solution using summary statistics and the full forecast distribution respectively. These still show portfolios which are dominated by the shares with the highest returns, especially for the risk neutral investor, however there is a noticeable diversification of the portfolio.

7.7 Summary

We have shown that we can apply both the EM algorithm and the block Gibbs sampling algorithm to fit both locally constant and locally linear dynamic linear models to some real market data. These two methodologies show consistent results, and the up to 50 step ahead forecasts

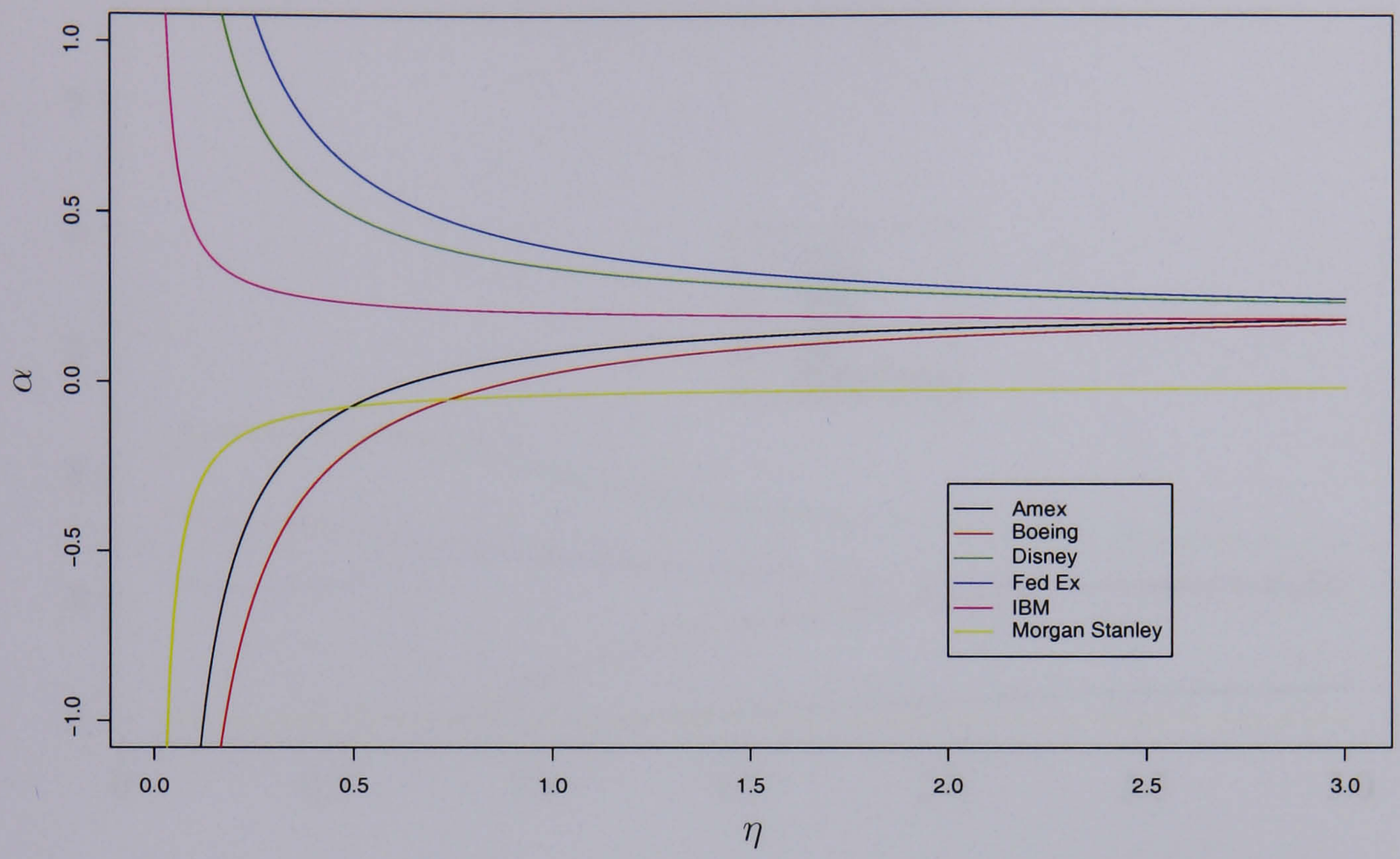


Figure 7.44: Plots of portfolio weights α for varying levels of risk preference η, based on 10 step ahead returns derived from the FSV model.

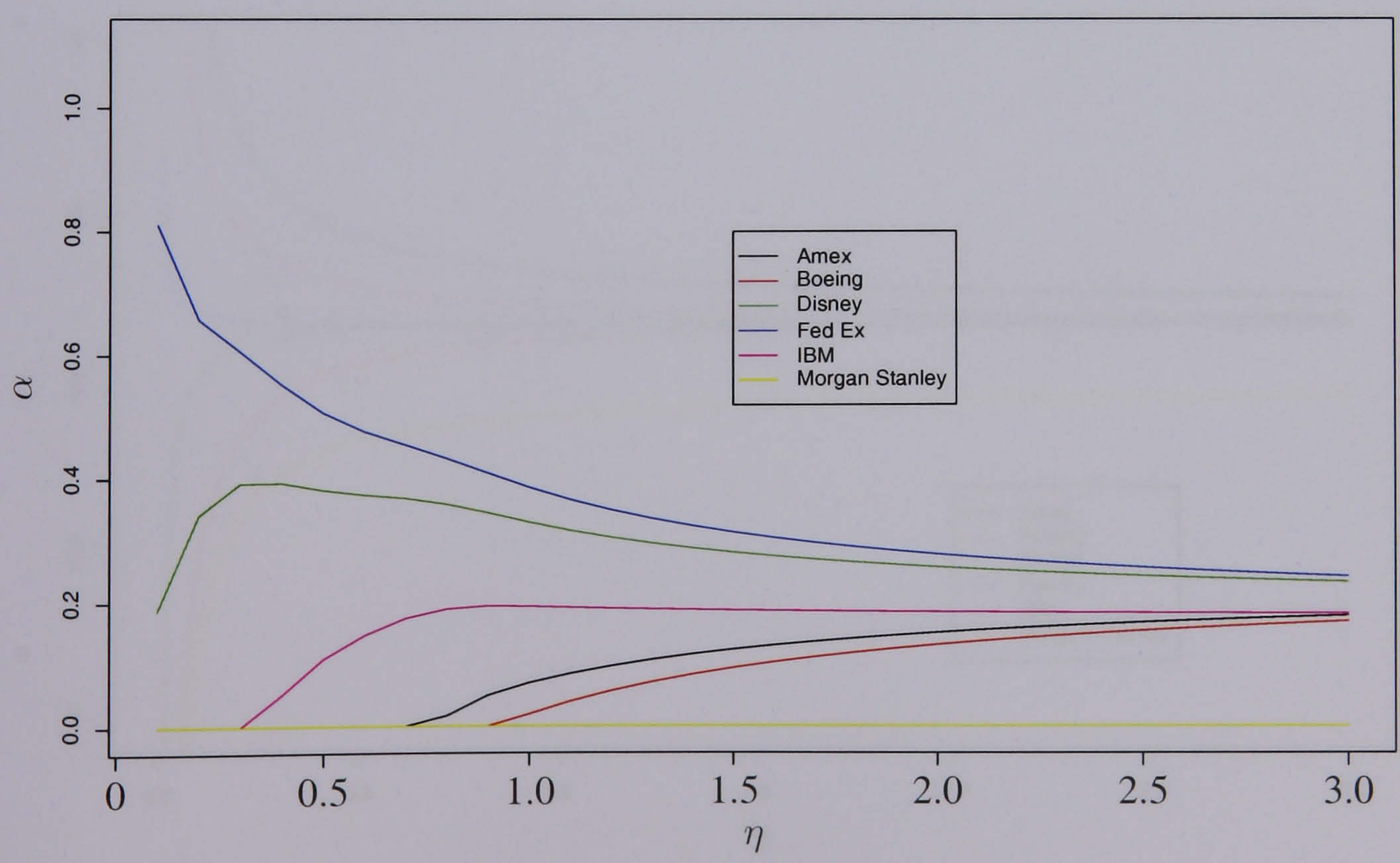


Figure 7.45: Plots of portfolio weights α for varying levels of risk preference η for the 10 step ahead forecast, by solving the constrained quadratic program based on the FSV model.

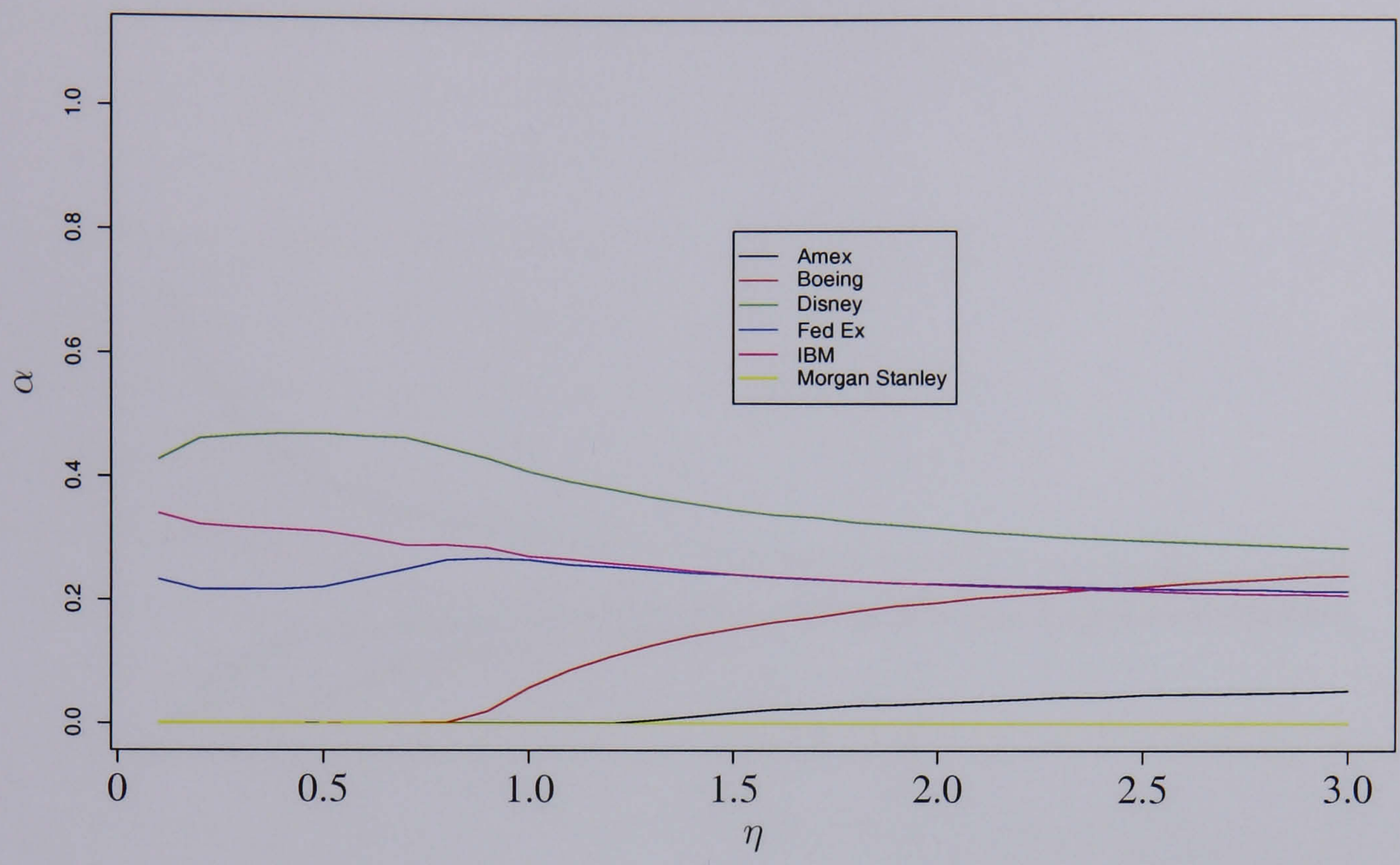


Figure 7.46: Plots of portfolio weights α for varying levels of risk preference η for the 10 step ahead forecast, by stochastic optimisation, utilising the full forecast distribution.

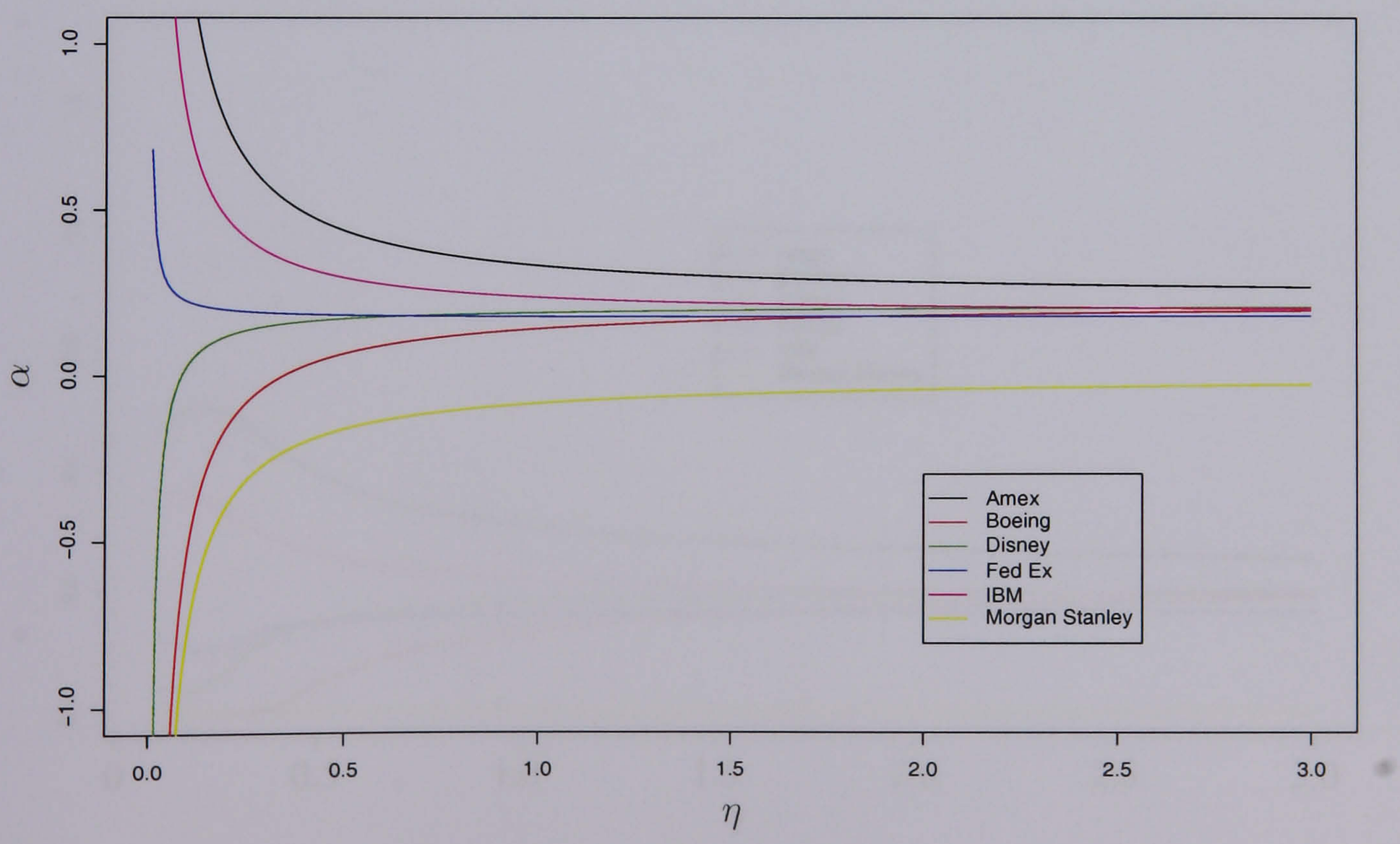


Figure 7.47: Plots of portfolio weights α for varying levels of risk preference η, based on 50 step ahead returns derived from the FSV model.

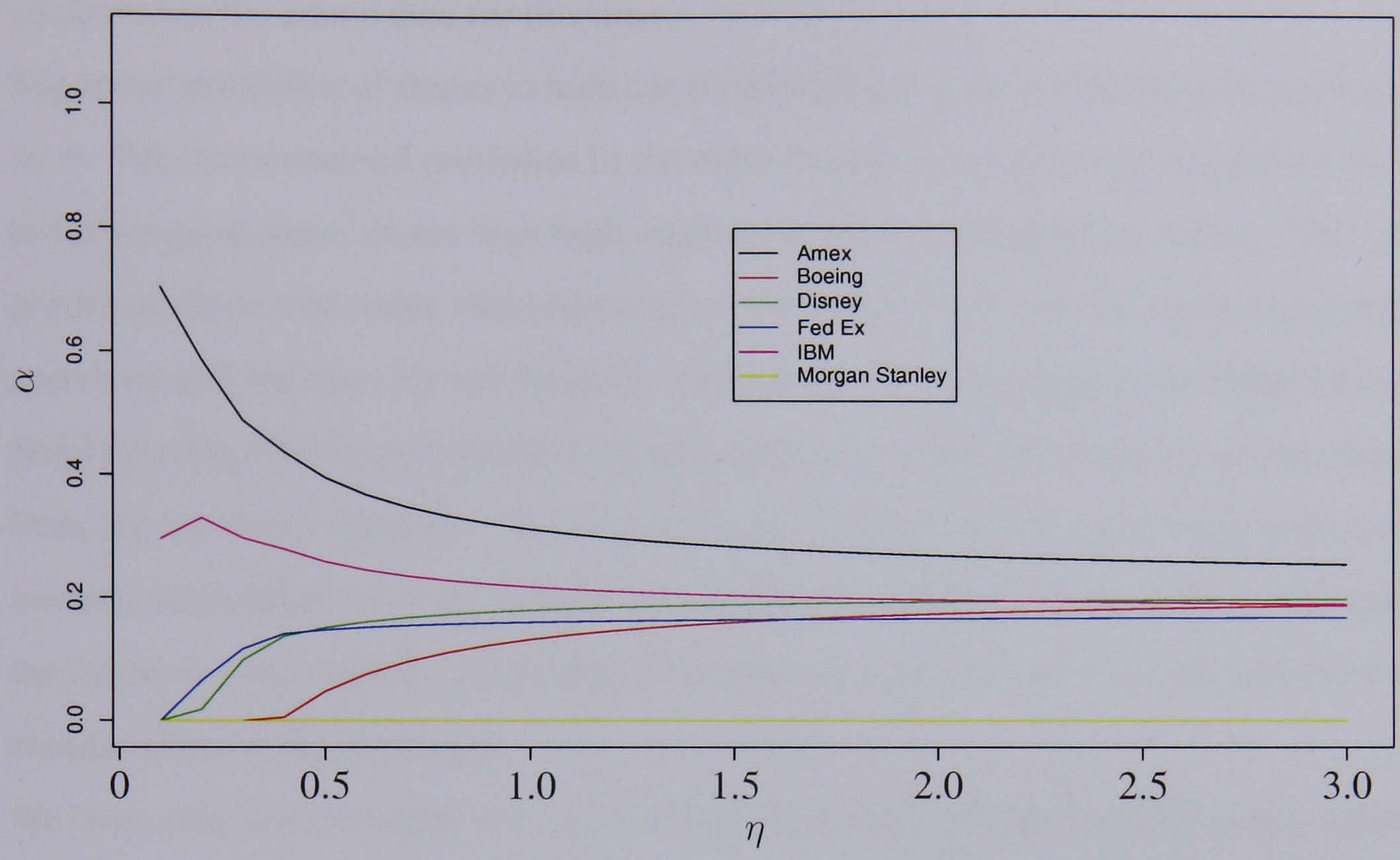


Figure 7.48: Plots of portfolio weights α for varying levels of risk preference η for the 50 step ahead forecast, by solving the constrained quadratic program based on the FSV model.

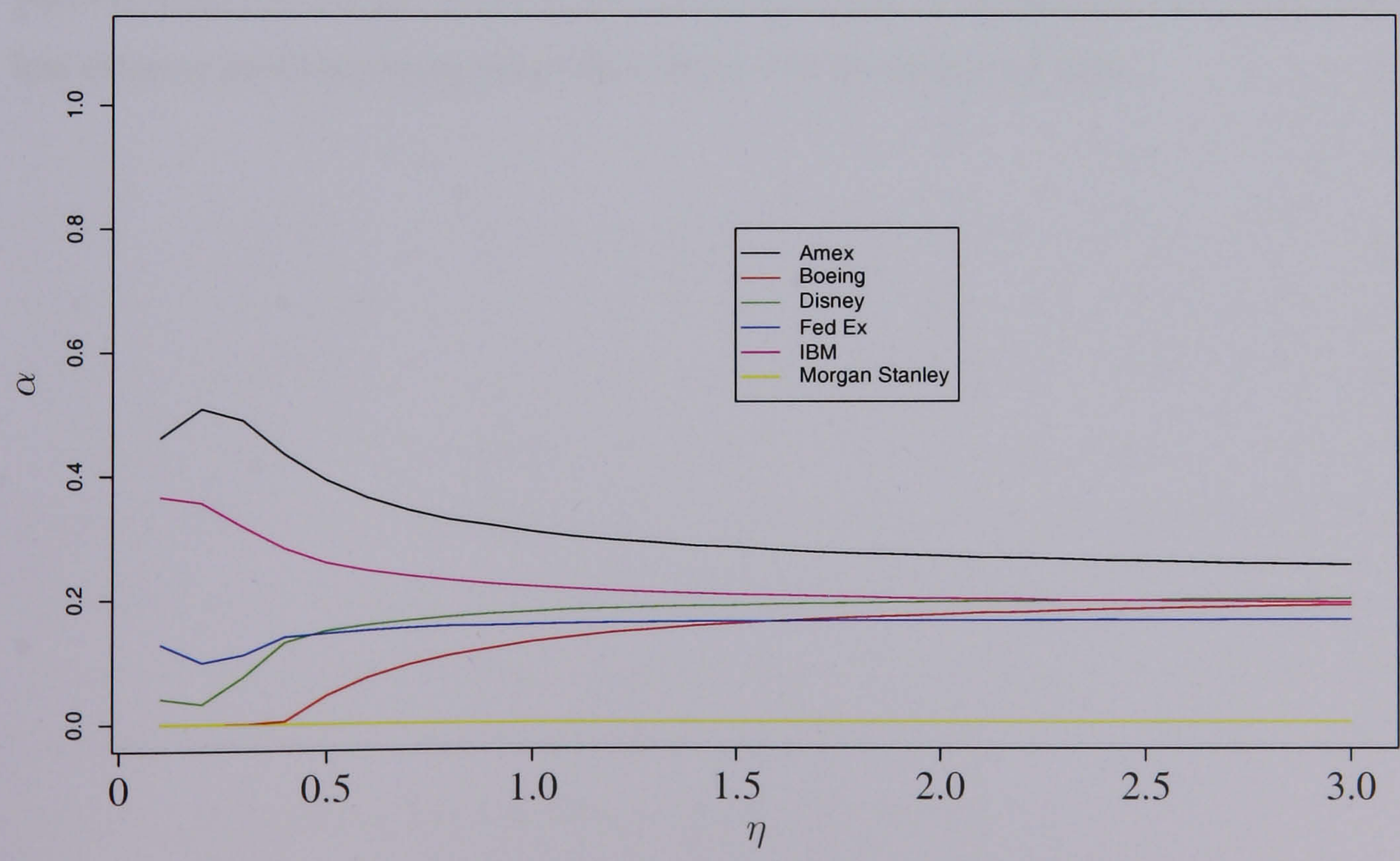


Figure 7.49: Plots of portfolio weights α for varying levels of risk preference η for the 50 step ahead forecast, by stochastic optimisation, utilising the full forecast distribution.

generated using the MCMC method appears to capture market behaviour reasonably well when compared to the actual data for this time.

Suggested portfolios of shares to hold can be derived using the methodologies outlined in Chapter 6. The unconstrained portfolios in the main include long and short positions which attempt to fully exploit those shares with high returns and those with negative returns. The constrained portfolios show noticeable differences when looking at those derived using summary statistics and those derived from the full forecast distribution. This is perhaps to be expected as those derived from the full forecast distribution are based on a greater information set than those derived from the summary statistics. The portfolios also exhibit changes over time, with a movement towards more balanced portfolios the further into the future we look. This is perhaps intuitive, the further we look into the future the less certain we can be of our forecasts and this uncertainty should reflect in the portfolios.

We have also demonstrated we can fit factor stochastic volatility model to the returns of this data. The portfolios derived are different from those suggested based on the dynamic linear models but share the same basic characteristics. Namely extreme long and short positions being taken by risk neutral investors with a tendency towards more balanced portfolios as the investors become more risk averse. There is also a change in the recommended portfolios over time with less extreme positions being taken the further into the future we look.

Chapter 8

Summary and Conclusions

8.1 Summary

In this thesis we have introduced methods whereby which we can model log shares prices and make inference upon these models. We have also considered methods in which an investor can be aided in their decision process.

In Chapter 1 we introduced the Bayesian paradigm as an appropriate one to follow when considering inference on complex models within the context of financial time series. Especially as this paradigm has mechanisms for the incorporation of prior beliefs into our modelling framework and provides probabilistic estimates of the model parameters. We further proposed that the MCMC methodologies which are a major element of modern Bayesian inference could be used to make inference about such models. We also introduced the concept of utility and showed that the comparison of utilities was a rational method making choices between proposed options.

In Chapter 2 we introduced normal dynamic linear models as an appropriate and flexible model structure for the modelling of financial time series. We then demonstrated that Kalman filtering and smoothing techniques could be used to make inference about the states of these models. We derived the Kalman filtering equations using standard normal theory based on the conditional independence structure of the model and also using both normal Bayes and Bayes Linear methods, demonstrating the flexibility of the model structure.

The inference of the states demonstrated in Chapter 2 requires knowledge about the variance matrices that are part of the model specification. In Chapter 3 we showed techniques whereby

we could make inference of these matrices, conditional upon knowing the states. This lead to the development of two algorithms. The first of which, the Expectation-Maximisation algorithm produces estimates for the two variance matrices V and W by maximising the expectation of the log posterior density. This requires the Kalman filter smoother to be run first to provide these expectations. Secondly a block Gibbs sampler which alternates between sampling from the states using a Kalman filter simulation smoother and sampling from the posterior distributions of V and W. The advantage of this latter method is that by using the model structure we can sample from the predictive distribution of the series. This ability to produce a predictive distribution is important in the portfolio selection decision process.

In Chapter 4 we introduced an alternative model structure, that of Stochastic Volatility models. We showed the relationship between these models and the continuous time models used extensively in options pricing. These models can be used to model the returns distribution of financial time series, for example see Pitt and Shephard (1999) and Aguilar and West (2000). Methods for making inference on these were demonstrated which were in part developments of the Kalman filtering and smoothing techniques introduced in Chapter 2.

In Chapter 5 we showed that the dynamic linear models and stochastic volatility models could be combined. Using the factor stochastic volatility models as a more flexible model for the error terms, ν in the dynamic linear model and hence allowing the variance V to evolve through time. Unfortunately this combination of the models leads to identifiability and convergence problems as illustrated in Chapter 5 which require further investigation before these combined models could be used practically in portfolio selection.

Chapter 6 introduced the concept of the portfolio as the appropriate unit of investigation. We compared the traditional approaches to portfolio selection to the radically different approach offered by using the full joint posterior distribution for future share prices produced by the MCMC schemes in Chapter 3. This allows us to develop maximisation schemes which maximise the investor's utility and hence aid them in making portfolio selection decisions.

In Chapter 7 we illustrated some of the techniques introduced in earlier chapters on real world data from the U.S. market.

8.2 Further Work

The examples illustrated in Chapter 7 are of course small compared to real market scenarios. The expansion of these methodologies to larger data sets is likely to prove an interesting area of further research. A direct expansion will of course be computationally expensive but could provide real insight for market practitioners. The model structure also allows the inclusion of market indices or other such indicators within the inference process and of course risk free assets can be included within the maximisation schemes allowing investors to choose some riskless assets within their portfolio. When considering these larger scale models it might prove useful to look at segmentation of the market and the work of Queen, Smith, and James (1994) could provide a means of developing this further.

The other major area for further development will be to look once again at the combined models introduced in Chapter 5. As we have seen convergence and identifiability problems arise in these models which perhaps illustrates the need for strong expert priors in such highly complex hierarchical models.

These are both areas for further development which the author hopes to be able to develop in future research.

Bibliography

Aguilar, O. and M. West (2000). Bayesian dynamic factor models and portfolio selection. *Journal of Business and Economic Statistics 18*, 338–357.

Bachelier, L. (1900). *Théorie de la spéculation*. Ph. D. thesis, Annales de l'Ecole Normale Supérieure.

Best, N., K. Cowles, and S. Vines (1995). *CODA Convergence Diagnostics and Output Analysis softwaree for Gibbs sampling output: Version 0.3*. Cambridge: MRC Biostatictics Unit.

Black, F. and M. Scholes (1973). The pricing of options and corporate liabilities. *The Journal of Political Economy 81*(3), 637–654.

Bollerslev, T. (1986). Generalized autoregressive conditional heteroskedasticity. *Journal of Econometrics 51*, 307–327.

Broad, J. and C. Sutcliffe (1994). Estimation methods in portfolio selection and effectiveness of short sales restrictions:U.K. evidence. *Management Science 40*, 516–534.

Cowles, M. and B. Carlin (1996). Markov Chain Monte Carlo Convergence Diagnostics: A Comparative Review. *Journal of the American Statistical Association 91*, 883–904.

Engle, R. (1982). Autoregressive conditional hetroscedasticity with estimates of the variance of United Kingdom inflation. *Econometrica 50*, 987–1007.

Fama, E. F. (1965). The behavior of stock-market prices. *Jounal of Business 38*, 34–105.

Frost, P. and J. Savarino (1986). An empirical Bayes approach to efficient portfolio selection. *Journal of Financial and Quantitative Analysis 21*(3), 293–305.

Gamerman, D. (1997). *Markov Chain Monte Carlo*. London: Chapman & Hall.

Gelfand, A. and A. Smith (1990). Sampling-based approaches to calculating marginal densities. *Journal of the American Statistical Association 85*, 398–409.

Goldfarb, D. and A. Indnani (1982). *Numerical Analysis*, Chapter Dual and Prinary-Dual Methods for solving strictly convex quadratic programmes, pp. 226–239. Berlin: Springer-Verlag.

Goldstein, M. (1998). *Encyclopaedia of Statistical Sciences*, Volume 3, Chapter Bayes Linear Analysis. Wiley.

Goldstein, M. and D. Wooff (1995). Bayes linear computation: concepts, implementation and programming environment. *Statistics and Computing*, 327–341.

Harvey, A., E. Ruiz, and N. Shephard (1994). Multivariate stochastic variance models. *The Review of Economic Studies 61*(2), 247–264.

Hull, J. (1999). *Options Futures and Other Derivatives* (4 ed.). Prentice Hall.

Jorion, P. (1986). Bayes-Stein estimation for portfolio analysis. *Journal of Financial and Quantitative Analysis 21*, 279–292.

Kim, S., N. Shephard, and S. Chib (1998). Stochastic volatility:likelihood inference and comparison with ARCH models. *Review of Economic Studies 65*, 361–393.

Koopman, S. J. (1993). Disturbance smoother for state space models. *Biometrika 80*(1), 117–126.

Markowitz, H. (1959). *Portfolio Selection: efficient diversification of investment* (second ed.). Oxford: Blackwell Publishers Ltd.

Mengersen, K., C. Robert, and C. Guihenneuc-Jouyaux (1998). MCMC convergence diagnostics: A "RevieWWW". In J. Berger, J. Bernardo, A. Dawid, and A. Smith (Eds.), *Bayesian Statistics 6*, Oxford. OUP.

Merton, R. (1972). An analytical derivation of the efficient portfolio frontier. *Journal of Financial and Quantitative Analysis 7*(4), 1851–1872.

Miller, M. H. (1999, Summer). The history of finance. *The Journal of Portfolio Management*, 95–101.

Novomestky, F. (1997). A dynamic, globally diversified, index neutral synthetc asset allocation strategy. *Management Science 43*(7), 998–1016.

Omohundro, S. and C.-C. Lim (1992, March). The Sather language and libraries. Technical Report TR-92-017, International Computer Science Institute, Berkeley.

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

Pitt, M. and N. Shephard (1999). Time varying covariances: a factor stochastic volatility approach, (with discussions). In J. M. B. et al (Ed.), *Bayesian Statistics 6*, Oxford, pp. 3–26. Oxford University Press.

Pole, A., M. West, and J. Harrison (1994). *Applied Bayesian Forecasting and Time series Analysis*. New York: Chapman and Hall.

Queen, C., J. Smith, and D. James (1994). Bayesian forecasts in markets with overlapping structures. *International Journal of Forecasting* (10), 209–233.

Queen, C. M. (1994). Using the multiregression dynamic model to forecast the brand sales in a competitive market. *The Statistician 43*(1), 87–98.

Quintana, J.-M. (1992). Optimal portfolios of forward currency contracts. In J. Bernardo, J. Berger, A. Dawid, and A. Smith (Eds.), *Bayesian Statistics 4*, Oxford, pp. 753–762. Oxford University Press.

Quintana, J.-M. and M. West (1987). Multivariate time series analysis: new techniques applied to international exchange rate data. *The Statistician 36*, 275–281.

Qunintana, J., V. Lourdes, O. Aguilar, and J. Liu (2003). Global gambling. In *Bayesian Statistics 7*, Oxford. OUP. To Appear.

Sharpe, W. F. (1970). *Porfolio Theory and Capital Markets*. New York: McGraw-Hill.

Watsham, T. (1993). *International Portfolio Management:A Modern Approach* (First ed.). London: Longman.

West, M. and J. Harrison (1997). *Bayesian forecasting and dynamic models* (second ed.). New York: Springer.

Young, M. and P. Lenk (1998). Heirarchical Bayes methods for multifactor model estimation and portfolio selection. *Management Science 44*(11), S111–S124.

Appendix A

Examples of sather Programs

A.1 sather Programs for Kalman Filtering and Smoothing

The **sather** class KALMAN can be found on Darren Wilkinson's software page at http://www.staff.ncl.ac.uk/d.j.wilkinson/software/ .

A.2 sather programs for Parameter Estimation of the Dynamic Linear Model

uemlc.sa a **sather** program for parameter estimation of an univariate locally constant DLM using the EM algorithm.

```
-- uemlc.sa

-- analysis of dataB
-- EM algorithm to estimate V and W

class MAIN is

  -- global class variables
   attr iters,deltaV,deltaW:INT;
   attr m:VECD;
   attr F,G,C,Y,V,W:MATD;

   main is

      F:=#MATD(1,1).ident;
```

```
G:=#MATD(1,1).ident;
m:=#VECD(1);
m[0]:=500.0d;

C:=(#MATD(1,1).ident)*1000.0d;
Y:=KALMAND::matrix_read("newB.txt",1000,1);

-- deliberately choose "bad" V and W to start with
      V:=(#MATD(1,1).ident)*10.0d;
      W:=(#MATD(1,1).ident)*10.0d;
      Vnew::=V*0.5d;
      Wnew::=W*0.5d;


      a::=1.0d;
      b::=(#MATD(1,1).ident)*0.001d;

      -- now run an EM algorithm to estimate the state
      -- and V and W
      loop until!((V-Vnew)[0,0].abs<0.00001d and (W-Wnew)[0,0].abs<0.00001d);
-- filter then smooth
V:=Vnew; W:=Wnew;
kal::=#KALMAND(F,G,V,W,m,C,Y);
kal.filter;
kal.smoother;
 -- calc new estimates of V and W
 Vnew:=(b + ((kal.sum_var_nu+kal.ss_e_nu).times_elt(0.5d))).times_elt(1.0d/(a+(Y.n
 Wnew:=(b+((kal.sum_var_omega+kal.ss_e_omega).times_elt(0.5d))).times_elt(1.0d/(a+

 -- output new estimates

         #OUT+Vnew[0,0]+" "+Wnew[0,0]+"\n";
 -- repeat until convergence to "optimal" V and W
      end;
      #OUT+"EM algorithm has converged to the above posterior mode\n";

   end;

end;

-- end
```

uemll.sa a **sather** program for parameter estimation of an univariate locally linear DLM using the EM algorithm.

```
-- uemll.sa

-- analysis of dataC
-- EM algorithm to estimate V and W
```

```
class MAIN is

  -- global class variables
   attr iters,deltaV,deltaW:INT;
   attr m:VECD;
   attr F,G,C,Y,V,W:MATD;

    main is

        F:=#MATD(2,1).ident;
        G:=#MATD(2,2).ident;
        G[0,1]:=1.0d;
        m:=#VECD(2);
m[0]:=500.0d;
m[1]:=0.01d;
        C:=(#MATD(2,2).ident)*100000.0d;
        Y:=KALMAND::matrix_read("newC.txt",1000,1);

        -- deliberately choose "bad" V and W to start with
        V:=(#MATD(1,1).ident)*1000.0d;
        W:=(#MATD(2,2).ident)*10.0d;
        Vnew::=V*0.5d;
        Wnew::=W*0.5d;
        Wnew[1,1]:=0.01d;


a::= 1.0d;
b::=(#MATD(1,1).ident)*0.0d;

        -- now run an EM algorithm to estimate the state
        -- and V and W
        loop until!((V-Vnew)[0,0].abs<0.00001d and (W-Wnew)[0,0].abs<0.00001d
             and (W -Wnew)[1,1].abs<0.00001d);
 -- filter then smooth
 V:=Vnew; W:=Wnew;
 kal::=#KALMAND(F,G,V,W,m,C,Y);
 kal.filter;
 kal.smoother;
 -- calc new estimates of V and W
 Vnew:=(b + ((kal.sum_var_nu+kal.ss_e_nu).times_elt(0.5d))).
                  times_elt(1.0d/(a+(Y.nr.fltd*0.5d)));
Wnew[0,0]:=(b[0,0]+((kal.sum_var_omega[0,0]+kal.ss_e_omega[0,0])*0.5d))
                     *(1.0d/(a+(Y.nr.fltd*0.5d)));
Wnew[1,1]:=(b[0,0]+((kal.sum_var_omega[1,1]+kal.ss_e_omega[1,1])*0.5d))
                     *(1.0d/(a+(Y.nr.fltd*0.5d)));

  #OUT+Vnew[0,0]+" "+Wnew[0,0]+" "+Wnew[1,1]+"\n";


-- repeat until convergence to "optimal" V and W
        if V[0,0].str = "nan" then raise "NaN ERROR!"; end;
      end;
```

```
        #OUT+"EM algorithm has converged to the above posterior mode\n";
    end;
end;
-- end
```

umclc.sa a **sather** program for parameter estimation of an univariate locally constant DLM using the block Gibbs sampler.

```
-- umclc.sa
--
-- MCMC algorithm for estimating V and W

class MAIN is

 -- global class variables
   attr iters,deltaV,deltaW:INT;
   attr m:VECD;
   attr F,G,C,Y,V,W:MATD;


   main is


      F:=#MATD(1,1).ident;
      G:=#MATD(1,1).ident;
      m:=#VECD(1);
m[0]:=500.0d;
      C:=(#MATD(1,1).ident)*1000.0d;
      Y:=KALMAND::matrix_read("newBtruc.txt",950,1);

      -- deliberately choose "bad" V and W to start with
      V:=(#MATD(1,1).ident)*10.0d;
      W:=(#MATD(1,1).ident)*10.0d;
      Vnew::=V*0.5d;
      Wnew::=W*0.5d;


      n::=50000; -- number of complete iterations of the sampler
      Vsum::=#MATD(1,1);
      Wsum::=#MATD(1,1);

      a::=1.0d;
      b::=0.0d;
```

```
-- now run an MCMC algorithm to estimate V and W
loop i::=1.upto!(n);
#ERR+i+" ";
-- filter then simsmooth
   V:=Vnew; W:=Wnew;
   kal::=#KALMAND(F,G,V,W,m,C,Y);
   kal.sample;
   -- calc new estimates of V and W
   Wnew[0,0]:=1.0d
              /RND::gamma(a+(0.5d*(Y.nr.fltd)),b+(0.5d*((kal.ss_omega)[0,0])))
   Vnew[0,0]:=1.0d
              /RND::gamma(a+(0.5d*(Y.nr.fltd)),b+(0.5d*((kal.ss_nu)[0,0]))));

   -- output new estimates
#OUT+Vnew[0,0]+" "+Wnew[0,0]+"\n";

--forecast state and series

forT::=kal.simforecast_theta(50);
forY::=kal.simforecast_y(50);

#OUT+forT[0][0]+" "+forT[10][0]+""+forT[20][0]+""+forT[30][0]+" "+forT[40][0]+
        " "+forT[49][0]+" "+forY[0][0]+" "+forY[10][0]+""+forY[20][0]+" "+forY[30][
        " "+forY[40][0]+" "+forY[49][0]+     "\n";



        -- keep track of the sums
        Vsum:=Vsum+Vnew;
        Wsum:=Wsum+Wnew;
      end;   --loop

      #ERR+"\nV mean "+(Vsum*(1.0d/n.fltd)).str
     +"\tW mean "+(Wsum*(1.0d/n.fltd)).str+"\n";
   end; -- main

end; -- MAIN

-- end
```

umcll.sa a **sather** program for parameter estimation of an univariate locally linear DLM using the block Gibbs sampler.

```
-- umcll.sa
-- sather code for univariate locally linear mcmc analysis


class MAIN is

   main is
   -- structural matrices
   F::=#MATD(2,1).ident;
   G::=#MATD(2,2).ident;
   G[0,1]:=1.0d;
   m::=#VECD(2);
   C::=(#MATD(2,2).ident)*100000.0d;
   -- data
   Y::=KALMAND::matrix_read("newC.txt",1000,1);
   -- initial values for underlying variance matrices
   V::=(#MATD(1,1).ident)*1000.0d;
   W::=(#MATD(2,2).ident)*10.0d;
   W[1,1]:=0.01d;
   Vsum::=#MATD(1,1);
   Wsum::=#MATD(2,2);
   -- other varaibles
   n::=50000;
   a::= 1.0d;
         b::= 0.001d;

Vnew::=V*0.5d;
Wnew::=W*0.5d;


         -- now run an MCMC algorithm to estimate V and W
         loop i::=1.upto!(n);
           #ERR+i+" ";
           -- filter then simsmooth
           V:=Vnew; W:=Wnew;

           kal::=#KALMAND(F,G,V,W,m,C,Y);

           kal.sample;

         -- calc new estimates of V and W
          Vnew[0,0]:=1.0d
                     /RND::gamma(a+(0.5d*(Y.nr.fltd)),b+(0.5d*((kal.ss_nu)[0,0])));
          Wnew[0,0]:=1.0d
                     /RND::gamma(a+(0.5d*(Y.nr.fltd)),b+(0.5d*((kal.ss_omega)[0,0])))
          Wnew[1,1]:=1.0d
                    /RND::gamma(a+(0.5d*(Y.nr.fltd)),b+(0.5d*((kal.ss_omega)[1,1]) ))


           -- output new estimates
           #OUT+Vnew[0,0]+" "+Wnew[0,0]+"  "+Wnew[1,1]+"\n";
if V[0,0].str = "nan" then raise "NaN Error!"; end;
Vsum:=Vsum+Vnew;
```

```
Wsum:=Wsum+Wnew;
      end;
   #ERR+"\nV mean "+(Vsum*(1.0d/n.fltd)).str
       +"\tW mean "+(Wsum*(1.0d/n.fltd)).str+"\n";


   end; -- main

end; -- MAIN
```

memlc.sa a **sather** program for parameter estimation of a locally constant dynamic linear model using the EM algorithm.

```
-- memlc.sa
-- sather code for Multivariate Locally Constant analysis
-- uses data "newlcY.txt"
--
class MAIN is

   -- global class variables
   attr iters,deltaV,deltaW:INT;
   attr m:VECD;
   attr F,G,C,Y,V,W:MATD;

   main is
      -- structural matrices
      F:=#MATD(4,4).ident;
      G:=#MATD(4,4).ident;
      -- prior for initial state
      m:=#VECD(4);
      C:=(#MATD(4,4).ident)*100000.0d;
      -- data
      Y:=KALMAND::matrix_read("newlcY.txt",1000,4);
      -- initial values for the underlying variance matrices
      V:=(#MATD(4,4).ident)*200.0d;
      W:=(#MATD(4,4).ident)*200.0d;
      -- other misc variables
      iters:=50000;
      deltaV:=5; -- df for inv wishart
      deltaW:=5;

       em_alg

   end; -- main


   em_alg is
```

```
A::=(#MATD(4,4).ident)*0.00001d;
B::=A;

          Vnew::=V; Wnew::=W;
          loop
 V::=Vnew; W::=Wnew;
 kal::=#KALMAND(F,G,V,W,m,C,Y);
 kal.sample;
 kal.smoother;
 Vnew::=(A+(kal.sum_var_nu+kal.ss_e_nu)).times_elt
               (1.0d/(V.nr.fltd+deltaV.fltd+1.0d+Y.nr.fltd));
          Wnew::=(B+(kal.sum_var_omega+kal.ss_e_omega)).times_elt
               (1.0d/(W.nr.fltd+deltaW.fltd+1.0d+Y.nr.fltd));
          -- output new estimates

          loop j::=0.upto!(3); loop k::=0.upto!(3);
               #OUT+V[j,k]+" ";
          end;  end;

          loop j::=0.upto!(3); loop k::=0.upto!(3);
               #OUT+W[j,k]+" ";
          end;  end;

          #OUT+"\n";

          until!((V-Vnew).norm<0.000001d and (W-Wnew).norm<0.000001d);
          if V[0,0].str = "NaN" then raise "NaN ERROR!"; end;
        end; -- loop
     end; -- em_alg


end; -- class MAIN
```

memll.sa a **sather** program for parameter estimation of a locally linear dynamic linear model using the EM algorithm.

```
-- mll.sa
-- sather code for Multivariate Locally Linear analysis
-- uses data "newllY.txt"
--

class MAIN is

   -- global class variables
   attr iters,deltaV,deltaW,deltaZ:INT;
   attr m:VECD;
   attr F,G,C,Y,V,W,Wdash,Z:MATD;

   main is
      -- structural matrices
      F:=#MATD(8,4).ident;
      G:=#MATD(8,8).ident;
```

```
G.inplace_submatrix_to_arg(0,3,4,7,#MATD(4,4).ident);
-- prior for initial state
m:=#VECD(8);
C:=(#MATD(8,8).ident)*10000.0d;
-- data
Y:=KALMAND::matrix_read("newllY.txt",1000,4);
-- initial values for the underlying variance matrices
V:=(#MATD(4,4).ident)*200.0d;
    Wdash:=(#MATD(4,4).ident)*100.0d;
    Z:=(#MATD(4,4).ident)*1.0d;
    W:=#MATD(8,8);
    W.inplace_submatrix_to_arg(0,3,0,3,Wdash);
    W.inplace_submatrix_to_arg(4,7,4,7,Z);
    -- other misc variables
    iters:=50000;
    deltaV:=5; -- df for inv wishart
    deltaW:=5;
    deltaZ:=5;


     em_alg

end; -- main


   em_alg is
A::=(#MATD(4,4).ident)*0.00001d;
B::=A;
CC::=A;

          Vnew::=V; Wnew::=W;
          loop
          V:=Vnew; W:=Wnew;
          kal::=#KALMAND(F,G,V,W,m,C,Y);
          kal.sample;
          kal.smoother;
 Vnew:=(A+kal.sum_var_nu+kal.ss_e_nu).times_elt(1.0d/((deltaV.fltd+Y.nr.fltd+5.0d))
          Wdash:=(B+kal.sum_var_omega.submatrix(0,3,0,3)+
               kal.ss_e_omega.submatrix(0,3,0,3)).times_elt(1.0d/((deltaW.fltd+Y.nr.
 Z:=(CC+kal.sum_var_omega.submatrix(4,7,4,7)+kal.ss_e_omega.submatrix(4,7,4,7)).
              times_elt(1.0d/((deltaZ.fltd+Y.nr.fltd+5.0d)));
 W.inplace_submatrix_to_arg(0,3,0,3,Wdash);
 W.inplace_submatrix_to_arg(4,7,4,7,Z);

-- output new estimates

          loop j::=0.upto!(3); loop k::=0.upto!(3);
          #OUT+V[j,k]+" ";
          end;  end;

          loop j::=0.upto!(7); loop k::=0.upto!(7);
          #OUT+W[j,k]+" ";
          end;  end;
```

```
#OUT+"\n";

until!((V-Vnew).norm + (W-Wnew).norm<0.001d);
if V[0,0].str = "nan" then raise "NaN ERROR!"; end;
     end; -- loop
  end; -- em_alg

end; -- class MAIN
```

mmclc.sa a **sather** program for sampling from the posterior distribution of the parameters of a multivariate locally constant DLM using the block Gibbs sampler.

```
-- mmclc.sa
-- sather code for block gibbs sampling analysis of multivariate,
-- Locally Constant DLM's

class MAIN is

   -- global class variables
   attr iters,deltaV,deltaW:INT;
   attr m:VECD;
   attr F,G,C,Y,V,W:MATD;

   main is
      -- structural matrices
      F:=#MATD(6,6).ident;
      G:=#MATD(6,6).ident;
      -- prior for initial state
      m:=#VECD(6);
      C:=(#MATD(6,6).ident)*1000.0d;
      -- data
      Y:=KALMAND::matrix_read("sharesdat.txt",1009,6);
      -- initial values for the underlying variance matrices
      V:=(#MATD(6,6).ident)*200.0d;
      W:=(#MATD(6,6).ident)*200.0d;
      -- other misc variables
      iters:=250000;
      deltaV:=7; -- df for inv wishart
      deltaW:=7;

      A::=(#MATD(6,6).ident)*0.0000001d;
      B::=A;

-- create headers
```

```
#OUT+"Iter ";
 loop i::=V.row_ind!; #OUT+"V"+i+" "; end;
 loop i::=W.row_ind!; #OUT+"W"+i+" "; end;
 loop k::=1.upto!(4);
 loop i::=1.upto!(10); #OUT+"T"+k+i+" "; end;
 loop i::=1.upto!(10); #OUT+"Y"+k+i+" "; end;
 end; --loop
#OUT+"\n";
loop i::=1.upto!(iters);
    kal::=#KALMAND(F,G,V,W,m,C,Y);
    kal.sample;
    V:=RND2::iwishart(Y.nr+deltaV,(A+kal.ss_nu));
    W:=RND2::iwishart(Y.nr+deltaW,(B+kal.ss_omega));

   -- output new estimates
    #OUT+i+" ";
   loop j::=0.upto!(5); loop k::=0.upto!(5);
   #OUT+V[j,k]+" ";
   end;  end;
   loop j::=0.upto!(5); loop k::=0.upto!(5);
   #OUT+W[j,k]+" ";
   end;  end;

   -- simulate from the forecast distribution

   forT::=kal.simforecast_theta(50);
   forY::=kal.simforecast_y(50);

   loop k::=0.upto!(3);
   loop j::=forT.elt!; #OUT+j[k]+" "; end;
          loop j::=forY.elt!; #OUT+j[k]+" "; end;
          end; --loop


   #OUT+"\n";
   if V[0,0].str = "nan" then raise "NaN ERROR!"; end;
        end; -- loop


     end; --main

end; -- class MAIN
```

mmcll.sa a **sather** program for parameter estimation of an multivariate locally linear DLM using the block Gibbs sampler.

```
-- mmcll.sa
-- sather code for Multivariate Locally Linear analysis
-- uses data newllY.txt
--
```

```
class MAIN is

   -- global class variables
   attr iters,deltaV,deltaW,deltaZ:INT;
   attr m:VECD;
   attr F,G,C,Y,V,W,Wdash,Z:MATD;

      .  .
main is
   -- structural matrices
   F:=#MATD(8,4).ident;
   G:=#MATD(8,8).ident;
   G.inplace_submatrix_to_arg(0,3,4,7,#MATD(4,4).ident);
   -- prior for initial state
   m:=#VECD(8);
   C:=(#MATD(8,8).ident)*100000.0d;
   -- data
   Y:=KALMAND::matrix_read("newllY.txt",1000,4);
   -- initial values for the underlying variance matrices
   V:=(#MATD(4,4).ident)*200.0d;
   Wdash:=(#MATD(4,4).ident)*100.0d;
   Z:=(#MATD(4,4).ident)*1.0d;
   W:=#MATD(8,8);
   W.inplace_submatrix_to_arg(0,3,0,3,Wdash);
   W.inplace_submatrix_to_arg(4,7,4,7,Z);
   -- other misc variables
   iters:=150000;
   deltaV:=5; -- df for inv wishart
   deltaW:=5;
   deltaZ:=8;


   A::=(#MATD(4,4).ident)*0.00001d;
   B::=A;
   CC::=(#MATD(4,4).ident)*0.1d;

-- create headers
   #OUT+"Iter ";
    loop i::=V.row_ind!; #OUT+"V"+i+" "; end;
    loop i::=W.row_ind!; #OUT+"W"+i+" "; end;
    loop i::=1.upto!(10); #OUT+"T"+i+" "; end;
    loop i::=1.upto!(10); #OUT+"Y"+i+" "; end;
   #OUT+"\n";

   loop i::=1.upto!(iters);
   kal::=#KALMAND(F,G,V,W,m,C,Y);
   kal.sample;
   V:=RND2::iwishart(Y.nr+deltaV,(A+kal.ss_nu));
   Wdash:=RND2::iwishart(Y.nr+deltaW,(B+kal.ss_omega.submatrix(0,3,0,3)));
   Z:=RND2::iwishart(Y.nr+deltaZ,(CC+kal.ss_omega.submatrix(4,7,4,7)));
   W.inplace_submatrix_to_arg(0,3,0,3,Wdash);
   W.inplace_submatrix_to_arg(4,7,4,7,Z);

   -- simulate forecast values
```

```
 forT::=kal.simforecast_theta(10);
 forY::=kal.simforecast_y(50);

-- output new estimates
 #OUT+i+" ";
 loop j::=0.upto!(3); loop k::=0.upto!(3);
 #OUT+V[j,k]+" ";
        end;  end;
        loop j::=0.upto!(7); loop k::=0.upto!(7);
        #OUT+W[j,k]+" ";
        end;  end;

        loop j::=forT.elt!; #OUT+j[0]+" "; end;
        loop j::=forY.elt!; #OUT+j[0]+" "; end;
        #OUT+"\n";
        if V[0,0].str = "NaN" then raise "NaN Error!"; end;
       end; -- loop
   end; -- main


end; -- class MAIN
```

A.3 sather programs for Parameter Estimation of Stochastic Volatility Models

The **sather** classes ISV and FSV which perform the perform the sampling from the parameters can be found at Darren Wilkinson's software page at

http://www.staff.ncl.ac.uk/d.j.wilkinson/software/

exampleisv.sa a **sather** program for simulating data from an univariate stochastic volatility model and then making inference on the data.

```
-- example of using the ISV class

-- compile with:
-- sacomp -verbose -output_C -o example_isv example_isv.sa

class MAIN is

   main(argv:ARRAY{STR}) is
      if (argv.asize < 2) then
 raise "Wrong number of args";
```

```
end;
iters ::= argv[1].cursor.get_int;
-- first simulate some data from the model
m,v:FLTD;
n::=2000;
true_phi::=0.9d;
true_sigma_eta::=0.01d;
true_mu::=1.5d;
true_alpha::=#VECD(n);
y::=#VECD(n);
m:=true_mu;
v:=true_sigma_eta*true_sigma_eta/(1.0d-true_phi*true_phi);
true_alpha[0]:=RND2::normal(m,v);
y[0]:=RND2::normal(0.0d,true_alpha[0].exp);
v:=true_sigma_eta*true_sigma_eta;
loop i::=1.upto!(n-1);
 m:=true_mu + true_phi*(true_alpha[i-1]-true_mu);
 true_alpha[i]:=RND2::normal(m,v);
 y[i]:=RND2::normal(0.0d,true_alpha[i].exp);
end; -- loop

#ERR+"true sigma_eta "+true_sigma_eta+"\n";
#ERR+"true mu "+true_mu+"\n";
#ERR+"true phi "+true_phi+"\n";
#ERR+"true alpha[10] "+true_alpha[10]+"\n";
#ERR+"true alpha[100] "+true_alpha[100]+"\n";
 #ERR+"true alpha[300] "+true_alpha[300]+"\n";
#ERR+"true alpha[600] "+true_alpha[600]+"\n";

-- now lets see how well we can recover the
-- true model from the data...

--          phi  sig    mu   y
isv::=#ISV(0.7d,0.015d,1.5d,y); -- initial values

#OUT+"Iter sigma_eta mu phi alpha[10] alpha[100] alpha[300] alpha[600] \n";
loop i::=1.upto!(iters);
 #ERR+i+" ";
 isv.update_alpha(200); -- parameter is block size
 isv.update_sigma_eta(1.0d,0.001d);
 isv.update_mu(0.0d,100.0d);
 -- isv.update_phi(0.9d,0.0025d);
 isv.update_phi_beta(20.0d,3.0d,0.25d);
 #OUT+i+" "
 +isv.sigma_eta+" "
 +isv.mu+" "
 +isv.phi+" "
 +isv.alpha[10]+" "
 +isv.alpha[100]+" "
 +isv.alpha[300]+" "
 +isv.alpha[600]+" "
 +"\n";
end; -- loop
#ERR+"\n";
```

```
   end; -- main
end; -- class MAIN
```

examplefsv.sa a **sather** program for simulating data from an multivariate factor stochastic volatility model and then making inference on the data.

```
-- example_fsv.sa
-- example of using the FSV class
-- compile with:
-- sacomp -verbose -output_C -o example_fsv example_fsv.sa
-- example_fsv 100000 > example_fsv.tab 2> example_fsv.err &

class MAIN is

   main(argv:ARRAY{STR}) is
      -- first grab arg for number of MCMC iterations
      if (argv.asize < 2) then
         raise "Wrong number of args";
      end;
      iters ::= argv[1].cursor.get_int;

      -- simulate some data from a single factor model
      y::=#MATD(2000,4);
      f::=sim_isv(0.8d,0.01d,0.0d,y.nr);
      loop i::=y.col_ind!;
      w::=sim_isv(0.7d,0.02d,i.fltd,y.nr);
      temp::=f.times(1.0d/(i+1).fltd)+w;
      y.inplace_swapped_col(i,temp); -- should be inplace_col!
      end; -- loop i
      -- #OUT+y.str+"\n";


      -- now test out FSV class using this test data
      fsv::=#FSV(0.7d,0.02d,1.5d,0.8d,0.01d,1,y); -- init vals
      #OUT+"Iter phi_f sig_f phi_w0 sig_w0 mu_w0 phi_w1 sig_w1
      mu_w1 phi_w2 sig_w2 mu_w2 phi_w3 sig_w3 mu_w3 b1 b2 b3
      alpha_f_50 alpha_f_500 alpha_w0_50 f_50 f_500 \n";
      loop it::=iters.times!;
         #ERR+it+" ";
         fsv.update_f;
         fsv.update_alpha_f(250); -- parameter is block size
         fsv.update_phi_f_beta(20.0d,3.0d,0.25d);
         fsv.update_sigma_f(1.0d,0.0001d);
         fsv.update_alpha_w(250); -- parameter is block size
         fsv.update_phi_w_beta(20.0d,3.0d,0.25d);
         fsv.update_sigma_w(1.0d,0.0001d);
         fsv.update_mu_w(0.0d,100.0d);
         fsv.update_b(0.0d,100.0d);
```

```
  #OUT+it+" "
+fsv.isvf[0].phi+" "
+fsv.isvf[0].sigma_eta+" "
+fsv.isvw[0].phi+" "
+fsv.isvw[0].sigma_eta+" "
+fsv.isvw[0].mu+" "
+fsv.isvw[1].phi+" "
+fsv.isvw[1].sigma_eta+" "
 +fsv.isvw[1].mu+" "
 +fsv.isvw[2].phi+" "
 +fsv.isvw[2].sigma_eta+" "
 +fsv.isvw[2].mu+" "
 +fsv.isvw[3].phi+" "
 +fsv.isvw[3].sigma_eta+" "
 +fsv.isvw[3].mu+" "
 +fsv.b[1]+" "
 +fsv.b[2]+" "
 +fsv.b[3]+" "
 +fsv.isvf[0].alpha[50]+" "
 +fsv.isvf[0].alpha[500]+" "
 +fsv.isvw[0].alpha[50]+" "
 +fsv.f[50,0]+" "
 +fsv.f[500,0]+" "
 +"\n";
end; -- loop it
    #ERR+"\n";
 end; -- main


  sim_isv(phi,sig,mu:FLTD,n:INT):VECD is
      alpha::=mu;
      res::=#VECD(n);
      loop i::=n.times!;
           alpha:=mu+phi*(alpha-mu)+sig*RND::standard_normal;
           res[i]:=((alpha*0.5d).exp)*RND::standard_normal;
      end; -- loop i
      return res;
  end; -- sim_isv

end; -- class MAIN
```

A.4 sather program for inference on the combined DLM and Stochastic Volatility model

The **sather** program *meanvar-Vfsv.sa* simulates some data from a combined multivariate combined model as described in Chapter 5 and then attempts to make inference on it.

```
-- meanvar-Vfsv.sa
-- example combining kalman filtering / smoothing to sample the state
-- of a series and Factor Stochastic Volatility to sample from the
-- evolving variance, based on a locally constant model
-- compile with
-- sacomp mymodule.module -output_C -o meanvar-new3 meanvar-new3.sa
-------------------------------------------------------------------------
-------------------------------------------------------------------------
------


class MAIN is

 main(argv:ARRAY{STR}) is
    -- first grab arg for number of MCMC iterations
    if (argv.asize < 2) then
       raise "Wrong number of args";
    end;
    iters ::= argv[1].cursor.get_int;  -- number of iterations
    n1 ::= argv[2].cursor.get_int;     -- number of data points
    n2 ::= argv[3].cursor.get_int;     -- number of series


-- structural matrices
    F::=#MATD(n2,n2).ident;
    G::=#MATD(n2,n2).ident;
    -- prior for initial state
    m::=#VECD(n2);
    C::=(#MATD(n2,n2).ident)*100.0d;

-- initial values for the underlying variance matrices
W::=#MATD(n2,n2).ident*0.001d;
V::=#ARRAY{MATD}(n1);
Wnew::=#MATD(n2,n2).ident*0.0015d;
Vnew::=#ARRAY{MATD}(n1);


deltaW::=n2+1; -- df for inv wishart prior
A::=(#MATD(n2,n2).ident)*0.0000001d;; -- part of prior spec on V
```

```
loop i::=0.upto!(n1-1);
     V[i]:=#MATD(4,4).ident*0.1d;
     Vnew[i]:=#MATD(4,4).ident*0.15d;
end;  --loop
```

```
-- simulate some data from a single factor model
-- with fixed variance on the observation equation of the
-- main DLM and FSV on the errors of the State DLM

numat::=#MATD(n1,n2);
dataY::=#MATD(n1,n2);

numat:=sim_fsv(0.8d,0.01d,0.7d,0.02d,1.0d,n1,n2);

w::=#MATD(n2,n2).ident*0.001d;

Theta::=#ARRAY{VECD}(n1);

Theta[0]:=#VECD(|0.2d,0.3d,0.2d,0.4d|);  --initial values(for 4 series)

loop i::=1.upto!(n1-1);
     Theta[i]:=(RND2::mvn(G*Theta[i-1],w));
end; -- loop over i


loop i::=1.upto!(n1-1);
     temp::= F*Theta[i] + numat.row(i);
     dataY.inplace_swapped_row(i,temp);
end; -- loop over i

-- end of data generation



-- output data and states

loop j::=dataY.nr.times!;
     loop k::=dataY.nc.times!;
          #OUT+dataY[j,k]+" ";
     end;
#OUT+"\n";
end;  --loop

loop j::=dataY.nr.times!;
     loop k::=dataY.nc.times!;
          #OUT+Theta[j][k]+" ";
     end;
#OUT+"\n";
end;  --loop
```

```
-- end of output of data and states


-- create headers
      #OUT+"Iter"+" ";

      loop r::=0.upto!(n2-1); loop s::=0.upto!(n2-1);
      #OUT+"V"+r+s+" ";
      end; end; --loops to set up headings for W

      #OUT+"Theta10_0 Theta10_1 Theta10_2 Theta10_3 ";

      #OUT+"phi_f sig_f phi_w0 sig_w0 mu_w0 phi_w1 sig_w1
      mu_w1 phi_w2 sig_w2 mu_w2 phi_w3 sig_w3 mu_w3 b1 b2 b3 ";
      #OUT+"\n";

      kal::=#KALMAND(F,G,V,W,m,C,dataY);
      fsv::=#FSV(0.8d,0.01d,0.0d,0.7d,0.02d,1,numat); -- init vals

      loop i::=1.upto!(iters);
#ERR+i+" ";
V:=Vnew; W:=Wnew;

kal.W:=W;
kal.Varray:=V;
        kal.sample;

Wnew:=RND2::iwishart(dataY.nr+deltaW,(A+kal.ss_omega));


nuvec::=kal.nu;
loop j::=0.upto!(n1-1);
fsv.y.inplace_row(j,nuvec[j]);
end; -- loop to place omegas in matrix format to run FSV

-- update fsv components

fsv.update_f;
fsv.update_alpha_f(250); -- parameter is block size
fsv.update_phi_f_beta(20.0d,3.0d,0.01d);
fsv.update_sigma_f(1.0d,0.0001d);
fsv.update_alpha_w(250); -- parameter is block size
fsv.update_phi_w_beta(20.0d,3.0d,0.25d);
fsv.update_sigma_w(1.0d,0.0001d);
fsv.update_mu_w(1.0d,1.0d);
fsv.update_b(0.0d,1.0d);


-- create new W matrices
```

```
loop k::=0.upto!(n1-1);
   loop l::=0.upto!(n2-1);
      Vnew[k][l,l]:=(fsv.isvw[l].alpha[k]).exp;
   end; -- loop
end; -- loop
```

```
--create new data matrix for kalman filter

loop k2::=0.upto!(n1-1);
   term1::=(dataY.row(k2))-(fsv.b*(fsv.f[k2])).col(0);
   kal.Y.inplace_row(k2,term1);
end; --loop

-- output parameters of interest

#OUT+i+" ";

         loop p::=0.upto!(n2-1);
   loop q::=0.upto!(n2-1);
      #OUT+Wnew[p,q]+" ";
   end; --loop
end; -- loop  to output W
```

```
loop p::=0.upto!(n2-1);
#OUT+kal.theta[10][p]+" ";
end; -- loop

#OUT
      +fsv.isvf[0].phi+" "
      +fsv.isvf[0].sigma_eta+" "
      +fsv.isvw[0].phi+" "
      +fsv.isvw[0].sigma_eta+" "
      +fsv.isvw[0].mu+" "
      +fsv.isvw[1].phi+" "
      +fsv.isvw[1].sigma_eta+" "
      +fsv.isvw[1].mu+" "
      +fsv.isvw[2].phi+" "
      +fsv.isvw[2].sigma_eta+" "
      +fsv.isvw[2].mu+" "
      +fsv.isvw[3].phi+" "
      +fsv.isvw[3].sigma_eta+" "
      +fsv.isvw[3].mu+" "
      +fsv.b[1]+" "
      +fsv.b[1]+" "
      +fsv.b[3]+" "
      +"\n";


   if W[0,0].str = "nan" then raise "ABORT nan detected"; end;

end; -- loop over iters
```

```
 #ERR+"\n";

end; -- main



sim_isv(phi,sig,mu:FLTD,n:INT):VECD is
   alpha::=mu;
   res::=#VECD(n);
   loop i::=n.times!;
        alpha:=mu+phi*(alpha-mu)+sig*RND::standard_normal;
        res[i]:=((alpha*0.5d).exp)*RND::standard_normal;
   end; -- loop i
   return res;
end; -- sim_isv


sim_fsv(phi_f,sig_f,phi_w,sig_w,mu_w:FLTD,n1,n2:INT):MATD is
   res::=#MATD(n1,n2);
   f::=sim_isv(phi_f,sig_f,0.0d,n1);
   loop i::=n2.times!;
        w::=sim_isv(phi_w,sig_w,(mu_w*i.fltd),n1);
        temp::=f.times(1.0d/(i+1).fltd) + w;
        res.inplace_swapped_row(i,temp);
   end; --loop in i
   return res;
end;  -- sim_fsv



forecast_y(alp:ARRAY{VECD},k:INT,kal:KALMAND,V:MATD):ARRAY{VECD} is
   ya::=#ARRAY{VECD}(k);
   n::=kal.Y.nr;
   Wmat::=#MATD(4,4);
   theta::=RND2::mvn(kal.marray[n],kal.Carray[n]);
   loop i::=0.upto!(k-1);
        loop l::=0.upto!(3);
             Wmat[l,l]:= (alp[i][l]);
        end; --loop
        ya[i]:=RND2::mvn(((kal.F).trans)*theta,V);
   end;  -- loop
   return ya;
   end;


end; -- class MAIN
```

A.5 Maximisation Programs

A **R** programs to calculate and plot the solution the the classic Markowitz problem(6.7). These programs require the summary statistic for the returns of the data as an input.

```
markowitzplot_function(mu,sigma)
{
res <- matrix(nrow=300,ncol=4)
for (j in 1:300){
res[j,] <- markowitz(mu,sigma,(j/100))
}
 plot(x_(1:300)/100,res[1:300,1],type="l",col=2,ylim=c(0,0.5),xlab="eta ",ylab="alp
 lines(x_(1:300)/100,res[1:300,2],col=1)
 lines(x_(1:300)/100,res[1:300,3],col=4)
 lines(x_(1:300)/100,res[1:300,4],col=6)

}


markowitz_function(mu, sigma, lambda)
{
    siginv <- solve(sigma)
    ones <- rep(1, 4)
    elem1 <- siginv %*% mu
    elem2 <- siginv %*% ones
    elem3 <- (lambda^-1) * elem1
    elem4 <- 1 - ((lambda^-1) * (ones %*% elem1))
    elem5 <- ones %*% elem2
    elem6 <- (elem4/elem5)[1, 1]
    wstar <- elem3 + ((elem6) * elem2)
    wstar
}
```

A **R** programs to solve the constrained quadratic program (6.9), and output the solutions for plotting. This program require the **R** library **quadprog** to be loaded.

```
noshorts_function(mu,sig)
{
res2 <- matrix(nrow=30,ncol=5)

dvec <- mu
bvec <- c(1,0,0,0,0,0)
Amat0 <- matrix(c(1,1,1,1,1,1,0,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,1),5,6
Dmat

   for (j in 1:30) {
   k <- ((j)/10)
   assign("currentlambda",k,inherits=TRUE)
   Dmat <- currentlambda * sig
```

```
    res <- solve.QP(Dmat,dvec,Amat0,bvec=bvec,meq=1)
    res2[j,] <- res$solution
    }
res2

}
```

The **sather** program *stochoptim.sa* enacts the stochastic optimisation algorithm as described in Section 6.4.5.

```
-- stochastic-optim.sa
-- sather programme to optimise a portfolio
-- based on stochastic simulation

class MAIN is

attr  util,optv,eta,alpha,interim2,variance:FLTD;
attr  w,res:INT;


  main (args:ARRAY{STR}) is

    numshares ::= args[1].cursor.get_int;
    iters::=args[2].cursor.get_int;


opt::=#VECD(numshares);
        opt2::=#VECD(numshares);
        interim::=#VECD(numshares);
        propvec::=#VECD(numshares);
        retv::=#VECD(iters);
        rets::=KALMAND::matrix_read("lc10ret.txt",iters,numshares);

variance:=0.0001d;

  interim2:=1.0d/numshares.fltd;

          loop interim.aset!(interim2);end;


loop
propvec:=vecgen(interim,variance,numshares);

until!(vecok(propvec,numshares) = 0); end; -- loop

        eta:=0.1d;
    retv:=rets.times_vec(propvec);
     util:=utility(iters,eta,retv);
     optv := util;
opt:=interim.copy;
         #OUT+util+"\n";
```

```
loop i::=0.upto!(5000);

  eta:=0.1d;
  loop
  propvec:=vecgen(opt,variance,numshares);
  until!(vecok(propvec,numshares) = 0); end; -- loop
    retv:=rets.times_vec(propvec);
  util:=utility(iters,eta,retv);
   if (util > optv) then
     optv:= util;
     opt:=propvec.copy;
 end; --if
end; --loop


#OUT+eta+" ";
loop j::=0.upto!(numshares-1); #OUT+opt[j]+" "; end; --loop
#OUT+optv+" "+"\n";

loop k::=1.upto!(29);

 optv:=-10.0d*(10.0d^(200.0d));

 eta:=(k.fltd+1.0d)/10.0d;
loop i::=0.upto!(5000);
  loop
 propvec:=vecgen(opt,variance,numshares);

  until!(vecok(propvec,numshares) = 0); end; -- loop
 retv:=rets.times_vec(propvec);
 util:=utility(iters,eta,retv);

 if (util > optv) then
    optv:=util;
    opt:=propvec.copy;

 end; --if
 end; --loop
      #OUT+eta+" ";
loop j::=0.upto!(numshares-1); #OUT+opt[j]+" "; end; --loop
#OUT+optv+" "+"\n";

    end; --loop

end; -- main



utility(iters:INT,eta:FLTD,retv:VECD):FLTD is
    ut::=0.0d;
    e:FLTD;
    loop i::=0.upto!(iters-1);
          e:=retv[i];
```

```
            ut:=ut + (1.0d-((-eta*e).exp));
    end; -- loop
      return ut/iters.fltd;
end; --utility

max(opt:VECD):INT is
    keep::=0.0d;
    index:INT;
    e:FLTD;
     len::=opt.asize;
     loop i::=0.upto!(len-1);
           e := opt[i];
           if (e > keep) then
           keep :=e;
           index:=i;
           end; -- if
     end; -- loop
     return index;
  end; -- max


vecgen(invec:VECD,variance:FLTD,numshares:INT):VECD is
sum::=0.0d;
outvec::=#VECD(numshares);
loop k::=0.upto!(numshares-2);
outvec[k]:= invec[k]+RND2::normal(0.0d,variance);
sum:=sum+outvec[k];
end; -- loop
outvec[numshares-1]:= 1.0d-sum;
return outvec;
end; -- vecgen

vecok(invec:VECD,numshares:INT):INT is
res:=0;
loop k::=0.upto!(numshares-1);
if (invec[k] < 0.0d) and (invec[k] >1.0d) then
res:=1;
end; --if
end; -- loop
return res;
end; -- vecok

end; -- class MAIN
```