4

Models with unknown parameters

In the previous chapters we presented some basic DLMs for time series analysis, assuming that the system matrices F_t, G_t, V_t, and W_t were known. This was done to more easily study their behavior and general properties. In fact, in time series applications the matrices in the DLM are very rarely completely known. In this chapter we let the model matrices depend on a vector of unknown parameters, ψ say. The unknown parameters are usually constant over time, but we also give some examples where they may have a temporal evolution. Anyway, the dynamics of ψ_t will be such as to maintain the linear, Gaussian structure of DLMs.

In a classical framework one typically starts by estimating ψ, usually by maximum likelihood. If the researcher is only interested in the unknown parameters, the analysis terminates here; if, on the other hand, he is interested in smoothing or forecasting the values of the observed series or those of the state vectors, the customary way to proceed is to use the estimated value of ψ as if it were a known constant, and apply the relevant techniques of Chapter 2 for forecasting or smoothing.

From a Bayesian standpoint, unknown parameters are instead random quantities, as we discussed in Chapter 1: therefore, in the context of DLMs, the posterior distribution of interest is the joint conditional distribution of the state vectors—or of future measurements—and the unknown parameter ψ, given the observations. As we shall see, Bayesian inference, even if simple in principle, involves computations that are usually not analytically manageable; however, Markov chain Monte Carlo and modern sequential Monte Carlo methods can be quite efficient in providing an approximation of the posterior distributions of interest.

In Section 4.1 we discuss maximum likelihood estimation of an unknown parameter occurring in the specification of a DLM, while the rest of the chapter is devoted to Bayesian inference.

G. Petris et al., *Dynamic Linear Models with R*, Use R, DOI: 10.1007/b135794_4,

4.1 Maximum likelihood estimation

Suppose that we have n random vectors, $Y_1, \ldots, Y_n$, whose distribution depends on an unknown parameter ψ. We will denote the joint density of the observations for a particular value of the parameter, by $p(y_1, \ldots, y_n; \psi)$. The likelihood function is defined to be, up to a constant factor, the probability density of the observed data read as a function of ψ, i.e., denoting the likelihood by L, we can write $L(\psi) = \text{const.} \cdot p(y_1, \ldots, y_n; \psi)$. For a DLM it is convenient to write the joint density of the observations in the form

$$p(y_1, \ldots, y_n; \psi) = \prod_{t=1}^{n} p(y_t | y_{1:t-1}; \psi), \tag{4.1}$$

where $p(y_t|y_{1:t-1}; \psi)$ is the conditional density of y_t given the data up to time $t-1$, assuming that ψ is the value of the unknown parameter. We know from Chapter 2 that the terms occurring in the RHS of (4.1) are Gaussian densities with mean f_t and variance Q_t. Therefore we can write the loglikelihood as

$$\ell(\psi) = -\frac{1}{2}\sum_{t=1}^{n} \log |Q_t| - \frac{1}{2}\sum_{t=1}^{n} (y_t - f_t)' Q_t^{-1} (y_t - f_t), \tag{4.2}$$

where the f_t and the Q_t depend implicitly on ψ. The expression (4.2) can be numerically maximized to obtain the MLE of ψ:

$$\hat{\psi} = \underset{\psi}{\operatorname{argmax}}\, \ell(\psi). \tag{4.3}$$

Denote by H the Hessian matrix of $-\ell(\psi)$, evaluated at $\psi = \hat{\psi}$. The matrix H^{-1} provides an estimate of the variance of the MLE, $\text{Var}(\hat{\psi})$. Conditions for consistency as well as asymptotic normality of the MLE can be found in Caines (1988) and Hannan and Deistler (1988). See also Shumway and Stoffer (2000) for an introduction. For most of the commonly used DLM, however, the usual consistency and asymptotic normality properties of MLE hold.

A word of caution about numerical optimization is in order. The likelihood function for a DLM may present many local maxima. This implies that starting the optimization routine from different starting points may lead to different maxima. It is therefore a good idea to start the optimizer several times from different starting values and compare the corresponding maxima. A rather flat likelihood is another problem that one may face when looking for a MLE. In this case the optimizer, starting from different initial values, may end up at very different points corresponding to almost the same value of the likelihood. The estimated variance of the MLE will typically be very large. This is a signal that the model is not well identifiable. The solution is usually to simplify the model, eliminating some of the parameters, especially when one is interested in making inference and interpreting the parameters themselves. On the other hand, if smoothing or forecasting is the focus, then

sometimes even a model that is poorly identified in terms of its parameters may produce good results.

R provides an extremely powerful optimizer with the function *optim*, which is used inside the function *dlmMLE* in package dlm. In the optimization world it is customary to minimize functions, and *optim* is no exception: by default it seeks a minimum. Statisticians too, when looking for an MLE, tend to think in terms of minimizing the negative loglikelihood. In line with this point of view, the function *dlmLL* returns the *negative* loglikelihood of a specified DLM for a given data set. In terms of the parameter ψ occurring in the definition of the DLM of interest, one can think of minimizing the compound function obtained in two steps by building a DLM first, and then evaluating its negative loglikelihood, as a function of the matrices defining it. A suggestive graphical representation is the following:

$$\psi \overset{\text{build}}{\Longrightarrow} \text{DLM} \overset{\text{loglik.}}{\Longrightarrow} -\ell(\psi).$$

That is exactly what *dlmMLE* does: it takes a user-defined function *build* that creates a DLM, defines a new function by composing it with *dlmLL*, and passes the result to *optim* for the actual minimization. Consider, for example, the annual precipitation data for Lake Superior (see page 91). By plotting the data, it seems that a polynomial model of order one can provide an adequate description of the phenomenon. The code below shows how to find the MLE of V and W.

R code

```
> y <- ts(read.table("Datasets/lakeSuperior.dat",
+                    skip = 3))[, 2], start = c(1900, 1)
> build <- function(parm) {
+   dlmModPoly(order = 1, dV = exp(parm[1]), dW = exp(parm[2]))
+ }
> fit <- dlmMLE(y, rep(0, 2), build)
> fit$convergence
[1] 0
> unlist(build(fit$par)[c("V", "W")])
        V         W
9.4654447 0.1211534
```

We have parametrized the two unknown variances in terms of their log, so as to avoid problems in case the optimizer went on to examine negative values of the parameters. The value returned by *dlmMLE* is the list returned by the call to *optim*. In particular, the component *convergenge* needs always to be checked: a nonzero value signals that convergence to a minimum has not been achieved. *dlmMLE* has a `...` argument that can be used to provide additional named arguments to *optim*. For example, a call to *optim* including the argument *hessian=TRUE* forces *optim* to return a numerically evaluated

Hessian at the minimum. This can be used to estimate standard errors of the components of the MLE, or more generally its estimated variance matrix, as detailed above. In the previous example we parametrized the model in terms of $\psi = (\log(V), \log(W))$, so that standard errors estimated from the Hessian refer to the MLE of these parameters. In order to get standard errors for the MLE of V and W, one can apply the delta method. Let us recall the general multivariate form of the delta method. Suppose that ψ is h-dimensional, and $g : \mathbb{R}^h \to \mathbb{R}^k$ is a function that has continuous first derivatives. Write $g(\psi) = (g_1(\psi), \ldots, g_k(\psi))$ for any $\psi = (\psi_1, \ldots, \psi_h) \in \mathbb{R}^h$, and define the derivative of g to be the k by h matrix

$$Dg = \begin{bmatrix} \frac{\partial g_1}{\partial \psi_1} & \cdots & \frac{\partial g_1}{\partial \psi_h} \\ \cdots & \cdots & \cdots \\ \frac{\partial g_k}{\partial \psi_1} & \cdots & \frac{\partial g_k}{\partial \psi_h} \end{bmatrix}, \tag{4.4}$$

that is, the ith row of Dg is the gradient of g_i. If $\widehat{\Sigma}$ is the estimated variance matrix of the MLE $\hat{\psi}$, then the MLE of $g(\psi)$ is $g(\hat{\psi})$, and its estimated variance is $Dg(\hat{\psi})\widehat{\Sigma}Dg(\hat{\psi})'$. In the example, $g(\psi) = (\exp(\psi_1), \exp(\psi_2))$, so that

$$Dg(\psi) = \begin{bmatrix} \exp(\psi_1) & 0 \\ 0 & \exp(\psi_2) \end{bmatrix}. \tag{4.5}$$

We can use the Hessian of the negative loglikelihood at the minimum and the delta method to compute in R standard errors of the estimated variances, as the code below shows.

R code

```
> fit <- dlmMLE(y, rep(0, 2), build, hessian = TRUE)
> avarLog <- solve(fit$hessian)
> avar <- diag(exp(fit$par)) %*% avarLog %*%
+   diag(exp(fit$par)) # Delta method
> sqrt(diag(avar)) # estimated standard errors
[1] 1.5059107 0.1032439
```

As an alternative to using the delta method, one can numerically compute the Hessian of the loglikelihood, expressed as a function of the new parameters $g(\psi)$, at $g(\hat{\psi})$. The recommended package **nlme** provides the function **fdHess**, which we put to use in the following piece of code.

R code

```
> avar1 <- solve(fdHess(exp(fit$par), function(x)
+                       dlmLL(y, build(log(x))))$Hessian)
> sqrt(diag(avar1))
[1] 1.5059616 0.1032148 # estimated standard errors
```

In this example one could parametrize the model in terms of V and W, and then use the Hessian returned by *dlmMLE* to compute the estimated standard errors directly. In this case, however, one needs to be careful about the natural restriction of the parameter space, and provide a lower bound for the two variances. Note that the default optimization method, *L-BFGS-B*, is the only method that accepts restrictions on the parameter space, expressed as bounds on the components of the parameter. In the following code, the lower bound 10^{-6} for V reflects the fact that the functions in *dlm* require the matrix V to be nonsingular. On the scale of the data, however, 10^{-6} can be considered zero for all practical purposes.

R code

```
> build <- function(parm) {
+   dlmModPoly(order = 1, dV = parm[1], dW = parm[2])
+ }
> fit <- dlmMLE(y, rep(0.23, 2), build, lower = c(1e-6, 0),
+               hessian = T)
> fit$convergence
[1] 0
> unlist(build(fit$par)[c("V", "W")])
        V         W
9.4654065 0.1211562
> avar <- solve(fit$hessian)
> sqrt(diag(avar))
[1] 1.5059015 0.1032355
```

To conclude, let us mention the function *StructTS*, in base R. This function can be used to find MLE for the variances occurring in some particular univariate DLM. The argument *type* selects the model to use. The available models are the first order polynomial model (*type="level"*), the second order polynomial model (*type="trend"*), and a second order polynomial model plus a seasonal component (*type="BSM"*). Standard errors are not returned by *StructTS*, nor are they easy to compute from its output.

R code

```
> StructTS(y, "level")

Call:
StructTS(x = y, type = "level")

Variances:
  level  epsilon
 0.1212   9.4654
```

4.2 Bayesian inference

The common practice of plugging the MLE $\hat{\psi}$ in the filtering and smoothing recursions suffers from the difficulties in taking properly into account the uncertainty about ψ. The Bayesian approach offers a more consistent formulation of the problem. The unknown parameters ψ are regarded as a *random* vector. The general hypotheses of state space models for the processes (Y_t) and (θ_t) (assumptions (A.1) and (A.2) on page 40) are assumed to hold *conditionally* on the parameters ψ. Prior knowledge about ψ is expressed through a probability law $\pi(\psi)$. Thus, for any $n \geq 1$, we assume that

$$(\theta_0, \theta_1, \ldots, \theta_n, Y_1, \ldots, Y_n, \psi) \sim \pi(\theta_0|\psi)\pi(\psi) \prod_{t=1}^{n} \pi(y_t|\theta_t, \psi)\pi(\theta_t|\theta_{t-1}, \psi) \tag{4.6}$$

(compare with (2.3)).

Given the data $y_{1:t}$, inference on the unknown state θ_s at time s and on the parameters is solved by computing their joint posterior distribution

$$\pi(\theta_s, \psi|y_{1:t}) = \pi(\theta_s|\psi, y_{1:t})\pi(\psi|y_{1:t}), \tag{4.7}$$

where, as usual, one might be interested in $s = t$, in filtering problems; $s > t$, for state prediction; or $s < t$, for smoothing. The marginal conditional density of θ_s is obtained from (4.7); for example, the filtering density is given by

$$\pi(\theta_t|y_{1:t}) = \int \pi(\theta_t|\psi, y_{1:t})\pi(\psi|y_{1:t})d\psi.$$

Here, the recursion formulae for filtering, given in Chapter 2, can be used to compute the conditional density $\pi(\theta_t|\psi, y_{1:t})$; however, this is now averaged with respect to the posterior distribution of ψ, given the data.

Often, one is interested in reconstructing all the unknown state history up to time t; inference on $\theta_{0:t}$ and ψ, given the data $y_{1:t}$, is expressed through their joint posterior density

$$\pi(\theta_{0:t}, \psi|y_{1:t}) = \pi(\theta_{0:t}|\psi, y_{1:t})\pi(\psi|y_{1:t}). \tag{4.8}$$

In principle, the posterior density (4.8) is obtained from the Bayes rule. In some simple models and using conjugate priors, it can be computed in closed form; examples are given in the following section. More often, computations are analytically intractable. However, MCMC methods and sequential Monte Carlo algorithms provide quite efficient tools for approximating the posterior distributions of interest, and this is one reason for the enormous impulse enjoyed by Bayesian inference for state space models in recent years.

Posterior distribution. MCMC and, in particular, Gibbs sampling algorithms are widely used to approximate the joint posterior $\pi(\theta_{0:t}, \psi|y_{1:t})$. Gibbs

sampling from π requires us to iteratively simulate from the full conditional distributions $\pi(\theta_{0:t}|\psi, y_{1:t})$ and $\pi(\psi|\theta_{0:t}, y_{1:t})$. Efficient algorithms for sampling from the full conditional $\pi(\theta_{0:t}|\psi, y_{1:t})$ have been developed, and will be presented in Section 4.4.1. Furthermore, exploiting the conditional independence assumptions of DLMs, the full conditional density $\pi(\psi|\theta_{0:t}, y_{1:t})$ is usually easier to compute than $\pi(\psi|y_{1:t})$. Clearly, this full conditional is problem-specific, but we will provide several examples in the next sections.

We can thus implement Gibbs sampling algorithms to approximate π. Samples from $\pi(\theta_{0:t}, \psi|y_{1:t})$ can also be used for approximating the filtering density $\pi(\theta_t|y_{1:t})$ and the marginal smoothing densities $\pi(\theta_s|y_{1:t}), s < t$. As we shall see, they also allow us to simulate samples from the predictive distribution of the states and observables, $\pi(\theta_{t+1}, y_{t+1}|y_{1:t})$. Thus, this approach solves at the same time the filtering, smoothing, and forecasting problems for a DLM with unknown parameters.

Filtering and on-line forecasting. The shortcoming of the MCMC procedures described above is that they are not designed for recursive or on-line inference. When a new observation y_{t+1} is available, the distribution of interest becomes $\pi(\theta_{0:t+1}, \psi|y_{1:t+1})$ and one has to run a new MCMC all over again to sample from it. This is computationally inefficient, especially in applications that require an *on-line* type of analysis, in which new data arrive rather frequently. As discussed in Chapter 2, one of the attractive properties of DLM is the recursive nature of the filter formulae, which allows us to update the inference efficiently as new data become available. In the case of no unknown parameters in the DLM, one could compute $\pi(\theta_{t+1}|y_{1:t+1})$ from $\pi(\theta_t|y_{1:t})$ by the estimation-error correction formulae given by the Kalman filter, without doing all the computations again. Analogously, when there are unknown parameters ψ, one would like to exploit the samples generated from $\pi(\theta_{0:t}, \psi|y_{1:t})$ in simulating from $\pi(\theta_{0:t+1}, \psi|y_{1:t+1})$, without running the MCMC all over again. Modern sequential Monte Carlo techniques, in particular the family of algorithms that go under the name of *particle filters*, can be used to this aim and allow efficient on-line analysis and simulation-based sequential updating of the posterior distribution of states and unknown parameters. A description of these techniques is postponed until Chapter 5.

4.3 Conjugate Bayesian inference

In some simple cases, Bayesian inference can be carried out in closed form using conjugate priors. We illustrate an example here.

Even in simple structural models such as those presented in Chapter 3, where the system matrices F_t and G_t are known, very rarely are the covariance matrices V_t and W_t completely known. Thus, a basic problem is estimating V_t and W_t. Here we consider a simple case where V_t and W_t are known up to a common scale factor; that is, $V_t = \sigma^2 \tilde{V}_t$ and $W_t = \sigma^2 \tilde{W}_t$, with σ^2 unknown.

This specification of the covariance matrices has been discussed in Section 1.5 of Chapter 1 for the static linear regression model. A classical example is $V_t = \sigma^2 I_m$; an interesting way of specifying $\tilde{W}_t$ is discussed later, using discount factors.

4.3.1 Unknown covariance matrices: conjugate inference

Let $((Y_t, \theta_t) : t = 1, 2, \ldots)$ be described by a DLM with

$$V_t = \sigma^2 \tilde{V}_t, \qquad W_t = \sigma^2 \tilde{W}_t, \qquad C_0 = \sigma^2 \tilde{C}_0. \tag{4.9}$$

Here all the matrices $\tilde{V}_t$, $\tilde{W}_t$, as well as $\tilde{C}_0$ and all the F_t and G_t, are assumed to be known. The scale parameter σ^2, on the other hand, is unknown. As usual in Bayesian inference, it is convenient to work with its inverse $\phi = 1/\sigma^2$. The uncertainty, therefore, is all in the state vectors and in the parameter ϕ. The DLM provides the conditional probability law of (Y_t, θ_t) given ϕ; in particular the model assumes, for any $t \geq 1$,

$$Y_t|\theta_t, \phi \sim \mathcal{N}_m(F_t\theta_t, \phi^{-1}\tilde{V}_t)$$

$$\theta_t|\theta_{t-1}, \phi \sim \mathcal{N}_p(G_t\theta_{t-1}, \phi^{-1}\tilde{W}_t).$$

As the prior for (ϕ, θ_0), a convenient choice is a conjugate Normal-Gamma prior (see Appendix A), that is

$$\phi \sim \mathcal{G}(\alpha_0, \beta_0), \quad \theta_0|\phi \sim \mathcal{N}(m_0, \phi^{-1}\tilde{C}_0),$$

denoted as $(\theta_0, \phi) \sim \mathcal{NG}(m_0, \tilde{C}_0, \alpha_0, \beta_0)$. Then we have the following recursive formulae for filtering. Note the analogy with the recursive formulae valid for a DLM with no unknown parameters.

Proposition 4.1. *For the DLM described above, if*

$$(\theta_{t-1}, \phi)|y_{1:t-1} \sim \mathcal{NG}(m_{t-1}, \tilde{C}_{t-1}, \alpha_{t-1}, \beta_{t-1})$$

where $t \geq 1$, then:

(i) The one-step-ahead predictive density of $(\theta_t, \phi)|y_{1:t-1}$ is Normal-Gamma with parameters $(a_t, \tilde{R}_t, \alpha_{t-1}, \beta_{t-1})$, where

$$a_t = G_t m_{t-1}, \qquad \tilde{R}_t = G_t \tilde{C}_{t-1} G_t' + \tilde{W}_t; \tag{4.10}$$

(ii) The one-step-ahead predictive density of $Y_t|y_{1:t-1}$ is Student-t, with parameters $(f_t, \tilde{Q}_t\beta_{t-1}/\alpha_{t-1}, 2\alpha_{t-1})$, where

$$f_t = F_t a_t, \qquad \tilde{Q}_t = F_t \tilde{R}_t F_t' + \tilde{V}_t; \tag{4.11}$$

(iii) The filtering density of $(\theta_t, \phi|y_{1:t})$ *is Normal-Gamma, with parameters*

$$\begin{aligned} m_t &= a_t + \tilde{R}_t F_t \tilde{Q}^{-1}(y_t - f_t) \\ \tilde{C}_t &= \tilde{R}_t - \tilde{R}_t F_t' \tilde{Q}_t^{-1} \tilde{R}_t', \\ \alpha_t &= \alpha_{t-1} + \frac{m}{2}, \\ \beta_t &= \beta_{t-1} + \frac{1}{2}(y_t - f_t)' \tilde{Q}_t^{-1}(y_t - f_t). \end{aligned} \tag{4.12}$$

Proof. (i) Suppose that $\theta_{t-1}, \phi|y_{1:t-1} \sim \mathcal{NG}(m_{t-1}, \tilde{C}_{t-1}, \alpha_{t-1}, \beta_{t-1})$ (this is true for $t = 1$). In particular, this implies that $\phi|y_{1:t-1} \sim \mathcal{G}(\alpha_{t-1}, \beta_{t-1})$.
By (i) of Proposition 2.2, we have that

$$\theta_t|\phi, y_{1:t-1} \sim \mathcal{N}_p(a_t, \phi^{-1}\tilde{R}_t)$$

with a_t and $\tilde{R}_t$ given by (4.10). Therefore $(\theta_t, \phi)|y_{1:t-1} \sim \mathcal{NG}(a_t, \tilde{R}_t, \alpha_t, \beta_t)$.
(ii) It also follows, from part (ii) of Proposition 2.2, that

$$Y_t|\phi, y_{1:t-1} \sim \mathcal{N}_m(f_t, \phi^{-1}\tilde{Q}_t)$$

where f_t and $\tilde{Q}_t$ are given by (4.11). Thus we obtain that $(Y_t, \phi)|y_{1:t-1} \sim \mathcal{NG}(f_t, \tilde{Q}_t, \alpha_{t-1}, \beta_{t-1})$; the corresponding marginal density of $Y_t|y_{1:t-1}$ is Student-t, as given in (ii).
(iii) For a new observation Y_t, the likelihood is

$$Y_t|\theta_t, \phi \sim \mathcal{N}_m(F_t\theta_t, \phi^{-1}\tilde{V}_t).$$

The theory of linear regression with a Normal-Gamma prior discussed in Chapter 1 (page 21; use (1.11) and (1.12)) applies, and leads to the conclusion that (θ_t, ϕ) given $y_{1:t}$ has again a $\mathcal{NG}(m_t, \tilde{C}_t, \alpha_t, \beta_t)$ distribution, with parameters defined by (4.12). □

By the properties of the Student-t distribution we have that the one-step-ahead forecast of Y_t given $y_{1:t-1}$ is $\mathrm{E}(Y_t|y_{1:t-1}) = f_t$, with covariance matrix $\mathrm{Var}(Y_t|y_{1:t-1}) = \tilde{Q}_t\,\beta_{t-1}/(\alpha_{t-1} - 1)$.

From (iii), the marginal filtering density of $\sigma^2 = \phi^{-1}$ given $y_{1:t}$ is Inverse-Gamma, with parameters (α_t, β_t), so that, for $\alpha_t > 2$,

$$\mathrm{E}(\sigma^2|y_{1:t}) = \frac{\beta_t}{\alpha_t - 1}, \quad \mathrm{Var}(\sigma^2|y_{1:t}) = \frac{\beta_t^2}{(\alpha_t - 1)^2(\alpha_t - 2)}.$$

The marginal filtering density of the states is Student-t:

$$\theta_t|y_{1:t} \sim \mathcal{T}(m_t, \tilde{C}_t\,\beta_t/\alpha_t, 2\alpha_t),$$

with

$$\mathrm{E}(\theta_t|y_{1:t}) = m_t, \quad \mathrm{Var}(\theta_t|y_{1:t}) = \frac{\beta_t}{\alpha_t - 1}\tilde{C}_t. \tag{4.13}$$

Were σ^2 known, the Kalman filter would give the same point estimate, $\mathrm{E}(\theta_t|y_{1:t}) = m_t$, with covariance matrix $\mathrm{Var}(\theta_t|y_{1:t}) = \sigma^2\tilde{C}_t$. Instead, the unknown value of σ^2 is replaced in (4.13) by its conditional expectation $\beta_t/(\alpha_t - 1)$. The larger uncertainty is reflected in a filtering density with thicker tails than the Gaussian density (see Exercise 4.2).

As far as smoothing is concerned, note that

$$(\theta_T, \phi|y_{1:T}) \sim \mathcal{NG}(s_T, \tilde{S}_T, \alpha_T, \beta_T), \tag{4.14}$$

with $s_T = m_T$ and $\tilde{S}_T = \tilde{C}_T$, and write

$$\pi(\theta_t, \phi|y_{1:T}) = \pi(\theta_t|\phi, y_{1:T})\pi(\phi|y_{1:T}), \qquad t = 0, \ldots, T. \tag{4.15}$$

Conditional on ϕ, the Normal theory of Chapter 1 applies, showing that (θ_t, ϕ), conditional on $y_{1:T}$ has a Normal-Gamma distribution. The parameters can be computed using recursive formulae that are the analog of those developed for the Normal case. Namely, for $t = T - 1, \ldots, 0$, let

$$\begin{aligned} s_t &= m_t + \tilde{C}_t G'_{t+1}\tilde{R}^{-1}_{t+1}(s_{t+1} - a_{t+1}), \\ \tilde{S}_t &= \tilde{C}_t - \tilde{C}_t G'_{t+1}\tilde{R}^{-1}_{t+1}(\tilde{R}_{t+1} - \tilde{S}_{t+1})\tilde{R}^{-1}_{t+1}G_{t+1}\tilde{C}_t. \end{aligned}$$

Then

$$\theta_t, \phi|y_{1:T} \sim \mathcal{NG}(s_t, \tilde{S}_t, \alpha_T, \beta_T). \tag{4.16}$$

4.3.2 Specification of W_t by discount factors

The Bayesian conjugate analysis discussed in the previous section applies if the matrices $\tilde{V}_t, \tilde{W}_t$ and $\tilde{C}_0$ are known, which is a restrictive assumption. However, an interesting case is when $\tilde{V}_t = I_m$ and $\tilde{W}_t$ is specified by a technique that is known as *discount factor*. We give here an informal explanation of discount-factors; for an in-depth discussion, see West and Harrison (1997, Section 6.3).

As often underlined (see, e.g., Chapter 2, pages 57-58), the structure and magnitude of the state covariance matrices W_t has a crucial role in determining the role of past observations in state estimation and forecasting. Think of W_t as diagonal, for simplicity. Large values in the diagonal of W_t imply high uncertainty in the state evolution, so that a lot of sample information is lost in passing from θ_{t-1} to θ_t: the past observations $y_{1:t-1}$ give information about θ_{t-1}, which, however, becomes of little relevance in forecasting θ_t; in fact, the current observation y_t is what mainly determines the estimate of $\theta_t|y_{1:t}$. In the Kalman filter recursions, the uncertainty about θ_{t-1} given $y_{1:t-1}$ is summarized in the conditional covariance matrix $\mathrm{Var}(\theta_{t-1}|y_{1:t-1}) = C_{t-1}$. Moving from θ_{t-1} to θ_t via the state equation $\theta_t = G_t\theta_{t-1} + w_t$, the uncertainty

increases and we have $\text{Var}(\theta_t|y_{1:t-1}) = R_t = G_t' C_{t-1} G_t + W_t$. Thus, if $W_t = 0$, i.e., there is no error in the state equation, we have $R_t = \text{Var}(G_t\theta_{t-1}|y_{1:t-1}) = P_t$, say. Otherwise, P_t is increased in $R_t = P_t + W_t$. In this sense, W_t expresses the loss of information in passing from θ_{t-1} to θ_t due to the stochastic error component in the state evolution, the loss depending on the magnitude of W_t with respect to P_t. Therefore, one can think of expressing W_t as a proportion of P_t:

$$W_t = \frac{1-\delta}{\delta} P_t$$

where $\delta \in (0, 1]$. It follows that $R_t = 1/\delta\, P_t$, with $1/\delta > 1$. The parameter δ is called *discount factor* since it "discounts" the matrix P_t that one would have with a deterministic state evolution into the matrix R_t. If $\delta = 1$, $W_t = 0$ and we have no loss of information from θ_{t-1} to θ_t: $\text{Var}(\theta_t|y_{1:t}) = \text{Var}(G_t\theta_{t-1}|y_{1:t-1}) = P_t$. For $\delta < 1$, for example $\delta = 0.8$, we have $\text{Var}(\theta_t|y_{1:t}) = (1/0.8)\ \text{Var}(G_t\theta_{t-1}|y_{1:t-1}) = 1.25 P_t$, showing bigger uncertainty. In practice, the value of the discount factor is usually fixed between 0.9 and 0.99, or it is chosen by model selection diagnostics, e.g., looking at the predictive performance of the model for different values of δ.

The discount factor specification may be used with conjugate priors, that is one can let

$$\tilde{W}_t = \frac{1-\delta}{\delta} G_t' \tilde{C}_{t-1} G_t \tag{4.17}$$

in (4.9). Given $\tilde{C}_0$ and $\tilde{V}_t$ (e.g., $\tilde{V}_t = I_m$), the value of $\tilde{W}_t$ can be recursively computed for every t. The evolution covariance matrix is not time-invariant; however, its dynamics is completely determined once $\tilde{C}_0$, $\tilde{V}_t$ and the system matrices F_t and G_t are given. Further refinements consider different discount factors δ_i for the different components of the state vector.

Example. As a simple illustration, let us consider again the data on the annual precipitations at Lake Superior, which we modeled as a random walk plus noise (see page 145). In Section 4.1, the unknown variances $V_t = \sigma^2$ and $W_t = W$ were estimated by maximum likelihood. Here, we consider Bayesian conjugate inference with $\tilde{W}_t$ specified by the discount factor.

Note that the quantities a_t, f_t, m_t, as well as $\tilde{C}_t$ and $\tilde{R}_t$, can be computed by the Kalman filter for a DLM with known covariance matrices $\tilde{C}_0$ and $\tilde{V}$ (as in (4.9), with $\sigma^2 = 1$) and $\tilde{W}_t$. In fact, the evolution variance $\tilde{W}_t$ is known for $t = 1$, while it is assigned sequentially according to (4.17) for $t > 1$. That is, for each $t = 2, 3, \ldots$, one has to compute $\tilde{W}_t$ from the results obtained at $t-1$, and then use the Kalman filter recursions (4.10)-(4.12). These steps can be easily implemented in R, with a slight modification of the function `dlmFilter`. For the user's convenience, we provide them in the function `dlmFilterDF`, available from the book website. The arguments of `dlmFilterDF` are the data `y`, the model `mod` (where F_t, G_t and V_t are assumed to be time-invariant) and the discount factor `DF`; the function returns, as for `dlmFilter`, the values m_t,

a_t, f_t and the SVDs of $\tilde{C}_t$ and $\tilde{R}_t$ for any t. In addition, it returns the SVDs of the matrices W_t obtained using the discount factor.

The parameters of the filtering distribution for the unknown precision ϕ can be computed from (4.12), which gives

$$\alpha_t = \alpha_0 + \tfrac{t}{2}$$
$$\beta_t = \beta_0 + \tfrac{1}{2}\sum_{i=1}^{t}(y_i - f_i)^2 \tilde{Q}_i^{-1} = \beta_0 + \tfrac{1}{2}\sum_{i=1}^{t}\tilde{e}_i^2,$$

where the standardized innovations $\tilde{e}_t$ and the standard deviation $\tilde{Q}_t^{1/2}$ can be obtained with a call to the `residuals` function. Finally, one can compute

$$Q_t = \mathrm{Var}(Y_t|y_{1:t-1}) = \tilde{Q}_t \frac{\beta_{t-1}}{\alpha_{t-1} - 1}$$
$$C_t = \mathrm{Var}(\theta_t|y_{1:t}) = \tilde{C}_t \frac{\beta_t}{\alpha_t - 1}.$$

As for smoothing, we can compute s_t and $\tilde{S}_t$ with a call to the `dlmSmooth` function. Note that we need, as inputs, the matrices W_t obtained as output of `dlmFilterDF`. Finally, we obtain

$$S_t = \mathrm{Var}(\theta_t|y_{1:T}) = \tilde{S}_t \frac{\beta_T}{\alpha_T - 1}.$$

For illustration with the Lake Superior data, we choose the prior hyperparameters $m_0 = 0$, $\tilde{C}_0 = 10^7$, $\alpha_0 = 2$, $\beta_0 = 20$, so that $\mathrm{E}(\sigma^2) = \beta_0/(\alpha_0 - 1) = 20$, and a discount factor $\delta = 0.9$. Figure 4.1 shows the filtering and smoothing estimates of the level, m_t and s_t, respectively, with 90% probability intervals (see Problem 4.2), as well as the one-step-ahead point forecasts f_t, with 90% probability intervals.

R code

```
> y <- ts(read.table("Datasets/lakeSuperior.dat",
+                    skip = 3)[, 2], start = c(1900, 1))
> mod <- dlmModPoly(1, dV = 1)
> modFilt <- dlmFilterDF(y, mod, DF = 0.9)
> beta0 <- 20; alpha0 <- 2
> ## Filtering estimates
> out <- residuals(modFilt)
> beta <- beta0 + cumsum(out$res^2) / 2
> alpha <- alpha0 + (1 : length(y)) / 2
> Ctilde <- unlist(dlmSvd2var(modFilt$U.C, modFilt$D.C))[-1]
> prob <- 0.95
> tt <- qt(prob, df = 2 * alpha)
> lower <-  dropFirst(modFilt$m) - tt * sqrt(Ctilde * beta /
+                                            alpha)
> upper <-  dropFirst(modFilt$m) + tt * sqrt(Ctilde * beta /
```

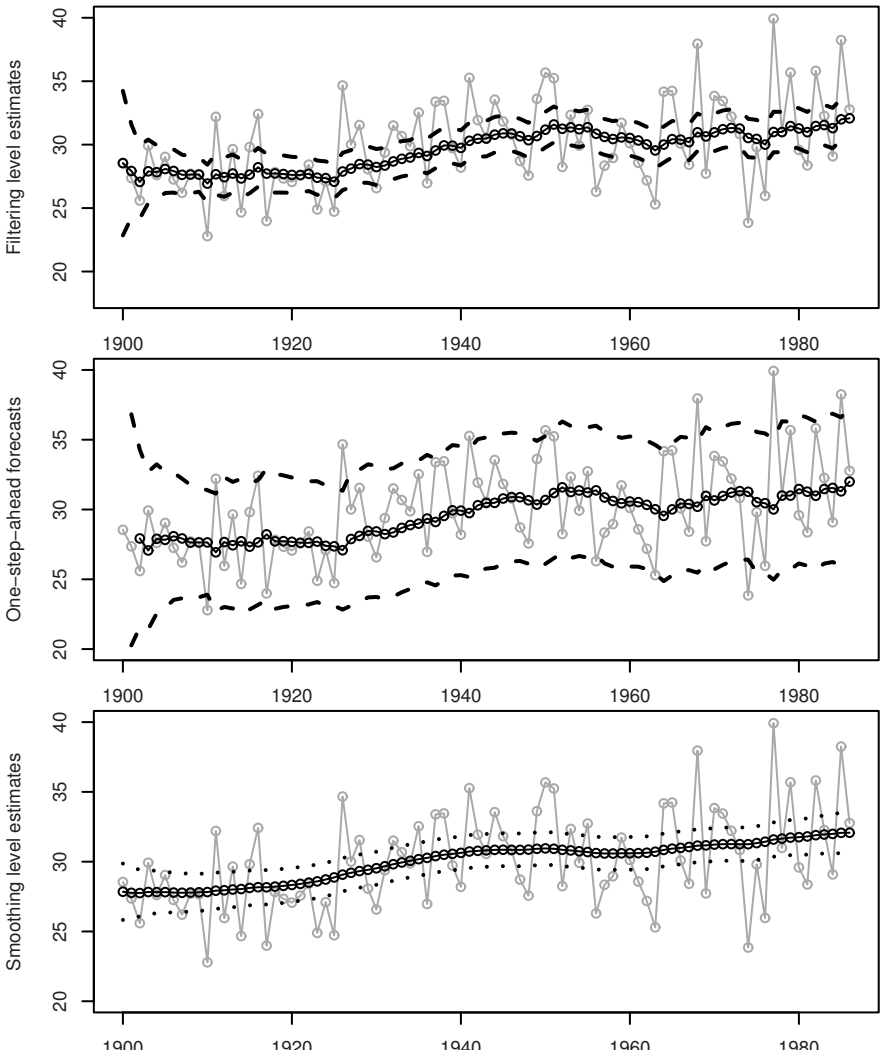

Fig. 4.1. Annual precipitation at Lake Superior (gray), filtering and smoothing level estimates, and one-step-ahead forecasts

```
16  +                                                alpha)
    > plot(y, ylab = "Filtering level estimates", type = "o",
18  +   ylim = c(18, 40), col = "darkgray")
    > lines(dropFirst(modFilt$m), type = "o")
20  > lines(lower, lty = 2, lwd = 2)
    > lines(upper, lty = 2, lwd = 2)
22  > ## One-step-ahead forecasts
```

```
> sigma2 <- c(beta0 / (alpha0-1), beta / (alpha-1))
> Qt <- out$sd^2 * sigma2[-length(sigma2)]
> alpha0T = c(alpha0,alpha)
> tt <- qt(prob, df = 2 * alpha0T[-length(alpha0T)])
> parf <- c(beta0 / alpha0, beta / alpha)
> parf <- parf[-length(parf)] * out$sd^2
> lower <- dropFirst(modFilt$f) - tt * sqrt(parf)
> upper <- dropFirst(modFilt$f) + tt * sqrt(parf)
> plot(y, ylab = "One-step-ahead forecasts", type = "o",
+      ylim = c(20, 40), col = "darkgray")
> lines(window(modFilt$f, start = 1902),  type = "o")
> lines(lower, lty = 2, lwd = 2)
> lines(upper, lty = 2, lwd = 2)
> ## Smoothing estimates
> modFilt$mod$JW <- matrix(1)
> X <- unlist(dlmSvd2var(modFilt$U.W, modFilt$D.W))[-1]
> modFilt$mod$X <- matrix(X)
> modSmooth <- dlmSmooth(modFilt)
> Stildelist <- dlmSvd2var(modSmooth$U.S, modSmooth$D.S)
> TT <- length(y)
> pars <- unlist(Stildelist) * (beta[TT] / alpha[TT])
> tt <- qt(prob, df = 2 * alpha[TT])
> plot(y, ylab = "Smoothing level estimates",
+      type = "o", ylim = c(20, 40), col = "darkgray")
> lines(dropFirst(modSmooth$s),  type = "o")
> lines(dropFirst(modSmooth$s - tt * sqrt(pars)),
+       lty = 3, lwd = 2)
> lines(dropFirst(modSmooth$s + tt * sqrt(pars)),
+       lty = 3, lwd = 2)
```

An important point is choosing the value of the discount factor δ, which determines the signal-to-noise ratio. In practice, one usually tries several values of δ, comparing the predictive performance of the corresponding models. Figure 4.2 shows the one-step-ahead point forecasts for the time window 1960-85, for different choices of δ ($\delta = 1$, which corresponds to the static model with $W_t = 0$ and $\theta_t = \theta_0$; $\delta = 0.9; \delta = 0.8$ and $\delta = 0.3$, a quite small value). The degree of adaptation to new data increases as δ becomes smaller. For $\delta = 0.3$ the forecasts $f_{t+1} = \mathrm{E}(Y_{t+1}|y_{1:t})$ are mainly determined by the current observation y_t. For δ closer to one, smoother forecasts are obtained.

Finally, Figure 4.3 shows the sequence of point estimates $\mathrm{E}(\sigma^2|y_{1:t}) = \beta_t/(\alpha_t - 1)$ of the observation variance, for $t = 1, 2, \ldots, 87$; the final estimates at $t = 87$ are reported in the table below.

discount factor	1.0	0.9	0.8	0.7
$\mathrm{E}(\sigma^2\|y_{1:87})$	12.0010	9.6397	8.9396	8.3601

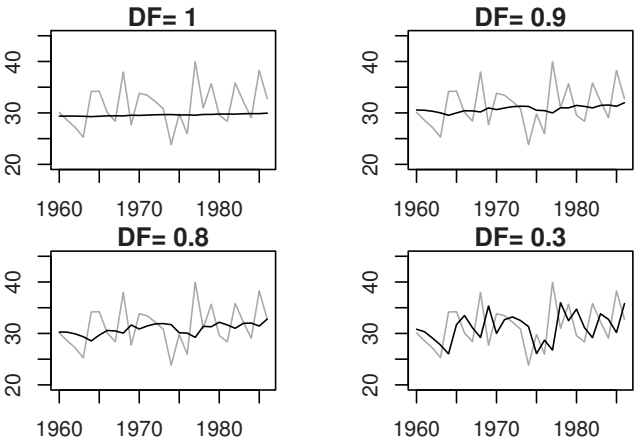

Fig. 4.2. Annual precipitation at Lake Superior (gray) and one-step-ahead forecasts, for different values of the discount factor (DF)

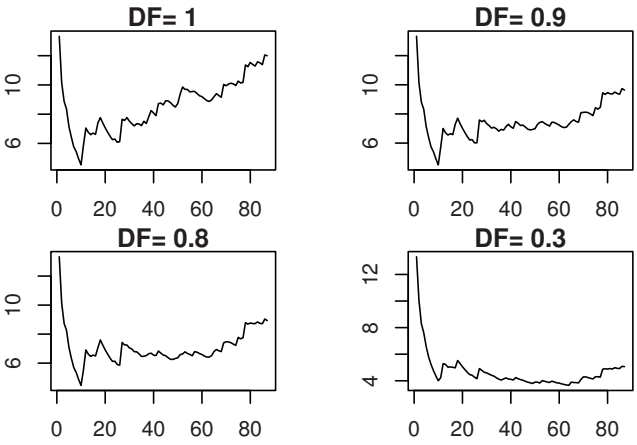

Fig. 4.3. Posterior expectation $\mathrm{E}(\sigma^2|y_{1:t})$, for different values of the discount factor (DF)

Smaller values of δ imply larger evolution variance; correspondingly, a smaller observation variance is expected. Note that, for $\delta = 0.9$, the Bayesian estimate $\mathrm{E}(\sigma^2|y_{1:87}) = 9.6397$ is close to the MLE $\hat{\sigma}^2 = 9.4654$, obtained on page 145.

The table below shows some measures of forecast accuracy for the four different choices of the discount factor, which suggest a better predictive performance for $\delta = 0.9$.

discount factor	MAPE	MAD	MSE
1.0	0.0977	3.0168	21.5395
0.9	0.0946	2.8568	19.9237
0.8	0.0954	2.8706	20.2896
0.3	0.1136	3.4229	25.1182

4.3.3 A discount factor model for time-varying V_t

In the DLM (4.9), the unknown precision factor ϕ is assumed to be constant over time. Since, for simplicity, the components $\tilde{V}_t$ are often taken as time-invariant too, this implies a constant observation covariance matrix, which is a restrictive assumption in many applications. Models for time varying V_t will be discussed in later sections. Here, we apply the technique of discount factors for introducing a fairly simple temporal evolution for the precision ϕ in (4.9) (see West and Harrison; 1997, Section 10.8).

Consider the DLM described in Section 4.3.1, and suppose again that at time $t-1$

$$\phi_{t-1}|y_{1:t-1} \sim \mathcal{G}(\alpha_{t-1}, \beta_{t-1}).$$

However, let now ϕ evolve from time $t-1$ to time t. Consequently, the uncertainty about ϕ_t, given the data $y_{1:t-1}$, will be bigger, that is $\mathrm{Var}(\phi_t|y_{1:t-1}) > \mathrm{Var}(\phi_{t-1}|y_{1:t-1})$. Let us *suppose* for the moment that $\phi_t|y_{1:t-1}$ has still a Gamma density, and in particular suppose that

$$\phi_t|y_{1:t-1} \sim \mathcal{G}(\delta^*\alpha_{t-1}, \delta^*\beta_{t-1}) \,, \tag{4.18}$$

where $0 < \delta^* < 1$. Notice that the expected value is not changed: $\mathrm{E}(\phi_t|y_{1:t-1}) = \mathrm{E}(\phi_{t-1}|y_{1:t-1}) = \alpha_{t-1}/\beta_{t-1}$, while the variance is bigger: $\mathrm{Var}(\phi_t|y_{1:t-1}) = 1/\delta^* \,\mathrm{Var}(\phi_{t-1}|y_{1:t-1})$, with $1/\delta^* > 1$. With this assumption, once a new observation y_t becomes available, one can use the updating formulae of Proposition 4.1, but starting from (4.18) in place of the $\mathcal{G}(\alpha_{t-1}, \beta_{t-1})$. Letting $\alpha_t^* = \delta^*\alpha_{t-1}, \beta_t^* = \delta^*\beta_{t-1}$, we obtain

$$Y_t|y_{1:t-1} \sim \mathcal{T}(f_t, \tilde{Q}_t \frac{\beta_t^*}{\alpha_t^*}, 2\alpha_t^*)$$

and

$$\theta_t|y_{1:t} \sim \mathcal{T}(m_t, \tilde{C}_t \frac{\beta_t}{\alpha_t}, 2\alpha_t)$$

where

$$\alpha_t = \alpha_t^* + \frac{m}{2}, \qquad \beta_t = \beta_t^* + \frac{1}{2}(y_t - f_t)' \tilde{Q}_t^{-1}(y_t - f_t). \tag{4.19}$$

For $m = 1$, the above expressions give

$$\begin{aligned} \alpha_t &= (\delta^*)^t\alpha_0 + \tfrac{1}{2}\textstyle\sum_{i=1}^{t}(\delta^*)^{i-1} \\ \beta_t &= (\delta^*)^t\beta_0 + \tfrac{1}{2}\textstyle\sum_{i=1}^{t}(\delta^*)^{t-i}\tilde{e}_i^2 \end{aligned}$$

where $\tilde{e}_t^2 = (y_t - f_t)^2/\tilde{Q}_t$.

It remains to motivate the assumption (4.18), and in fact we have not specified yet the dynamics that leads from ϕ_{t-1} to ϕ_t. It can be proved that assumption (4.18) is equivalent to the following multiplicative model for the dynamics of ϕ_t

$$\phi_t = \frac{\gamma_t}{\delta^*}\phi_{t-1} \ ,$$

where γ_t is a random variable independent on ϕ_{t-1}, having a beta distribution with parameters $(\delta^*\alpha_{t-1}, (1-\delta^*)\alpha_{t-1})$, so that $\mathrm{E}(\gamma_t) = \delta^*$. Therefore, ϕ_t is equivalent to ϕ_{t-1} multiplied by a random impulse with expected value 1 $(\mathrm{E}(\gamma_t/\delta^*) = 1)$.

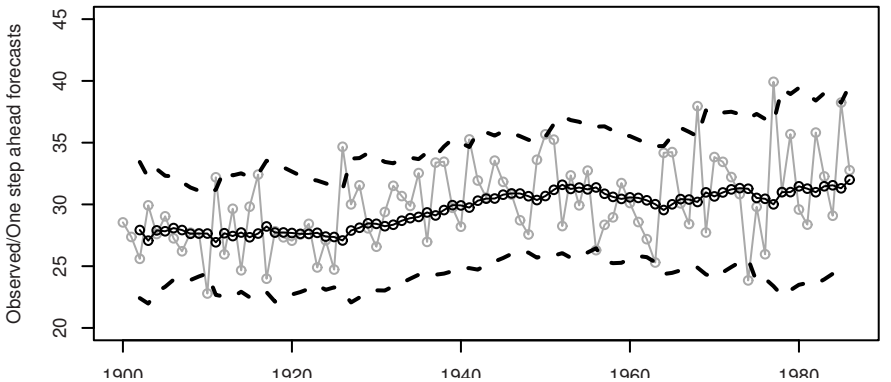

Fig. 4.4. Lake Superior data (gray) and one-step-ahead forecasts, with 90% probability intervals. Both V_t and W_t are specified through discount factors

For illustration with the Lake Superior data, let again $m_0 = E(\theta_0|\phi_0) = 0$, $\tilde{C}_0 = 10^7$, $\alpha_0 = 2$, $\beta_0 = 20$ and use discount factors $\delta = 0.9$, for $\tilde{W}_t$, and $\delta^* = 0.9$, so that $\mathrm{Var}(\phi_t|y_{1:t-1}) = (1/0.9)\mathrm{Var}(\phi_{t-1}|y_{1:t-1})$. Figure 4.4 shows the one-step-ahead point forecasts f_t, with 90% probability intervals.

R code

```
> y <- ts(read.table("Datasets/lakeSuperior.dat",
+                    skip = 3)[, 2], start = c(1900, 1))
> beta0 <- 20; alpha0 <- 2 ; TT <- length(y)
> mod <- dlmModPoly(1, dV = 1)
> modFilt <- dlmFilterDF(y, mod, DF = 0.9)
> out <- residuals(modFilt)
> DFstar <- 0.9
> delta <- DFstar^0 : TT
> alpha <- delta[-1] * alpha0 + cumsum(delta[-TT]) / 2
> res <- as.vector(out$res)
```

```
> beta <- delta[-1] * beta0
> for (i in 1 : TT)
+     beta[i] <- beta[i] + 0.5 * sum(delta[i:1] * res[1:i]^2)
> alphaStar <- DFstar * c(alpha0, alpha)
> betaStar <- DFstar * c(beta0, beta)
> tt <- qt(0.95, df = 2 * alphaStar[1 : TT])
> param <- sqrt(out$sd) * (betaStar / alphaStar)[1 : TT]
> plot(y, ylab = "Observed/One step ahead forecasts",
+      type = "o", col = "darkgray", ylim =c(20, 45))
> lines(window(modFilt$f, start = 1902), type = "o")
> lines(window(modFilt$f, start = 1902) - tt * sqrt(param),
+       lty = 2, lwd = 2)
> lines(window(modFilt$f, start = 1902) + tt * sqrt(param),
+       lty = 2, lwd = 2)
```

4.4 Simulation-based Bayesian inference

For a DLM including a possibly multidimensional unknown parameter ψ in its specification, and observations $y_{1:T}$, the posterior distribution of the parameter and unobservable states is

$$\pi(\psi, \theta_{0:T}|y_{1:T}). \tag{4.20}$$

As mentioned in Section 4.2, in general it is impossible to compute this distribution in closed form. Therefore, in order to come up with posterior summaries one has to resort to numerical methods, almost invariably stochastic, Monte Carlo methods. The customary MCMC approach to analyze the posterior distribution (4.20) is to generate a (dependent) sample from it and evaluate posterior summaries from the simulated sample. The inclusion of the states in the posterior distribution usually simplifies the design of an efficient sampler, even when one is only interested in the posterior distribution of the unknown parameter, $\pi(\psi|y_{1:T})$. In fact, drawing a random variable/vector from $\pi(\psi|\theta_{0:T}, y_{1:T})$ is almost invariably much easier than drawing it from $\pi(\psi|y_{1:T})$; in addition, efficient algorithms to generate the states conditionally on the data and the unknown parameter are available, see Section 4.4.1. This suggests that a sample from (4.20) can be obtained from a Gibbs sampler alternating draws from $\pi(\psi|\theta_{0:T}, y_{1:T})$ and $\pi(\theta_{0:T}|\psi, y_{1:T})$. The simulated sample from the posterior can in turn be used as input to generate a sample from the predictive distribution of states and observables, $\pi(\theta_{T+1:T+k}, y_{T+1:T+k}|y_{1;T})$. In fact,

$$\pi(\theta_{T+1:T+k}, y_{T+1:T+k}, \psi, \theta_T|y_{1;T}) =$$
$$\pi(\theta_{T+1:T+k}, y_{T+1:T+k}|\psi, \theta_T) \cdot \pi(\psi, \theta_T|y_{1;T}).$$

Therefore, for every pair (ψ, θ_T) drawn from $\pi(\psi, \theta_T|y_{1:T})$, one can generate the "future" $\theta_{T+1:T+k}, y_{T+1:T+k}$ from $\pi(\theta_{T+1:T+k}, y_{T+1:T+k}|\psi, \theta_T)$ (see Section 2.8) to obtain a sample from the predictive distribution.

The approach sketched above completely solves the filtering, smoothing, and forecasting problems for a DLM with unknown parameters. However, if one needs to update the posterior distribution after one or more new observations become available, then one has to run the Gibbs sampler all over again, and this can be extremely inefficient. As already mentioned, on-line analysis and simulation-based sequential updating of the posterior distribution of states and unknown parameters are best dealt with employing sequential Monte Carlo techniques.

4.4.1 Drawing the states given $y_{1:T}$: forward filtering backward sampling

In a Gibbs sampling from $\pi(\theta_{0:T}, \psi|y_{1:T})$, one needs to simulate from the full conditional densities $\pi(\psi|\theta_{0:T}, y_{1:T})$ and $\pi(\theta_{0:T}|\psi, y_{1:T})$. While the first density is problem specific, the general expression of the latter density and efficient algorithms for sampling from it are available.

The smoothing recursions provide an algorithm for computing the mean and variance of the distribution of θ_t conditional on $y_{1:T}$ and ψ ($t = 0, 1, \ldots, T$). Since all the involved distributions are Normal, this completely determines the *marginal* posterior distribution of θ_t given $y_{0:T}$ and ψ. If one is interested in the joint posterior distribution of $\theta_{0:T}$ given $(y_{1:T}, \psi)$, then also the posterior covariances between θ_t and θ_s have to be computed. General formulae to recursively evaluate these covariances are available, see Durbin and Koopman (2001). However, when $\pi(\theta_{0:T}|\psi, y_{1:T})$ has the role of full conditional in a Gibbs sampling from $\pi(\theta_{0:T}, \psi|y_{1:T})$, the main question becomes: how can one generate a draw from the distribution of $\theta_{0:T}$ given $(y_{1:T}, \psi)$? We will use a method due to Carter and Kohn (1994), Früwirth-Schnatter (1994), and Shephard (1994), which is now widely known as the forward filtering backward sampling (FFBS) algorithm. By reading the description that follows, the reader will realize that FFBS is essentially a simulation version of the smoothing recursions.

We can write the joint distribution of $\theta_{0:T}$ given $y_{1:T}$ as

$$\pi(\theta_{0:T}|y_{1:T}) = \prod_{t=0}^{T} \pi(\theta_t|\theta_{t+1:T}, y_{1:T}), \tag{4.21}$$

where the last factor in the product is simply $\pi(\theta_T|y_{1:T})$, i.e., the filtering distribution of θ_T, which is $\mathcal{N}(m_T, C_T)$. Formula (4.21) suggests that in order to obtain a draw from the distribution on the left-hand side, one can start by drawing θ_T from a $\mathcal{N}(m_T, C_T)$ and then, for $t = T-1, T-2, \ldots, 0$, recursively draw θ_t from $\pi(\theta_t|\theta_{t+1:T}, y_{1:T})$. We have seen in the proof of Proposition 2.4

that $\pi(\theta_t|\theta_{t+1:T}, y_{1:T}) = \pi(\theta_t|\theta_{t+1}, y_{1:t})$, and we showed that this distribution is $\mathcal{N}(h_t, H_t)$, with

$$h_t = m_t + C_t G'_{t+1} R^{-1}_{t+1}(\theta_{t+1} - a_{t+1}),$$
$$H_t = C_t - C_t G'_{t+1} R^{-1}_{t+1} G_{t+1} C_t.$$

Therefore, having already $(\theta_{t+1}, \ldots, \theta_T)$, the next step consists in drawing θ_t from a $\mathcal{N}(h_t, H_t)$. Note that h_t explicitly depends on the value of θ_{t+1} already generated. The FFBS algorithm is summarized in Algorithm 4.1

1. Run Kalman filter.
2. Draw $\theta_T \sim \mathcal{N}(m_T, C_T)$.
3. For $t = T-1, \ldots, 0$, draw $\theta_t \sim \mathcal{N}(h_t, H_t)$.

Algorithm 4.1: Forward filtering backward sampling

FFBS is commonly used as a building block of a Gibbs sampler, as we will illustrate in many examples in the remaining of the chapter, in particular in Section 4.5. However, it can be of interest also in DLMs containing no unknown parameters. In this case, the marginal smoothing distribution of each θ_t is usually enough to evaluate posterior probabilities of interest. However, the posterior distribution of a nonlinear function of the states may be difficult or impossible to derive, even when all the parameters of the model are known. In this case FFBS provides an easy way to generate an independent sample from the posterior of the nonlinear function of interest. Note that in this type of application the "forward filtering" part of the algorithm only needs to be performed once.

4.4.2 General strategies for MCMC

For a completely specified DLM, i.e., one not containing unknown parameters, draws from the posterior distribution of the states, and possibly from the forecast distribution of states and observations, can be obtained using the FFBS algorithm described in the previous subsection. In the more realistic situation of a DLM containing an unknown parameter vector ψ, with prior distribution $\pi(\psi)$, in the observation, system, or variance matrices, one typically uses MCMC to obtain posterior summaries of the distributions of interest. Almost all Markov chain samplers for posterior analysis of a DLM fall in one of the following categories: Gibbs samplers, which include the states as latent variables; marginal samplers; and hybrid samplers, which combine aspects of both. Note that, depending on the context, the analyst may be interested in making inference about the unobservable states, the unknown parameter, or both. Of the three types of samplers, two (Gibbs and hybrid

samplers) generate draws from the joint posterior of the states and the parameter, while the other (marginal samplers) only generates draws from the posterior of the parameter. Keep in mind, however, that once a sample from the posterior distribution of the parameter is available, a sample from the joint posterior of states and parameter can be easily obtained in view of the decomposition

$$\pi(\theta_{0:T}, \psi | y_{1:T}) = \pi(\theta_{0:T} | \psi, y_{1:T}) \cdot \pi(\psi | y_{1:T}).$$

More specifically, for each $\psi^{(i)}$ in the sample ($i = 1, \ldots, N$), it is enough to draw $\theta_{0:T}^{(i)}$ from $\pi(\theta_{0:T} | \psi = \psi^{(i)}, y_{1:T})$ using FFBS, and $\{(\theta_{0:T}^{(i)}, \psi^{(i)}) : i = 1, \ldots, N\}$ will be the required sample from the joint posterior distribution.

The Gibbs sampling approach, consisting in drawing in turn the states from their conditional distribution given the parameter and observations, and the parameter from its conditional distribution given the states and observations, is summarized in Algorithm 4.2. Package dlm provides the function

0. Initialize: set $\psi = \psi^{(0)}$.
1. For $i = 1, \ldots, N$:
   a) Draw $\theta_{0:T}^{(i)}$ from $\pi(\theta_{0:T} | y_{1:T}, \psi = \psi^{(i-1)})$ using FFBS.
   b) Draw $\psi^{(i)}$ from $\pi(\psi | y_{1:T}, \theta_{0:T} = \theta_{0:T}^{(i)})$.

Algorithm 4.2: Forward Filtering Backward Sampling in a Gibbs sampler

dlmBSample which, in conjunction with *dlmFilter*, can be used to perform step (a). Step (b), on the other hand, depends heavily on the model under consideration, including the prior distribution on ψ. In fact, when ψ is an r-dimensional vector, it is often simpler to perform a Gibbs step for each component of ψ instead of drawing ψ at once. The intermediate approach of drawing blocks of components of ψ together is another option. In any case, when a full conditional distribution is difficult to sample from, a Metropolis–Hastings step can replace the corresponding Gibbs step. A generic sampler for nonstandard distributions is ARMS (Section 1.6), available in package dlm in the function *arms*.

The second approach, marginal sampling, is conceptually straightforward, consisting of drawing a sample from $\pi(\psi | y_{0:T})$. The actual implementation of the sampler depends on the model under consideration. Typically, if ψ is multivariate, one can use a Gibbs sampler, drawing each component, or block of components, from its full conditional distributions, possibly using a Metropolis–Hastings step when the relevant full conditional is not a standard distribution. Again, ARMS can be used in the latter case.

A hybrid sampler can be used when the parameter can be decomposed in two components, that is, when ψ can be written as (ψ_1, ψ_2), where each component may be univariate or multivariate. Algorithm 4.3 gives an algo-

0. Initialize: set $\psi_2 = \psi_2^{(0)}$.
1. For $i = 1, \ldots, N$:
   a) Draw $\psi_1^{(i)}$ from $\pi(\psi_1 | y_{0:T}, \psi_2 = \psi_2^{(i-1)})$.
   b) Draw $\theta_{0:T}^{(i)}$ from $\pi(\theta_{0:T} | y_{1:T}, \psi_1 = \psi_1^{(i)}, \psi_2 = \psi_2^{(i-1)})$ using FFBS.
   c) Draw $\psi_2^{(i)}$ from $\pi(\psi_2 | y_{1:T}, \theta_{0:T} = \theta_{0:T}^{(i)}, \psi_1 = \psi_1^{(i)})$.

Algorithm 4.3: Forward Filtering Backward Sampling in a hybrid sampler

rithmic description of a generic hybrid sampler. As for the previous schemes, Metropolis–Hastings steps, and ARMS in particular, can be substituted for direct sampling in steps (a) and (c). Step (b) can always be performed using `dlmFilter` followed by `dlmBSample`. For the theoretically inclined reader, let us point out a subtle difference between this sampler and a Gibbs sampler. In a Gibbs sampler, each step consists of applying a Markov transition kernel whose invariant distribution is the target distribution, so that the latter is also invariant for the composition of all the kernels. In a hybrid sampler, on the other hand, the target distribution is not invariant for the Markov kernel corresponding to step (a), so the previous argument does not apply directly. However, it is not difficult to show that the composition of step (a) and (b) does preserve the target distribution and so, when combined with step (c), which is a standard Gibbs step, it produces a Markov kernel having the correct invariant distribution. An example of Bayesian inference with hybrid sampling will be given in Section 4.6.1.

The output produced by a Markov chain sampler must always be checked to assess convergence to the stationary distribution and mixing of the chain. Given that the chain has practically reached the stationary distribution, mixing can be assessed by looking at autocorrelation functions of parameters or functions of interest. Ideally, one would like to have as low a correlation as possible between draws. Correlation may be reduced by *thinning* the simulated chain, i.e., discarding a fixed number of iterations between every saved value. Although this method is very easy to implement, the improvements are usually only marginal, unless the number of discarded simulations is substantial, which significantly increases the time required to run the entire sampler. As far as assessing convergence, a fairly extensive body of literature exists on diagnostic tools for MCMC. In R the package BOA provides a suite of functions implementing many of these diagnostics. In most cases, a visual inspection of the output, after discarding a first part of the simulated chain as *burn in*, can reveal obvious departures from stationarity. For an organic treatment of MCMC diagnostics, we refer to Robert and Casella (2004) and references therein.

4.4.3 Illustration: Gibbs sampling for a local level model

Before moving on to more general models, we give in the present section a simple illustration of how a Gibbs sampler can be implemented in practice. The example that we are going to examine is the local level model with unknown observation and evolution variances. A convenient, yet flexible, family of priors for each variance is the inverse-gamma family. More specifically, let $\psi_1 = V^{-1}$ and $\psi_2 = W^{-1}$, and assume that ψ_1 and ψ_2 are a priori independent, with

$$\psi_i \sim \mathcal{G}(a_i, b_i), \qquad i = 1, 2.$$

The parameters of the prior, $a_i, b_i\,(i = 1, 2)$ can be specified so as to match the prior opinion of the analyst about the unknown precisions, expressed in terms of their means and variances. Most people find it easier to specify prior moments (mean and variance) of an unknown variance than it is to specify those of a precision. In this case, just recall that a $\mathcal{G}(a, b)$ prior for ψ_i is the same as an Inverse Gamma prior with the same parameters for ψ_i^{-1}.

A Gibbs sampler for this model will iterate the following three steps: draw $\theta_{0:T}$, draw ψ_1, and draw ψ_2. The random quantities generated in each of the three steps should be drawn from their *full conditional* distribution, i.e., the distribution of that quantity given all the other random variables in the model, including the observations. Note that the states $\theta_{0:T}$ must be included among the conditioning variables when drawing ψ_1 and ψ_2. To generate $\theta_{0:T}$ we can use the FFBS algorithm, setting ψ_1 and ψ_2 to their most recently simulated value. This is something that can be done in a straightforward way using the function `dlmBSample`. On the other hand, a simple calculation is needed to determine the full conditional distribution of ψ_1. The standard approach is based on the fact that the full conditional distribution is proportional to the joint distribution of all the random variables considered. For ψ_1, this is

$$\begin{aligned}
\pi(\psi_1|\psi_2, \theta_{0:T}, y_{1:T}) &\propto \pi(\psi_1, \psi_2, \theta_{0:T}, y_{1:T}) \\
&\propto \pi(y_{1:T}|\theta_{0:T}, \psi_1, \psi_2)\pi(\theta_{0:T}|\psi_1, \psi_2)\pi(\psi_1, \psi_2) \\
&\propto \prod_{t=1}^{T} \pi(y_t|\theta_t, \psi_1) \prod_{t=1}^{T} \pi(\theta_t|\theta_{t-1}, \psi_2)\pi(\psi_1)\pi(\psi_2) \\
&\propto \pi(\psi_1)\psi_1^{T/2} \exp\left(-\frac{\psi_1}{2}\sum_{t=1}^{T}(y_t - \theta_t)^2\right) \\
&\propto \psi_1^{a_1+T/2-1} \exp\left(-\psi_1 \cdot \left[b_1 + \frac{1}{2}\sum_{t=1}^{T}(y_t - \theta_t)^2\right]\right).
\end{aligned}$$

From the previous equations we deduce that ψ_1 and ψ_2 are conditionally independent, given $\theta_{0:T}$ and $y_{1:T}$, and

$$\psi_1|\theta_{0:T}, y_{1:T} \sim \mathcal{G}\left(a_1 + \frac{T}{2}, b_1 + \frac{1}{2}\sum_{t=1}^{T}(y_t - \theta_t)^2\right).$$

A similar argument shows that

$$\psi_2|\theta_{0:T}, y_{1:T} \sim \mathcal{G}\left(a_2 + \frac{T}{2}, b_2 + \frac{1}{2}\sum_{t=1}^{T}(\theta_t - \theta_{t-1})^2\right).$$

The following code implements the Gibbs sampler discussed above for the Nile River data. The actual sampler consists of the loop starting on line 18; what comes before is just book keeping, space allocation for the output, and the definition of starting values and variables that do not change within the main loop.

R code

```
> a1 <- 2
> b1 <- 0.0001
> a2 <- 2
> b2 <- 0.0001
> ## starting values
> psi1 <- 1
> psi2 <- 1
> mod_level <- dlmModPoly(1, dV = 1 / psi1, dW = 1 / psi2)
>
> mc <- 1500
> psi1_save <- numeric(mc)
> psi2_save <- numeric(mc)
> n <- length(Nile)
> sh1 <- a1 + n / 2
> sh2 <- a2 + n / 2
> set.seed(10)
>
> for (it in 1 : mc)
+ {
+     ## draw the states: FFBS
+     filt <- dlmFilter(Nile, mod_level)
+     level <- dlmBSample(filt)
+     ## draw observation precision psi1
+     rate <- b1 + crossprod(Nile - level[-1]) / 2
+     psi1 <- rgamma(1, shape = sh1, rate = rate)
+     ## draw system precision psi2
+     rate <- b2 + crossprod(level[-1] - level[-n]) / 2
+     psi2 <- rgamma(1, shape = sh2, rate = rate)
+     ## update and save
+     V(mod_level) <- 1 / psi1
```

```
+     W[mod_level] <- 1 / psi2
+     psi1_save[it] <- psi1
+     psi2_save[it] <- psi2
+ }
```

A visual analysis of trace plots (not shown) shows that the chain reached convergence after the first two or three hundred iterations. In the code we did not save the simulated states, assuming the focus of the analysis was on the unknown variances. Note, however, that a sample from the posterior distribution of the states does not require us to run the Gibbs sampler all over again, as one can just run the FFBS algorithm once for each simulated value of (ψ_1, ψ_2), thanks to the equality

$$\pi(\theta_{0:T}, \psi_1, \psi_2 | y_{1:T}) = \pi(\theta_{0:T} | \psi_1, \psi_2 y_{1:T}) \pi(\psi_1, \psi_2 | y_{1:T}).$$

4.5 Unknown variances

In many of the models analyzed in Chapter 3, the system and observation matrices G_t and F_t are set to specific values as part of the model specification. This is the case for polynomial and seasonal factor models, for example. The only possibly unknown parameters are therefore part of the variance matrices W_t and V_t. In Section 4.3.1, we discussed Bayesian conjugate inference for the simple case of V_t and W_t known up to a scale factor; in more general cases, analytical computations become more complex and MCMC approximations are used, as we illustrate in this section.

4.5.1 Constant unknown variances: *d* Inverse Gamma prior

This is the simplest and most commonly used model for unknown variances. Let us assume that the observations are univariate ($m = 1$). If the observation variance and the evolution covariance matrices are unknown, the simplest assumption is to consider them as time-invariant, with W diagonal. More specifically, and working as usual with the precisions, a *d-inverse-gamma* prior assumes that

$$V_t = \phi_y^{-1}, \qquad W_t = \text{diag}(\phi_{\theta,1}^{-1}, \ldots, \phi_{\theta,p}^{-1})$$

and $\phi_y, \phi_{\theta,1}, \ldots, \phi_{\theta,p}$ have independent Gamma distributions. This implies that the prior on the vector of the variances $(\phi_y^{-1}, \phi_{\theta,1}^{-1}, \ldots, \phi_{\theta,p}^{-1})$ is the product of $d = (p + 1)$ Inverse-Gamma densities. To fix the prior hyperparamters, it is usually convenient to express a prior guess on the unknown precisions, $\text{E}(\phi_y) = a_y$, and $\text{E}(\phi_{\theta,i}) = a_{\theta,i}$, with prior uncertainty summarized by the prior variances $\text{Var}(\phi_y) = b_y$, $\text{Var}(\phi_{\theta,i}) = b_{\theta,i}$, $i = 1 \ldots, p$, so that the Gamma priors can be parametrized as

$$\phi_y \sim \mathcal{G}(\alpha_y, \beta_y), \qquad \phi_{\theta,i} \sim \mathcal{G}(\alpha_{\theta,i}, \beta_{\theta,i}) \qquad i = 1, \ldots, p,$$

with

$$\alpha_y = \frac{a_y^2}{b_y}, \quad \beta_y = \frac{a_y}{b_y}, \qquad \alpha_{\theta,i} = \frac{a_{\theta,i}^2}{b_{\theta,i}}, \quad \beta_{\theta,i} = \frac{a_{\theta,i}}{b_{\theta,i}}, \qquad i = 1, \ldots, p.$$

As particular cases, this framework includes a Bayesian treatment of nth order polynomial models as well as the structural time series models of Harvey and coauthors, discussed in Chapter 3.

Given the observations $y_{1:T}$, the joint posterior of the states $\theta_{0:T}$ and the unknown parameters $\psi = (\phi_y, \phi_{\theta,1}, \ldots, \phi_{\theta,p})$ is proportional to the joint density

$$\begin{aligned}\pi(y_{1:T}, \theta_{0:T}, \psi) &= \pi(y_{1:T}|\theta_{0:T}, \psi) \cdot \pi(\theta_{0:T}|\psi) \cdot \pi(\psi) \\ &= \prod_{t=1}^{T} \pi(y_t|\theta_t, \phi_y) \cdot \prod_{t=1}^{T} \pi(\theta_t|\theta_{t-1}, \phi_{\theta,1}, \ldots, \phi_{\theta,p}) \cdot \pi(\theta_0) \cdot \pi(\phi_y) \cdot \prod_{i=1}^{p} \pi(\phi_{\theta,i}).\end{aligned}$$

Note that the second product in the factorization can also be written as a product over $i = 1, \ldots, p$, due to the diagonal form of W. This alternative factorization is useful when deriving the full conditional distribution of the $\phi_{\theta,i}$'s. A Gibbs sampler for the d-Inverse-Gamma model draws from the full conditional distribution of the states and from the full conditional distributions of $\phi_y, \phi_{\theta,1}, \ldots, \phi_{\theta,p}$ in turn. Sampling the states can be done using the FFBS algorithm of Section 4.4.1. Let us derive the full conditional distribution[1] of ϕ_y:

$$\begin{aligned}\pi(\phi_y|\ldots) &\propto \prod_{t=1}^{T} \pi(y_t|\theta_t, \phi_y) \cdot \pi(\phi_y) \\ &\propto \phi_y^{\frac{T}{2}+\alpha_y-1} \exp\left\{-\phi_y \cdot \left[\frac{1}{2}\sum_{t=1}^{T}(y_t - F_t\theta_t)^2 + \beta_y\right]\right\}.\end{aligned}$$

The full conditional of ϕ_y is therefore again a Gamma distribution,

$$\phi_y|\ldots \sim \mathcal{G}\left(\alpha_y + \frac{T}{2}, \beta_y + \frac{1}{2}SS_y\right),$$

with $SS_y = \sum_{t=1}^{T}(y_t - F_t\theta_t)^2$. Similarly, it is easy to show that the full conditionals of the $\phi_{\theta,i}$'s are as follows:

$$\phi_{\theta,i}|\ldots \sim \mathcal{G}\left(\alpha_{\theta,i} + \frac{T}{2}, \beta_{\theta,i} + \frac{1}{2}SS_{\theta,i}\right), \qquad i = 1, \ldots, p,$$

[1] The dots on the right-hand side of the conditioning vertical bar in $\pi(\phi_y|\ldots)$ stay for every other random variable in the model except ϕ_y, including the states $\theta_{0:T}$. This standard convention will be used throughout.

with $SS_{\theta,i} = \sum_{t=1}^{T}(\theta_{t,i} - (G_t\theta_{t-1})_i)^2$.

Example. Let us consider again the data on Spain investment (Section 3.2.1). We are going to fit a 2nd-order polynomial model—local linear growth—to the data. The priors for the precisions of the observation and evolution errors are (independent) gamma distributions with means a_y, a_μ, a_β and variances b_y, b_μ, b_β. We decide for the values $a_y = 1$, $a_\mu = a_\beta = 10$, with a common variance equal to 1000, to express a large uncertainty in the prior estimate of the precisions. The function `dlmGibbsDIG` can be called to generate a sample from the posterior distribution of the parameters and the states. The means and variances of the gamma priors are passed to the function via the arguments `a.y`, `b.y` (prior mean and variance of observation precision), `a.theta`, `b.theta` (prior mean(s) and variance(s) of evolution precision(s)). Alternatively, the prior distribution can be specified in terms of the usual shape and rate parameters of the gamma distribution. The arguments to pass in this case are `shape.y`, `rate.y`, `shape.theta`, and `rate.theta`. The number of samples from the posterior to generate is determined by the argument `n.sample`, while the logical argument `save.states` is used to determine whether to include the generated unobservable states in the output. In addition, a thinning parameter can be specified via the integer argument `thin`. This gives the number of Gibbs iterations to discard for every saved one. Finally, the data and the model are passed via the arguments `y` and `mod`, respectively. The following display show how `dlmGibbsDIG` works in practice.

R code

```
> invSpain <- ts(read.table("Datasets/invest2.dat",
+                           colClasses = "numeric")[,2]/1000,
+                start = 1960)
> set.seed(5672)
> MCMC <- 12000
> gibbsOut <- dlmGibbsDIG(invSpain, mod = dlmModPoly(2),
+                         a.y = 1, b.y = 1000,
+                         a.theta = 10, b.theta = 1000,
+                         n.sample = MCMC,
+                         thin = 1, save.states = FALSE)
```

Setting `thin = 1` means that the function actually generates a sample of size 24,000 but only keeps in the output every other value. In addition, the states are not returned (`save.states = FALSE`). Considering the first 2000 saved iterations as burn in, one can proceed to graphically assess the convergence and mixing properties of the sampler. Figure 4.5 displays a few diagnostic plots obtained from the MCMC output for the variances V, W_{11}, and W_{22}. The first row shows the traces of the sampler, i.e., the simulated values, the second the running ergodic means of the parameters (starting

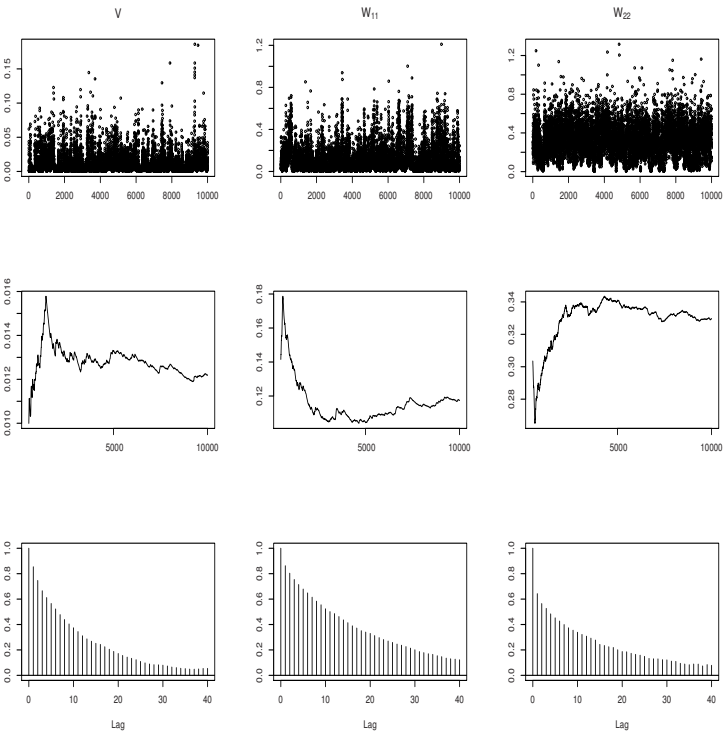

Fig. 4.5. Diagnostic plots for d-inverse-Gamma model applied to Spain investments

at iteration 500), and the last the estimated autocovariance functions. We obtained the running ergodic means using the function *ergMean*. For example, the plot in the first column and second row was created using the following commands.

R code

```
> use <- MCMC - burn
> from <- 0.05 * use
> plot(ergMean(gibbsOut$dV[-(1 : burn)], from), type = "l",
+      xaxt = "n", xlab = "", ylab = "")
> at <- pretty(c(0, use), n = 3)
> at <- at[at >= from]
> axis(1, at = at - from, labels = format(at))
```

From a visual assessment of the MCMC output it seems fair to deduce that convergence has been achieved and, while the acf's of the simulated variances do not decay very fast, the ergodic means are nonetheless pretty stable in the last part of the plots. One can therefore go ahead and use the MCMC

output to estimate the posterior means of the unknown variances. The function *mcmcMean* computes the (column) means of a matrix of simulated values, together with an estimate of the Monte Carlo standard deviation, obtained using Sokal's method (Section 1.6).

R code

```
> mcmcMean(cbind(gibbsOut$dV[-(1 : burn), ],
+           gibbsOut$dW[-(1 : burn), ]))
      V.1         V.2         V.3
  0.012197    0.117391    0.329588
 (0.000743) (0.007682) (0.007833)
```

Bivariate plots of the simulated parameters may provide additional insight. Consider the plots in Figure 4.6. The joint realizations seem to suggest that

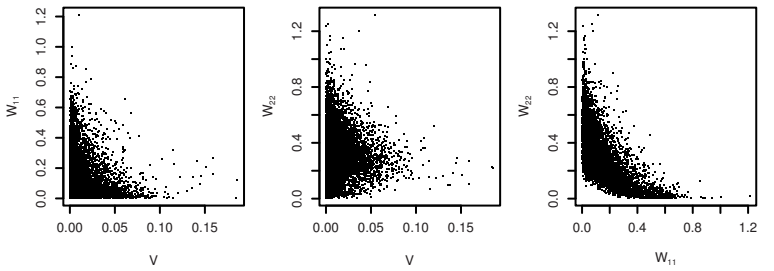

Fig. 4.6. Bivariate plots for d-inverse-Gamma model applied to Spain investments

one, or maybe two, of the model variances may be zero. From the third plot we can see that when W_{11} is close to zero then W_{22} is clearly positive, and vice versa: W_{11} and W_{22} cannot both be zero. The second plot suggests that this is also true for V and W_{22}. From the first plot it seems that V and W_{11} may be zero, possibly at the same time. In summary, by looking at the bivariate plots, three reduced models come up as alternative to the full model worth exploring: the submodel obtained by setting $V = 0$, the submodel $W_{11} = 0$, the submodel $V = W_{11} = 0$, and the submodel $W_{22} = 0$. We do not pursue the issue of Bayesian model selection here; a recent reference is Früwirth-Schnatter and Wagner (2008), who suggest a different prior on the unknown variances, more suitable for model comparison.

4.5.2 Multivariate extensions

Multivariate extensions of the d Inverse-Gamma model can be devised, using independent inverse-Wishart priors. Suppose that Y_t is m-variate, $m \geq 1$, and

W is block-diagonal with elements $(W_1, \ldots, W_h)$, W_i having dimension $p_i \times p_i$. Examples of DLMs with a block-diagonal state covariance matrix W include additive compositions of structural models or SUTSE models, presented in Chapter 3. At least in principle, the case of a general matrix W is obtained by letting $h = 1$.

Let us parametrize the model in the precision matrices $\Phi_0 = V^{-1}$ and $\Phi = W^{-1}$, the latter being block-diagonal with elements $\Phi_i = W_i^{-1}$, $i = 1, \ldots, h$. We assume that $\Phi_0, \Phi_1, \ldots, \Phi_h$ have independent Wishart priors, $\Phi_i \sim \mathcal{W}(\nu_i, S_i)$, $i = 0, \ldots, h$, where S_i is a symmetric positive definite matrix of dimensions $p_i \times p_i$, with $p_0 = m$. Then the posterior density $\pi(\theta_{0:T}, \Phi_0, \ldots, \Phi_h | y_{1:T})$ is proportional to

$$\begin{aligned} &\prod_{t=1}^{T} \mathcal{N}(y_t; F_t\theta_t, \Phi_0^{-1})\, \mathcal{N}(\theta_t; G\theta_{t-1}, \Phi^{-1}) \mathcal{N}(\theta_0; m_0, C_0) \\ &\qquad \cdot \mathcal{W}(\Phi_0; \nu_0, S_0) \prod_{i=1}^{h} \mathcal{W}(\Phi_i; \nu_i, S_i). \end{aligned} \tag{4.22}$$

A Gibbs sampler for π is obtained by iteratively sampling the states $\theta_{0:T}$ (via the FFBS algorithm) and the precisions $\Phi_0, \ldots, \Phi_h$ from their full conditionals. From (4.22) we see that the full conditional density of Φ_i, for $i = 1, \ldots, h$, is proportional to

$$\begin{aligned} &\prod_{t=1}^{T} \prod_{j=1}^{h} |\Phi_j|^{1/2} \exp\{-\frac{1}{2}(\theta_t - G_t\theta_{t-1})'\Phi(\theta_t - G_t\theta_{t-1})\} \cdot \\ &\qquad\qquad |\Phi_i|^{\nu_i - (p_i+1)/2} \exp\{-\mathrm{tr}(S_i\Phi_i)\} \\ \propto\ &|\Phi_i|^{T/2+\nu_i-(p_i+1)/2} \exp\{-\mathrm{tr}(\frac{1}{2}\sum_{t=1}^{T}(\theta_t - G_t\theta_{t-1})(\theta_t - G_t\theta_{t-1})'\Phi) - \mathrm{tr}(S_i\Phi_i)\} \end{aligned}$$

(see Section 1.5). Let

$$SS_t = (\theta_t - G_t\theta_{t-1})(\theta_t - G_t\theta_{t-1})'$$

and partition it in accordance with Φ:

$$SS_t = \begin{bmatrix} SS_{11,t} & \cdots & SS_{1h,t} \\ \vdots & \ddots & \vdots \\ SS_{h1,t} & \cdots & SS_{hh,t} \end{bmatrix}. \tag{4.23}$$

Then $\mathrm{tr}(SS_t\Phi) = \sum_{j=1}^{h} \mathrm{tr}(SS_{jj,t}\Phi_j)$, so that the full conditional of Φ_i results to be proportional to

$$|\Phi_i|^{T/2+\nu_i-(p_i+1)/2} \exp\left\{-\mathrm{tr}\left(\left(\frac{1}{2}SS_{i\cdot} + S_i\right)\Phi_i\right)\right\},$$

with $SS_{i\cdot} = \sum_{t=1}^{T} SS_{ii,t}$. That is, for $i = 1, \ldots, h$, the full conditional of Φ_i is Wishart, with parameters $(\nu_i + T/2, 1/2SS_{i\cdot} + S_i)$. In particular, for a DLM obtained by combining component models as in Section 3.2, we have

$$\theta_t = \begin{bmatrix} \theta_{1,t} \\ \vdots \\ \theta_{h,t} \end{bmatrix} \qquad G_t = \begin{bmatrix} G_{1,t} & \cdots & 0 \\ 0 & \ddots & 0 \\ 0 & \cdots & G_{h,t} \end{bmatrix},$$

with $\theta_{i,t}$ and $G_{i,t}$ referring to the ith component model. Then, the full conditional of Φ_i is $\mathcal{W}(\nu_i + T/2, S_i + 1/2SS_{i\cdot})$, and $S_{ii,t} = (\theta_{i,t} - G_{i,t}\theta_{i,t-1})(\theta_{i,t} - G_{i,t}\theta_{i,t-1})'$.

Similarly, one finds that the full conditional of Φ_0 is

$$\mathcal{W}(\nu_0 + T/2, S_0 + 1/2SS_y),$$

with $SS_y = \sum_{t=1}^{T}(y_t - F_t\theta_t)(y_t - F_t\theta_t)'$.

Example: SUTSE models.
As an illustration, let us consider again the data on Spain and Denmark investments. In Section 3.3.2, we used a SUTSE system where each series was described through a linear growth model; there, we just plugged in the MLE estimates of the unknown variances, while here we illustrate Bayesian inference. The prior distributions for the precisions $\Phi_0 = V^{-1}$, $\Phi_1 = W_\mu^{-1}$ and $\phi_2 = W_\beta^{-1}$ are independent Wishart, namely $\Phi_j \sim \mathcal{W}(\nu_j, S_j)$, $j = 0, 1, 2$.

It is convenient to express the Wishart hyperparameters as $\nu_0 = (\delta_0 + m - 1)/2 = (\delta_0 + 1)/2$, $\nu_j = (\delta_j + p_j - 1)/2 = (\delta_j + 1)/2$, $j = 1, 2$ and $S_0 = V_0/2, S_1 = W_{\mu,0}/2$, $S_2 = W_{\beta,0}/2$, so that, if $\delta_j > 2$, $j = 0, 1, 2$,

$$\mathrm{E}(V) = \frac{1}{\delta_0 - 2}V_0, \quad \mathrm{E}(W_\mu) = \frac{1}{\delta_1 - 2}W_{\mu,0} \quad \mathrm{E}(W_\beta) = \frac{1}{\delta_2 - 2}W_{\beta,0}$$

(see Appendix A). Thus, the matrix V_0 can be fixed as $V_0 = (\delta_0 - 2)\mathrm{E}(V)$, and similarly for $W_{\mu,0}$ and $W_{\beta,0}$. The parameters δ_i give an idea of the prior uncertainty: note that, from the full conditional distributions, we have

$$\mathrm{E}(V|y_{1:T}, W_\mu, W_\beta, \theta_{0:t}) =$$

$$= \frac{\delta_0 - 2}{(\delta_0 + T) - 2}\mathrm{E}(V) + \frac{T}{(\delta_0 + T) - 2}\frac{\sum_{t=1}^{T}(y_t - F_t\theta_t)(y_t - F_t\theta_t)'}{T},$$

and analogous expressions hold for the conditional expectations of W_μ and W_β. So, values of δ_i close to 2 imply a smaller weight of the prior in the updating. Expressing honest prior information on a covariance matrix is difficult, and data-dependent priors are sometimes used; but for this exercise let's suppose that

$$V_0 = (\delta_0 - 2) \begin{bmatrix} 10^2 & 0 \\ 0 & 500^2 \end{bmatrix},$$

$$W_{\mu,0} = (\delta_1 - 2) \begin{bmatrix} 0.01^2 & 0 \\ 0 & 0.01^2 \end{bmatrix}, \quad W_{\beta,0} = (\delta_2 - 2) \begin{bmatrix} 5^2 & 0 \\ 0 & 100^2 \end{bmatrix},$$

with $\delta_0 = \delta_2 = 3$ and $\delta_1 = 100$. The choice of $W_{\mu,0}$ and δ_1 expresses the prior assumption that the two individual linear growth models are in fact (independent) integrated random walks; the other choices give a prior idea of the signal-to-noise ratio.

The Gibbs sampling from the joint posterior $\pi(\theta_{0:T}, \Phi_0, \Phi_1, \Phi_2 | y_{1:T})$ generates in turn the state vectors $\theta_{0:T}$ and the precision Φ_0, Φ_1, Φ_2. The full conditional of Φ_0 is

$$\mathcal{W}\left(\frac{\delta_0 + 1 + T}{2}, \frac{1}{2}(V_0 + SS_y)\right)$$

and the full conditionals of $\Phi_1 = W_\mu^{-1}$ and $\Phi_2 = W_\beta^{-1}$ are, respectively, $\mathcal{W}((\delta_1 + 1 + T)/2, (W_{\mu,0} + SS_{1\cdot})/2)$ and $\mathcal{W}((\delta_2 + 1 + T)/2, (W_{\beta,0} + SS_{2\cdot})/2)$. Note that the function `rwishart` in the package dlm takes as inputs the degrees of freedom and the scale matrix (if $\Phi \sim \mathcal{W}(\delta/2, V_0/2)$, the degrees of freedom are δ and the scale matrix is V_0^{-1}).

R code

```
> inv <- read.table("Datasets/invest2.dat",
+                   col.names = c("Denmark", "Spain"))
> y <- ts(inv, frequency = 1, start = 1960)
> # prior hyperparameters
> delta0 <- delta2 <- 3 ; delta1 <- 100
> V0 <- (delta0 - 2) *diag(c(10^2, 500^2))
> Wmu0 <- (delta1 - 2) * diag(0.01^2, 2)
> Wbeta0 <- (delta2 - 2) * diag(c(5^2, 100^2))
> ## Gibbs sampling
> MC <- 30000
> TT <- nrow(y)
> gibbsTheta <- array(0, dim = c(TT + 1, 4, MC - 1))
> gibbsV <- array(0, dim = c(2, 2, MC))
> gibbsWmu <- array(0, dim = c(2, 2, MC))
> gibbsWbeta <- array(0, dim = c(2, 2, MC))
> mod <- dlm(FF = matrix(c(1, 0), nrow = 1) %x% diag(2),
+            V = diag(2),
+            GG = matrix(c(1, 0, 1, 1), 2, 2) %x% diag(2),
+            W = bdiag(diag(2), diag(2)),
+            m0 = c(inv[1, 1], inv[1, 2], 0, 0),
+            C0 = diag(x = 1e7, nrow = 4))
> # starting values
> mod$V <- gibbsV[,, 1] <- V0 / (delta0 - 2)
```

```
> gibbsWmu[,, 1] <- Wmu0 / (delta1 - 2)
> gibbsWbeta[,, 1] <- Wbeta0 / (delta2 - 2)
> mod$W <- bdiag(gibbsWmu[,, 1], gibbsWbeta[,, 1])
> # MCMC loop
> set.seed(3420)
> for(it in 1 : (MC - 1))
+    {
+    # generate states - FFBS
+    modFilt <- dlmFilter(y, mod, simplify = TRUE)
+    gibbsTheta[,, it] <- theta <- dlmBSample(modFilt)
+    # update V
+    S <- crossprod(y - theta[-1, ] %*% t(mod$FF)) + V0
+    gibbsV[,, it+1] <- solve(rwishart(df = delta0 + 1 + TT,
+                                      p = 2, Sigma = solve(S)))
+    mod$V <- gibbsV[,, it+1]
+    # update Wmu and Wbeta
+    theta.center <- theta[-1, ] - (theta[-(TT + 1), ]  %*%
+                                   t(mod$GG))
+    SS1 <- crossprod(theta.center)[1 : 2, 1 : 2] + Wmu0
+    SS2 <- crossprod(theta.center)[3 : 4, 3 : 4] + Wbeta0
+    gibbsWmu[,, it+1] <- solve(rwishart(df =delta1 + 1 + TT,
+                                        Sigma = solve(SS1)))
+    gibbsWbeta[,, it+1] <- solve(rwishart(df = delta2 + 1 + TT,
+                                          Sigma = solve(SS2)))
+    mod$W <- bdiag(gibbsWmu[,, it+1], gibbsWbeta[,, it+1])
+    }
```

We set the number of MCMC samples to 30,000 and remove the first 20,000 iterations as burn-in. Some convergence diagnostic plots are shown in Figures 4.7 and 4.8. In general, mixing in the Gibbs sampling is poorer if the parameters are strongly correlated; here, the prior restrictions on W_μ facilitate identifiability and MCMC mixing.

R code

```
> burn<- 1 : 20000
> par(mar = c(2, 4, 1, 1) + 0.1, cex = 0.8)
> par(mfrow=c(3,2))
> plot(ergMean(sqrt(gibbsV[1,1, -burn])), type="l",
+      main="", cex.lab=1.5, ylab=expression(sigma[1]),
+      xlab="MCMC iteration")
> acf(sqrt(gibbsV[1,1,-burn]),  main="")
```

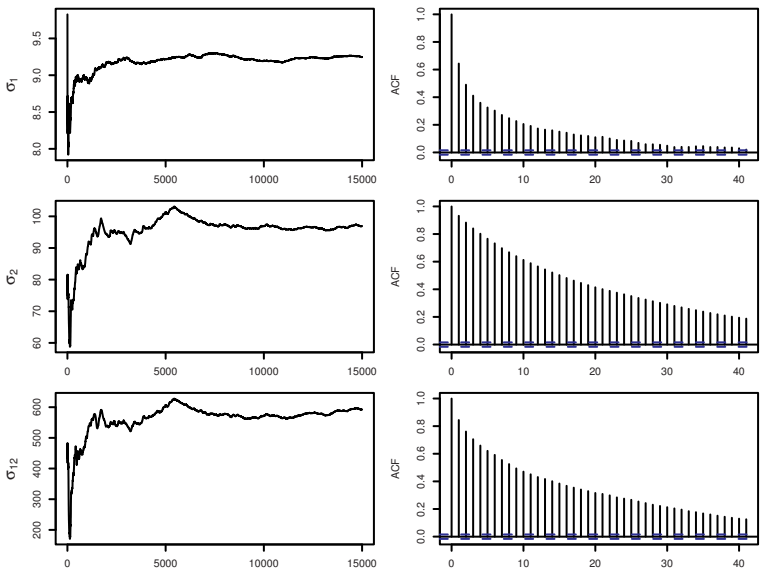

Fig. 4.7. MCMC ergodic means and autocorrelation for the observation covariance matrix V ($\sigma_1^2 = V_{11}, \sigma_2^2 = V_{22}, \sigma_{12} = V_{12}$)

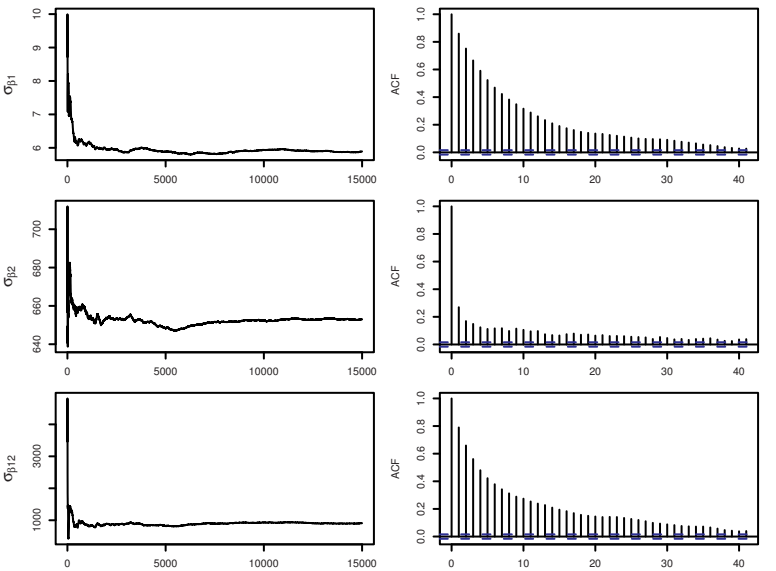

Fig. 4.8. MCMC ergodic means and autocorrelation for the covariance matrix W_β ($\sigma_{\beta,1}^2 = W_{\beta,11}, \sigma_{\beta,2}^2 = W_{\beta,22}, \sigma_{\beta,12} = W_{\beta,12}$)

The display below shows the MCMC estimates of the posterior means for the parameters in V and W_β, together with their estimated standard errors (in parenthesis), obtained from the function *mcmcMean*.

R code

```
> cbind(mcmcMean(gibbsV[1,1,-burn]),
+       mcmcMean(gibbsV[2,2,-burn]),
+       mcmcMean(gibbsV[2,1,-burn]))
```

$$\mathrm{E}(V|y_{1:T}) = \begin{bmatrix} 88.358 & 1018.469 \\ (1.1562) & (22.4865) \\ 1018.469 & 60066.43 \\ (22.4865) & (856.219) \end{bmatrix},$$

$$\mathrm{E}(W_\beta|y_{1:T}) = \begin{bmatrix} 37.399 & 396.591 \\ (0.9666) & (42.661) \\ 396.591 & 308729.71 \\ (42.661) & (2135.008) \end{bmatrix}.$$

Figure 4.9 shows the MCMC approximation of the Bayesian smoothing estimates of the level of the investments for Denmark and Spain, with marginal 5% and 95% quantiles.

The choice of an Inverse-Wishart prior on the unknown blocks of the covariance matrices has several advantages; in this exercise, computation of the full conditionals is made simple by the conjugacy properties of the Inverse-Wishart for the Gaussian model. In fact, inference on a covariance matrix is quite delicate, implying assumptions on the dependence structure of the data. We just note that the Inverse-Wishart prior may be too restrictive in modeling the prior uncertainty on the elements of the covariance matrix, as discussed earlier by Lindley (1978), and several generalizations have been proposed; some references are Dawid (1981), Brown et al. (1994), Dawid and Lauritzen (1993) in the context of graphical models, Consonni and Veronese (2003), Rajaratnam et al. (2008) and references therein.

4.5.3 A model for outliers and structural breaks

In this section we consider a generalization of the d-Inverse-Gamma model that is appropriate to account for outliers and structural breaks. As in Section 4.5.1, we assume that observations are univariate, that W_t is diagonal, and that the specification of F_t and G_t does not include any unknown parameter. To introduce the model, let us focus on observational outliers first. Structural breaks—or outliers in the state series—will be dealt with in a similar way later on. From the observation equation $Y_t = F_t\theta_t + v_t$, we see that a

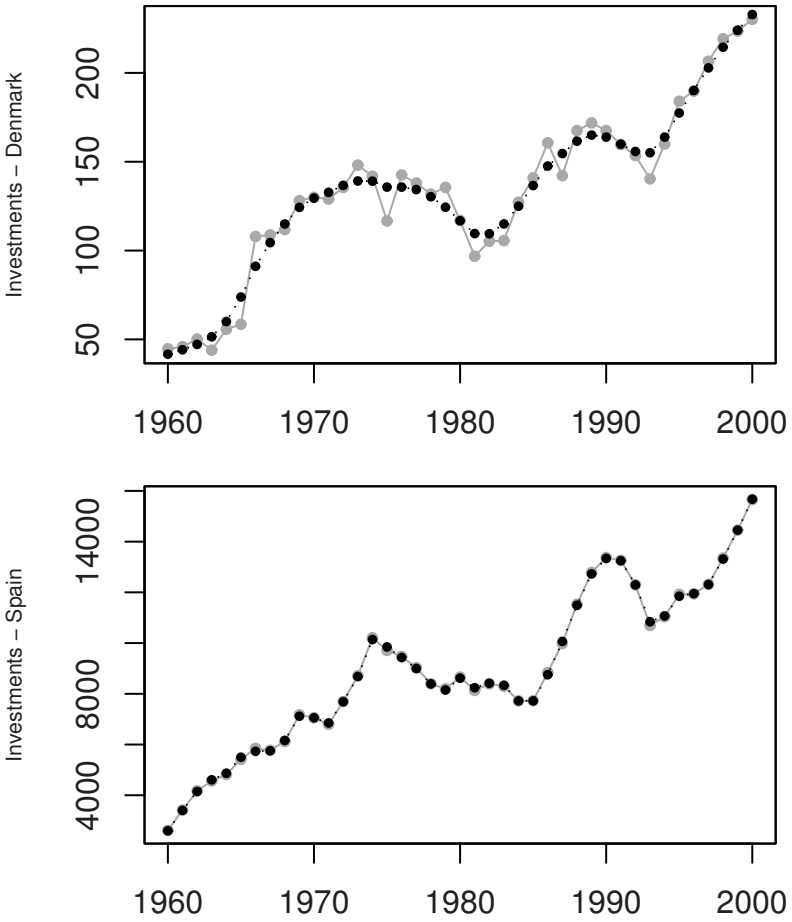

Fig. 4.9. MCMC smoothing estimates of the investments' level for Denmark and Spain, with posterior 90% probability intervals. The data are plotted in gray.

simple way to account for observations that are unusually far from their one-step-ahead predicted value is to replace the Normal distribution of v_t with a heavy-tailed distribution. The Student-t distribution family is particularly appealing in this respect for two reasons. On one hand, it can accommodate, through its degrees-of-freedom parameter, different degrees of heaviness in the tails. On the other hand, the Student-t distribution admits a simple representation as a scale mixture of Normal distributions, which allows one to treat a DLM with t-distributed observation errors as a Gaussian DLM, conditionally on the scale parameters. The obvious advantage is that all the standard algorithms for DLMs—from the Kalman filter to FFBS—can still be used, albeit

conditionally. In particular, within a Gibbs sampler, one can still draw the states from their full conditional distribution using the FFBS algorithm. We assume that the v_t have Student-t distributions with $\nu_{y,t}$ degrees of freedom and common scale parameter λ_y^{-1}:

$$v_t|\lambda_y, \nu_{y,t} \overset{indep}{\sim} \mathcal{T}(0, \lambda_y^{-1}, \nu_{y,t}).$$

Introducing latent variables $\omega_{y,t}$, distributed as $\mathcal{G}\big(\frac{\nu_{y,t}}{2}, \frac{\nu_{y,t}}{2}\big)$, we can equivalently write:

$$\begin{aligned} v_t|\lambda_y, \omega_{y,t} &\overset{indep}{\sim} \mathcal{N}\big(0, (\lambda_y \omega_{y,t})^{-1}\big), \\ \omega_{y,t}|\nu_{y,t} &\overset{indep}{\sim} \mathcal{G}\left(\frac{\nu_{y,t}}{2}, \frac{\nu_{y,t}}{2}\right). \end{aligned}$$

In other words, we are assuming that, given λ_y, the precisions $\phi_{y,t} = \lambda_y \omega_{y,t}$ of the observations in the DLM vary in a random fashion over time.

The latent variable $\omega_{y,t}$ in the previous representation can be informally interpreted as the degree of nonnormality of v_t. In fact, taking the $\mathcal{N}(0, \lambda_y^{-1})$ as baseline—corresponding to $\omega_{y,t} = \mathrm{E}(\omega_{y,t}) = 1$—values of $\omega_{y,t}$ lower than 1 make larger absolute values of v_t more likely. Figure 4.10 shows a plot of the

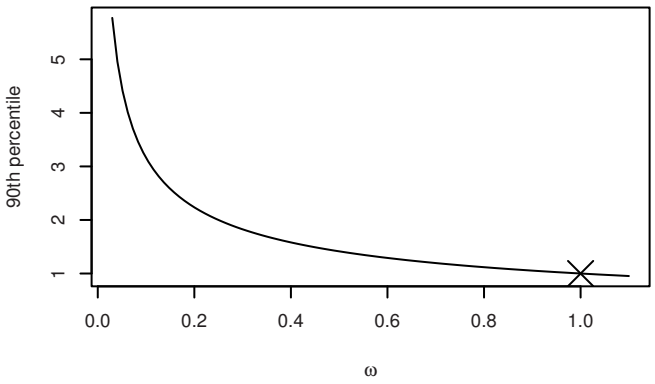

Fig. 4.10. 90th percentile of the conditional distribution of v_t as a function of $\omega_{y,t}$

90th percentile of the $\mathcal{N}(0, (\lambda_y \omega_{y,t})^{-1})$ as a function of $\omega_{y,t}$ (λ_y is selected so that the percentile is one when $\omega_{y,t}$ is one). From the previous discussion, it follows that the posterior mean of the $\omega_{y,t}$ can be used to flag possible outliers. As a prior for the precision parameter λ_y we choose a Gamma distribution with mean a_y and variance b_y,

$$\lambda_y|a_y, b_y \sim \mathcal{G}\left(\frac{a_y^2}{b_y}, \frac{a_y}{b_y}\right),$$

taking in turn a_y and b_y uniformly distributed on a large, but bounded, interval,

$$a_y \sim \mathcal{U}nif(0, A_y), \qquad b_y \sim \mathcal{U}nif(0, B_y).$$

Although the degrees-of-freedom parameter of a Student-t distribution can take any positive real value, we restrict for simplicity the set of possible values to a finite set of integers and set

$$\nu_{y,t}|p_y \overset{iid}{\sim} \mathcal{M}ult(1, p_y),$$

where $p_y = (p_{y,1}, \ldots, p_{y,K})$ is a vector of probabilities, the levels of the multinomial distribution are the integers $n_1, \ldots, n_K$, and the $\nu_{y,t}$'s are independent across t. As a convenient, yet flexible choice for $n_1, \ldots, n_K$ we use the set $\{1, 2, \ldots, 10, 20, \ldots, 100\}$. Note that for $\nu_{y,t} = 100$, v_t is approximately normally distributed, given λ_y. As a prior for p_y we adopt a Dirichlet distribution with parameter $\alpha_y = (\alpha_{y,1}, \ldots, \alpha_{y,K})$, $p_y \sim \mathcal{D}ir(\alpha_y)$. This completes the hierarchical specification of the prior distribution of the observation variances V_t. To account for possible outliers in the state components, a similar hierarchical structure is assumed for each diagonal element of W_t, i.e., for the precision parameters of the state innovations.

Note that in this model the precisions, or, equivalently, the variances, are allowed to be different at different times, although in a way that does not account for a possible correlation in time. In other words, the sequences of precisions at different times are expected to look more like independent, or exchangeable, sequences, rather than time series. For this reason the model is appropriate to account for occasional abrupt changes—corresponding to innovations having a large variance—in the state vector. For example, for polynomial and seasonal factor models, an outlier in a component of w_t corresponds to an abrupt change in the corresponding component of the state, such as a jump in the level of the series. However, the modeler does not anticipate these changes to present a clear pattern in time.

Writing $W_{t,i}$ for the ith diagonal element of W_t, $i = 1, \ldots, p$, the hierarchical prior can be summarized in the following display.

$$\begin{aligned}
V_t^{-1} &= \lambda_y \omega_{y,t}, & W_{t,i}^{-1} &= \lambda_{\theta,i} \omega_{\theta,ti}, \\
\lambda_y|a_y, b_y &\sim \mathcal{G}\left(\frac{a_y^2}{b_y}, \frac{a_y}{b_y}\right), & \lambda_{\theta,i}|a_{\theta,i}, b_{\theta,i} &\overset{indep}{\sim} \mathcal{G}\left(\frac{a_{\theta,i}^2}{b_{\theta,i}}, \frac{a_{\theta,i}}{b_{\theta,i}}\right), \\
\omega_{y,t}|\nu_{y,t} &\overset{indep}{\sim} \mathcal{G}\left(\frac{\nu_{y,t}}{2}, \frac{\nu_{y,t}}{2}\right), & \omega_{\theta,ti}|\nu_{\theta,ti} &\overset{indep}{\sim} \mathcal{G}\left(\frac{\nu_{\theta,ti}}{2}, \frac{\nu_{\theta,ti}}{2}\right), \\
a_y &\sim \mathcal{U}nif(0, A_y), & a_{\theta,i} &\overset{indep}{\sim} \mathcal{U}nif(0, A_{\theta,i}), \\
b_y &\sim \mathcal{U}nif(0, B_y), & b_{\theta,i} &\overset{indep}{\sim} \mathcal{U}nif(0, B_{\theta,i}), \\
\nu_{y,t} &\overset{indep}{\sim} \mathcal{M}ult(1; p_y) & \nu_{\theta,ti} &\overset{indep}{\sim} \mathcal{M}ult(1; p_{\theta,i}) \\
p_y &\sim \mathcal{D}ir(\alpha_y) & p_{\theta,i} &\overset{indep}{\sim} \mathcal{D}ir(\alpha_{\theta,i}),
\end{aligned}$$

with $\alpha_{\theta,i} = (\alpha_{\theta,i,1}, \ldots, \alpha_{\theta,i,K})$, $i = 1, \ldots, K$. Again, the levels of all the multinomial distributions are the integers $n_1, \ldots, n_K$.

A Gibbs sampler can be implemented to draw from the posterior distribution of parameters and states of the model specified above. Given all the unknown parameters, the states are generated at once from their joint full conditional distribution using the standard FFBS algorithm. The full conditional distributions of the parameters are easy to derive. We provide here a detailed derivation of the full conditional distribution of λ_y, as an example:

$$\begin{aligned}\pi(\lambda_y|\ldots) &\propto \pi(y_{1:T}|\theta_{1:T}, \omega_{y,1:T}, \lambda_y) \cdot \pi(\lambda_y|a_y, b_y) \\ &\propto \prod_{t=1}^{T} \lambda_y^{\frac{1}{2}} \exp\left\{-\frac{\omega_{y,t}\lambda_y}{2}(y_t - F_t\theta_t)^2\right\} \cdot \lambda_y^{\frac{a_y^2}{b_y}-1} \exp\left\{-\lambda_y \frac{a_y}{b_y}\right\} \\ &\propto \lambda_y^{\frac{T}{2}+\frac{a_y^2}{b_y}-1} \exp\left\{-\lambda_y \left[\frac{1}{2}\sum_{t=1}^{T} \omega_{y,t}(y_t - F_t\theta_t)^2 + \frac{a_y}{b_y}\right]\right\}.\end{aligned}$$

Therefore,

$$\lambda_y|\ldots \sim \mathcal{G}\left(\frac{a_y^2}{b_y} + \frac{T}{2}, \frac{a_y}{b_y} + \frac{1}{2}SS_y^*\right),$$

with $SS_y^* = \sum_{t=1}^{T} \omega_{y,t}(y_t - F_t\theta_t)^2$. A summary of all the full conditional distributions of the unknown parameters is shown in Table 4.1. All the full conditional distributions, except for those of $a_y, b_y, a_{\theta,i}, b_{\theta,i}$, are standard. The latter can be drawn from using ARMS. More specifically, we suggest using ARMS separately on each pair (a, b).

As an example of the use of the model discussed above, consider the time series of quarterly gas consumption in the UK from 1960 to 1986. The data are available in R as `UKgas`. A plot of the data, on the log scale, suggests a possible change in the seasonal factor around the third quarter of 1970. After taking logs, we employ a model built on a local linear trend plus seasonal component DLM to analyze the data. In this model the five-by-five variance matrix W_t has only three nonzero diagonal elements: the first refers to the level of the series, the second to the slope of the stochastic linear trend, and the third to the seasonal factor. We packaged the entire Gibbs sampler in the function `dlmGibbsDIGt`, available from the book website. The parameters $a_y, b_y, a_{\theta,1}, b_{\theta,1}, \ldots, a_{\theta,3}, b_{\theta,3}$ are taken to be uniform on $(0, 10^5)$, and the parameters of the four Dirichlet distributions of $p_y, p_{\theta,1}, p_{\theta,2}, p_{\theta,3}$ are all equal to 1/19. The posterior analysis is based on 10000 iterations, after a burn-in of 500 iterations. To reduce the autocorrelation, two extra sweeps were run between every two consecutive saved iterations, implying that the iterations of the sampler after burn-in were actually 30000.

R code

```
> y <- log(UKgas)
> set.seed(4521)
```

$$\lambda_y|\ldots \sim \mathcal{G}\left(\frac{a_y^2}{b_y}+\frac{T}{2}, \frac{a_y}{b_y}+\frac{1}{2}SS_y^*\right),$$

with $SS_y^* = \sum_{t=1}^T \omega_{y,t}(y_t - F_t\theta_t)^2$;

$$\lambda_{\theta,i}|\ldots \sim \mathcal{G}\left(\frac{a_{\theta,i}^2}{b_{\theta,i}}+\frac{T}{2}, \frac{a_{\theta,i}}{b_{\theta,i}}+\frac{1}{2}SS_{\theta,i}^*\right),$$

with $SS_{\theta,i}^* = \sum_{t=1}^T \omega_{\theta,ti}(\theta_{ti} - (G_t\theta_{t-1})_i)^2$, for $i = 1,\ldots,p$;

$$\omega_{y,t}|\ldots \sim \mathcal{G}\left(\frac{\nu_{y,t}+1}{2}, \frac{\nu_{y,t}+\lambda_y(y_t - F_t\theta_t)^2}{2}\right),$$

for $t = 1,\ldots,T$;

$$\omega_{\theta,ti}|\ldots \sim \mathcal{G}\left(\frac{\nu_{\theta,ti}+1}{2}, \frac{\nu_{\theta,ti}+\lambda_{\theta,i}(\theta_{ti} - (G_t\theta_{t-1})_i)^2}{2}\right),$$

for $i = 1,\ldots,p$ and $t = 1,\ldots,T$;

$$\pi(a_y, b_y|\ldots) \propto \mathcal{G}(\lambda_y; a_y, b_y),$$

on the set $0 < a_y < A_y, \quad 0 < b_y < B_y$;

$$\pi(a_{\theta,i}, b_{\theta,i}|\ldots) \propto \mathcal{G}(\lambda_{\theta,i}; a_{\theta,i}, b_{\theta,i}),$$

on $0 < a_{\theta,i} < A_{\theta,i}, \quad 0 < b_{\theta,i} < B_{\theta,i}$, for $i = 1,\ldots,p$;

$$\pi(\nu_{y,t} = k) \propto \mathcal{G}\left(\omega_{y,t}; \frac{k}{2}, \frac{k}{2}\right) \cdot p_{y,k},$$

on the set $\{n_1,\ldots,n_K\}$, for $t = 1,\ldots,T$;

$$\pi(\nu_{\theta,ti} = k) \propto \mathcal{G}\left(\omega_{\theta,ti}; \frac{k}{2}, \frac{k}{2}\right) \cdot p_{\theta,i,k},$$

on the set $\{n_1,\ldots,n_K\}$, for $i = 1,\ldots,p$ and $t = 1,\ldots,T$;

$$p_y|\ldots \sim \mathcal{D}ir(\alpha_y + N_y),$$

where $N_y = (N_{y,1},\ldots,N_{y,K})$ with, for each k, $N_{y,k} = \sum_{t=1}^T (\nu_{y,t} = k)$;

$$p_{\theta,i}|\ldots \sim \mathcal{D}ir(\alpha_{\theta,i} + N_{\theta,i}),$$

where $N_{\theta,i} = (N_{\theta,i,1},\ldots,N_{\theta,i,K})$ with, for each k, $N_{\theta,i,k} = \sum_{t=1}^T (\nu_{\theta,ti} = k)$, for $i = 1,\ldots,p$;

Table 4.1. Full conditional distributions for the model of Section 4.5.3

```
> MCMC <- 10500
> gibbsOut <- dlmGibbsDIGt(y, mod = dlmModPoly(2) + dlmModSeas(4),
+                          A_y = 10000, B_y = 10000, p = 3,
+                          n.sample = MCMC, thin = 2)
```

Figure 4.11, obtained with the code below, graphically summarizes the posterior means of the $\omega_{y,t}$ and $\omega_{\theta,ti}$, $t = 1, ..108$, $i = 1, 2, 3$.

R code

```
> burn <- 1 : 500
> nuRange <- c(1 : 10, seq(20, 100, by = 10))
> omega_y <- ts(colMeans(gibbsOut$omega_y[-burn, ]),
+               start = start(y), freq=4)
> omega_theta <- ts(apply(gibbsOut$omega_theta[,, -burn], 1 : 2,
+                         mean), start = start(y), freq = 4)
> layout(matrix(c(1, 2, 3, 4), 4, 1, TRUE))
> par(mar = c(5.1, 4.1, 2.1, 2.1))
> plot(omega_y, type = "p", ylim = c(0, 1.2), pch = 16,
+      xlab = "", ylab = expression(omega[list(y, t)]))
> abline(h = 1, lty = "dashed")
> for (i in 1 : 3)
+ {
+     plot(omega_theta[,i], ylim=c(0,1.2), pch = 16,
+          type = "p", xlab = "",
+          ylab = bquote(omega[list(theta, t * .(i))]))
+     abline(h = 1, lty = "dashed")
+ }
```

It is clear that there are no observational outliers, except perhaps for a mild outlier in the third quarter of 1983, with $\mathrm{E}(\omega_{\theta,t3}|y_{1:T}) = 0.88$. The trend is fairly stable, in particular its slope parameter. The seasonal component, on the other hand, presents several structural breaks, particularly in the first couple of years of the seventies. The most extreme change in the seasonal component happened in the third quarter of 1971, when the corresponding ω_t had an estimated value of 0.012. It can also be seen that after that period of frequent shocks, the overall variability of the seasonal component remained higher than in the first period of observation.

From the output of the Gibbs sampler one can also estimate the unobserved components—trend and seasonal variation—of the series. Figure 4.12 provides a plot of estimated trend and seasonal component, together with 95% probability intervals. An interesting feature of a model with time-specific variances, like the one considered here, is that confidence intervals need not be of constant width—even after accounting for boundary effects. This is clearly

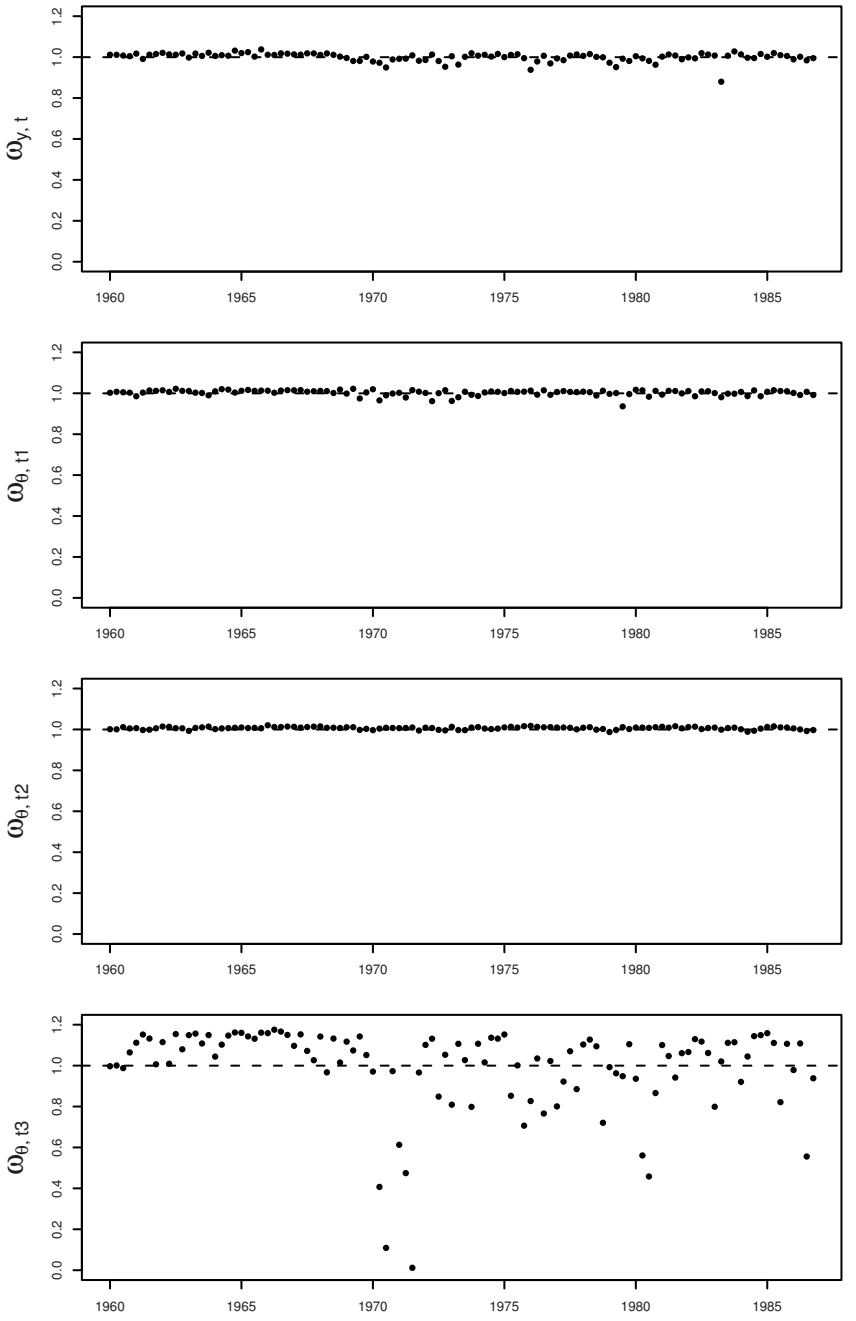

Fig. 4.11. UK gas consumption: posterior means of the ω_t's

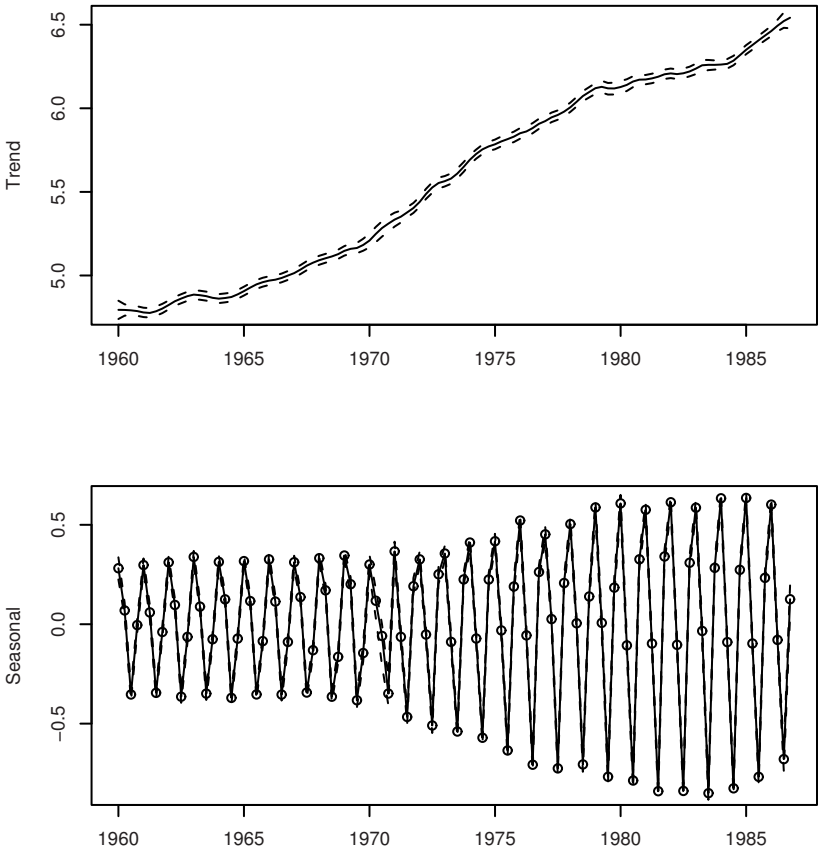

Fig. 4.12. UK gas consumption: trend and seasonal component, with 95% probability intervals

seen in the example, where the 95% probability interval for the seasonal component is wider in the period of high instability of the early seventies. The following code was used to obtain the plot.

R code

```
> thetaMean <- ts(apply(gibbsTheta, 1 : 2, mean),
+                 start = start(y),
+                 freq = frequency(y))
> LprobLim <- ts(apply(gibbsTheta, 1 : 2, quantile,
+                      probs = 0.025),
+                start = start(y), freq = frequency(y))
```

```
> UprobLim <- ts(apply(gibbsTheta, 1 : 2, quantile,
+                     probs = 0.975),
+                start = start(y), freq = frequency(y))
> par(mfrow = c(2, 1), mar = c(5.1, 4.1, 2.1, 2.1))
> plot(thetaMean[, 1], xlab = "", ylab = "Trend")
> lines(LprobLim[, 1], lty = 2); lines(UprobLim[, 1], lty = 2)
> plot(thetaMean[, 3], xlab = "", ylab = "Seasonal", type = "o")
> lines(LprobLim[, 3], lty = 2); lines(UprobLim[, 3], lty = 2)
```

4.6 Further examples

We give further examples of Bayesian analysis via MCMC for DLMs. The first is useful to compare MLE and Bayesian estimates.

4.6.1 Estimating the output gap: Bayesian inference

Let us consider again the problem of estimating the output gap, which we treated in Subsection 3.2.6 using maximum likelihood estimates of the unknown parameters of the model. Here we illustrate, instead, Bayesian inference. More specifically, we show how to implement in R a hybrid sampler using *dlmBSample* and *arms*. The data consist of the quarterly time series of deseasonalized real gross domestic product (GDP) of the US from 1950 to 2004, on a logarithmic scale. Following standard econometric practice, we had assumed that the GDP can be decomposed into two unobservable components: a stochastic trend and a stationary component. Here we compute Bayesian estimates of the two components as well as the parameters of the model.

The stochastic trend was described by a local linear trend, while the stationary component was an AR(2) process. As a DLM, the model is the sum, in the sense discussed in Section 3.2, of a polynomial model of order two and a DLM representation of a stationary AR(2) process, observed with no error. The matrices of the resulting DLM were given by (3.36) in Section 3.2.6. The first component of the state vector represents the trend, while the third is the AR(2) stationary component. The AR parameters ϕ_1 and ϕ_2 must lie in the stationarity region $\mathcal{S}$ defined by

$$\begin{aligned} \phi_1 + \phi_2 &< 1, \\ \phi_1 - \phi_2 &> -1, \\ |\phi_2| &< 1. \end{aligned}$$

The prior we select for (ϕ_1, ϕ_2) is a product of a $\mathcal{N}(0, (2/3)^2)$ and a $\mathcal{N}(0, (1/3)^2)$, restricted to $\mathcal{S}$. In this way, the prior penalizes those values of the AR parameters close to the boundary of the stationarity region. For the three precisions,

i.e., the inverses of the variances σ_μ^2, σ_δ^2, and σ_u^2, we assume independent gamma priors with mean a and variance b:

$$\mathcal{G}\left(\frac{a^2}{b}, \frac{a}{b}\right).$$

In this specific case, we set $a = 1$, $b = 1000$. A hybrid sampler can draw in turn the AR parameters from $\pi(\phi_1, \phi_2 | \sigma_\mu^2, \sigma_\delta^2, \sigma_u^2, y_{0:T})$, the states, and the three precisions from their full conditional distribution given the states and the AR parameters. In the notation used in Algorithm 4.3, $\psi_1 = (\phi_1, \phi_2)$ and $\psi_2 = ((\sigma_\mu^2)^{-1}, (\sigma_\delta^2)^{-1}, (\sigma_u^2)^{-1})$. The precisions, given the states and the AR parameters, are conditionally independent and gamma-distributed. Specifically,

$$\begin{aligned}
(\sigma_\mu^2)^{-1} | \ldots &\sim \mathcal{G}\left(\frac{a^2}{b} + \frac{T}{2}, \frac{a}{b} + \frac{1}{2}\sum_{t=1}^{T}\big(\theta_{t,1} - (G\theta_{t-1})_1\big)^2\right), \\
(\sigma_\delta^2)^{-1} | \ldots &\sim \mathcal{G}\left(\frac{a^2}{b} + \frac{T}{2}, \frac{a}{b} + \frac{1}{2}\sum_{t=1}^{T}\big(\theta_{t,2} - (G\theta_{t-1})_2\big)^2\right), \\
(\sigma_u^2)^{-1} | \ldots &\sim \mathcal{G}\left(\frac{a^2}{b} + \frac{T}{2}, \frac{a}{b} + \frac{1}{2}\sum_{t=1}^{T}\big(\theta_{t,3} - (G\theta_{t-1})_3\big)^2\right).
\end{aligned} \tag{4.24}$$

The AR parameters, given the precisions (but not the states), have a non-standard distribution and we can use ARMS to draw from their joint full conditional distribution. One can write a function to implement the sampler in R. One such function, on which the analysis that follows is based, is available from the book web site. We reproduce below the relevant part of the main loop. In the code, `theta` is a $(T + 1)$ matrix of states and `gibbsPhi` and `gibbsVars` are matrices in which the results of the simulation are stored. The states, generated in the loop, can optionally be saved, but they can also be easily generated again, given the simulated values of the AR and variance parameters.

R code

```
for (it in 1 : mcmc)
{
    ## generate AR parameters
    mod$GG[3 : 4, 3] <- arms(mod$GG[3 : 4, 3],
                             ARfullCond, AR2support, 1)
    ## generate states - FFBS
    modFilt <- dlmFilter(y, mod, simplify = TRUE)
    theta[] <- dlmBSample(modFilt)
    ## generate W
    theta.center <- theta[-1, -4, drop = FALSE] -
        (theta[-(nobs + 1), , drop = FALSE] %*% t(mod$GG))[, -4]
```

```
12      SStheta <- drop(sapply( 1 : 3, function(i)
                        crossprod(theta.center[, i])))
14      diag(mod$W)[1 : 3] <-
            1 / rgamma(3, shape = shape.theta,
16          rate = rate.theta + 0.5 * SStheta)
        ## save current iteration, if appropriate
18      if ( !(it %% every) )
        {
20          it.save <- it.save + 1
            gibbsTheta[, , it.save] <- theta
22          gibbsPhi[it.save, ] <- mod$GG[3 : 4,3]
            gibbsVars[it.save, ] <- diag(mod$W)[1 : 3]
24      }
    }
```

The 'if' statement on line 18 takes care of the thinning, saving the draw only when the iteration counter *it* is divisible by *every*. The object *SStheta* (line 12) is a vector of length 3 containing the sum of squares appearing in the full conditional distributions of the precisions (equations (4.24)). The two functions *ARfullCond* and *AR2support*, which are the main arguments of *arms* (line 5) are defined, inside the main function, as follows.

R code

```
AR2support <- function(u)
2 {
      ## stationarity region for AR(2) parameters
4     (sum(u) < 1) && (diff(u) < 1) && (abs(u[2]) < 1)
  }
6 ARfullCond <- function(u)
  {
8     ## log full conditional density for AR(2) parameters
      mod$GG[3 : 4, 3] <- u
10    -dlmLL(y, mod) + sum(dnorm(u, sd = c(2, 1) * 0.33,
                                 log = TRUE))
12 }
```

The sampler was run using the following call, where *gdp* is a time series object containing the data.

R code

```
outGibbs <- gdpGibbs(gdp, a.theta = 1, b.theta = 1000, n.sample =
2                    2050, thin = 1, save.states = TRUE)
```

Discarding the first 50 draws as burn in, we look at some simple diagnostic plots. The traces of the simulated variances (Figure 4.13) do not show any particular sign of a nonstationary behavior. We have also plotted the

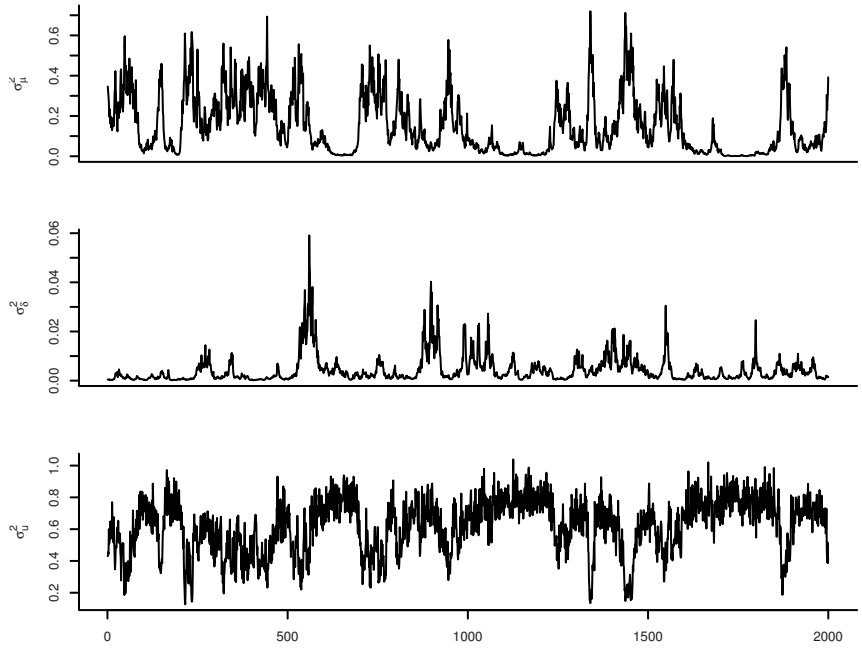

Fig. 4.13. GDP: traces of simulated variances

running ergodic means of the simulated standard deviations σ_μ σ_δ, and σ_u (Figure 4.14). The first plot shows $n^{-1}\sum_{i=1}^{n}\sigma_\mu^{(i)}$ versus i, and similarly for the second and third. In other words, this is the MC estimate of σ_μ versus the number of iterations of the sampler. The estimates look reasonably stable in the last part of the plot. (This impression was also confirmed by the results from a longer run, not shown here). The empirical autocorrelation functions of the three variances (Figure 4.15) give an idea of the degree of autocorrelation in the sampler. In the present case, the decay of the ACF is not very fast; this will reflect in a relatively large Monte Carlo standard error of the Monte Carlo esimates. Clearly, a smaller standard error can always be achieved by running the sampler longer. Similar diagnostic plots can be done for the AR parameters. The reader can find in the display below, for the three standard deviations in the model and the two AR parameters, the estimates of the posterior means and their estimated standard errors, obtained using Sokal's method (see Section 1.6). In addition, equal-tail 90% probability intervals are

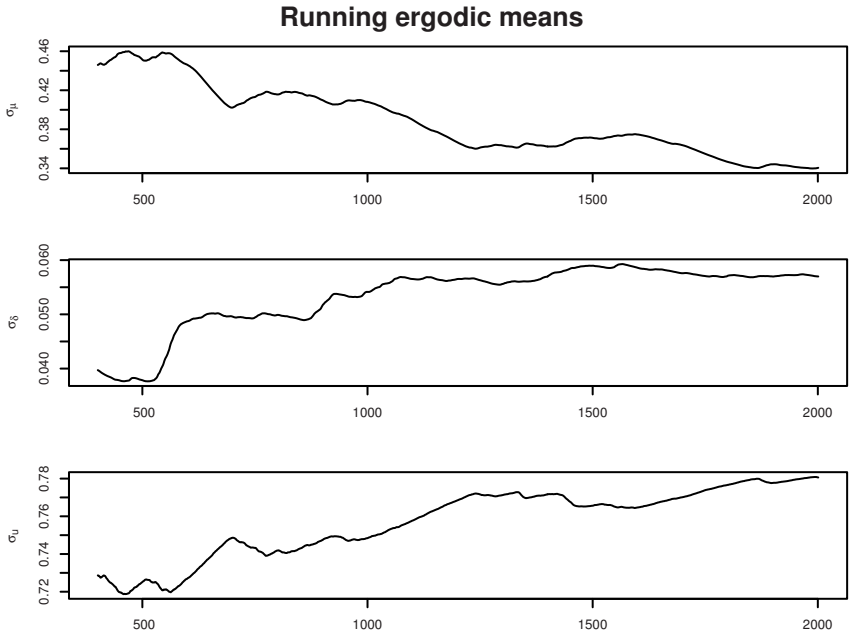

Fig. 4.14. GDP: ergodic means

derived for the five parameters. These probability intervals give an idea of the region where most of the posterior probability is contained.

R code

```
> mcmcMeans(outGibbs$phi[-burn,], names = paste("phi", 1:2))
    phi 1     phi 2
   1.3422   -0.4027
 ( 0.0112) ( 0.0120)
> apply(outGibbs$phi[-burn,], 2, quantile, probs = c(.05,.95))
        [,1]       [,2]
5%  1.174934 -0.5794382
95% 1.518323 -0.2495367
> mcmcMeans(sqrt(outGibbs$vars[-burn,]),
            names = paste("Sigma", 1:3))
   Sigma 1    Sigma 2    Sigma 3
  0.34052    0.05698    0.78059
 (0.03653)  (0.00491)  (0.01766)
> apply(sqrt(outGibbs$vars[-burn,]), 2, quantile,
        probs = c(.05,.95))
          [,1]       [,2]      [,3]
```

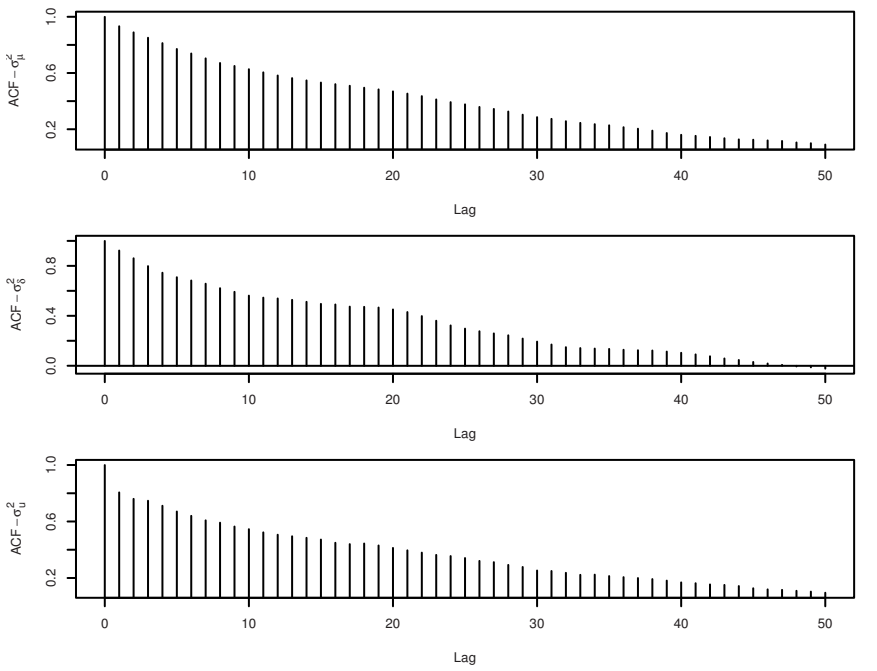

Fig. 4.15. GDP: Autocorrelation functions

```
5%  0.06792123 0.02057263 0.5596150
95% 0.65583319 0.12661949 0.9294306
```

One can also plot histograms based on the output of the sampler, to gain some insight about the shape of posterior distributions of parameters or functions thereof—at least for univariate marginal posteriors. Figure 4.16 displays the histograms of the posterior distributions of the three variances.

Scatterplots are sometimes useful to explore the shape of bivariate distributions, especially for pairs of parameters that are highly dependent on each other. Figure 4.17 displays a bivariate scatterplot of (ϕ_1, ϕ_2), together with their marginal histograms. From the picture, it is clear that there is a strong dependence between ϕ_1 and ϕ_2, which, incidentally, confirms that drawing the two at the same time was the right thing to do in order to improve the mixing of the chain.

Finally, since the sampler also included the unobservable states as latent variables, one can obtain posterior distributions and summaries of the states. In particular, in this example it is of interest to separate the trend of the GDP from the (autocorrelated) noise. The posterior mean of the trend at time t is estimated by the mean of the simulated values $\theta_{t,1}^{(i)}$. Figure 4.18 displays

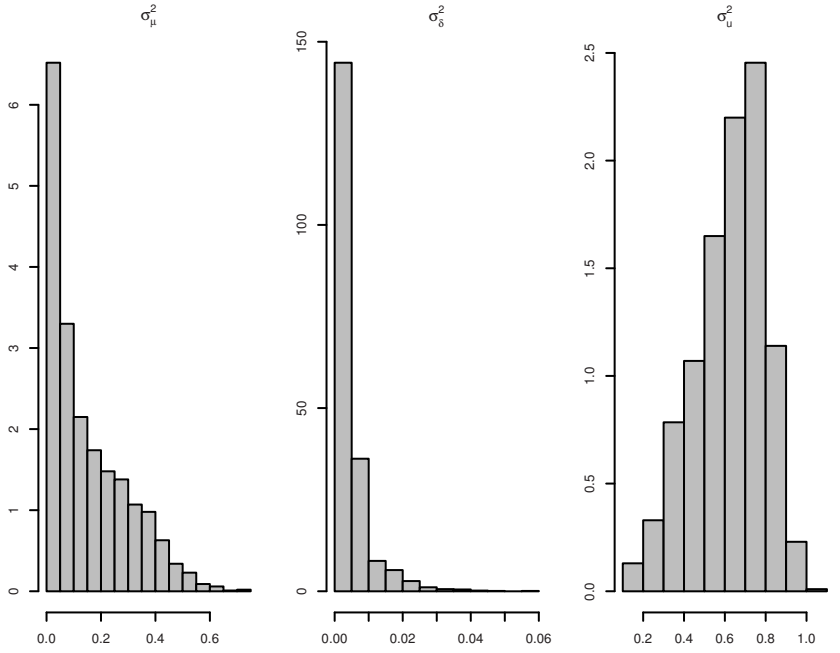

Fig. 4.16. GDP: posterior distributions of the model variances

graphically the posterior mean of the trend, together with the data, and the posterior mean of the AR(2) noise process, represented by $\theta_{t,3}$.

4.6.2 Dynamic regression

Suppose we have a time series of cross sectional data, $(Y_{i,t}, x_{i,t}), i = 1, \ldots, m$, where $Y_{i,t}$ is the value of a response variable Y corresponding to the value $x_{i,t}$ of one or more covariates X, observed over time. Typically, the interest is in estimating the regression function $m_t(x) = E(Y_t|x)$ from the cross-sectional data, at time t; moreover, one usually wants to make inference on the evolution of the regression curve over time. In Section 3.3.5 we introduced a dynamic regression model in the form of DLM for data of this nature. The model is applied here to a problem of interest in financial applications, namely, estimating the term structure of interest rates.

The problem is briefly described as follows. Let $P_t(x)$ be the price at time t of a zero-coupon bond that gives 1 euro at time to maturity x. The curve $P_t(x)$, $x \in (0, T)$, is called discount function. Other curves of interest are obtained as one-by-one transformations of the discount function; the yield curve is $\gamma_t(x) = -\log P_t(x)/x$, and the instantaneous (nominal) forward rate curve is $f_t(x) = d(-\log P_t(x)/dx) = (dP_t(x)/dx)/P_t(x)$. The yield curve, or one of its transformations, allows us to price any coupon bond as the sum

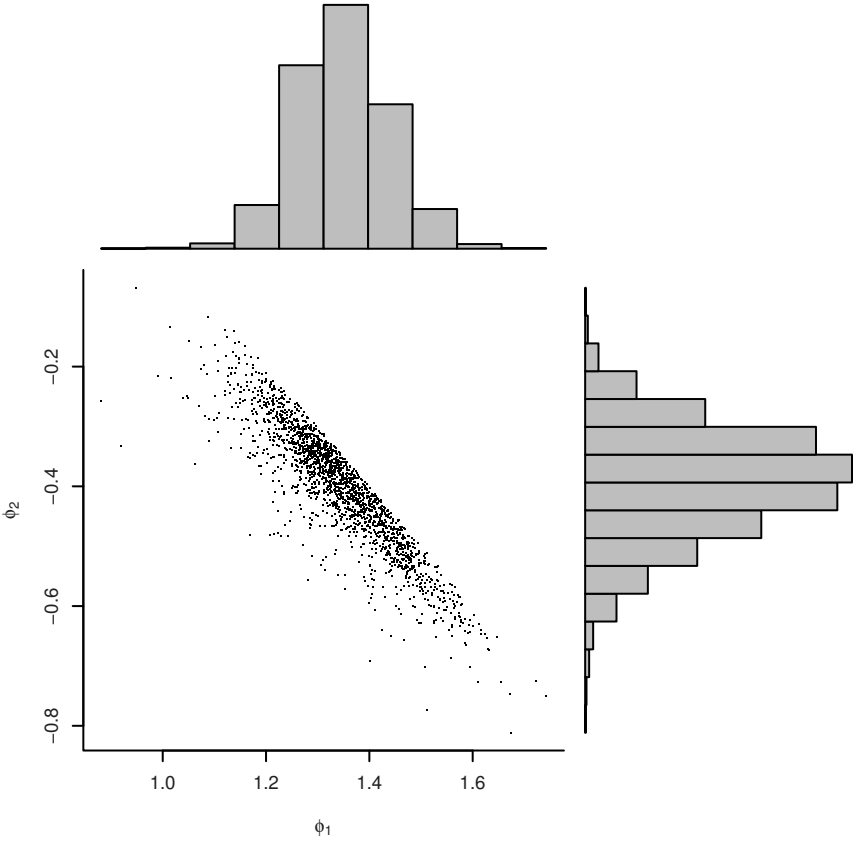

Fig. 4.17. GDP: posterior distribution of AR parameters

of the present values of future coupon and principal payments. Of course, the whole curve cannot be observed, but it has to be estimated from the bond prices observed for a finite number of times-to-maturity, $x_1, \ldots, x_m$ say. More precisely, at time t we have data $(y_{i,t}, x_i)$, $i = 1, \ldots, m$, where $y_{i,t}$ is the observed yield corresponding to time-to-maturity x_i. Due to market frictions, yields are subject to measurement error, so that the observed yields are described by

$$Y_{i,t} = \gamma_t(x_i) + v_{i,t}, \quad v_{i,t} \overset{iid}{\sim} \mathcal{N}(0, \sigma^2),\ i = 1, \ldots, m.$$

Several cross sectional models for the yield curve have been proposed in the literature; one of the most popular is the Nelson and Siegel (1987) model. In fact, Nelson and Siegel modeled the forward rate curve, as

$$f_t(x) = \beta_{1,t} + \beta_{2,t} e^{-\lambda x} + \beta_{3,t} \lambda\, x\, e^{-\lambda x},$$

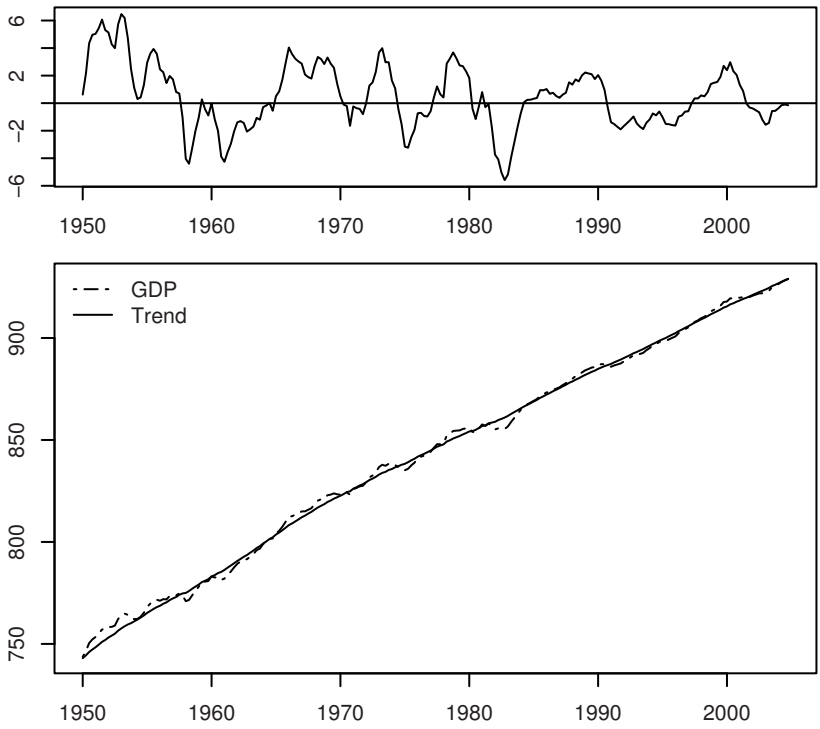

Fig. 4.18. GDP: posterior mean of trend and AR(2) noise

from which the yield curve can be obtained

$$\gamma_t(x) = \beta_{1,t} + \beta_{2,t}\frac{1-e^{-\lambda x}}{\lambda\, x} + \beta_{3,t}\left(\frac{1-e^{-\lambda x}}{\lambda\, x} - e^{-\lambda x}\right).$$

This model is not linear in the parameters $(\beta_{1,t}, \beta_{2,t}, \beta_{3,t}, \lambda)$; however, the decay parameter λ is usually approximated with a fixed value, so that the model is more simply treated as a linear model in the unknown parameters $\beta_{1,t}, \beta_{2,t}, \beta_{3,t}$. Thus, for a fixed value of λ, the model is of the form (3.43), with $k = 3$ and

$$h_1(x) = 1, \quad h_2(x) = \frac{1-e^{-\lambda x}}{\lambda\, x}, \quad h_3(x) = \left(\frac{1-e^{-\lambda x}}{\lambda\, x} - e^{-\lambda x}\right).$$

Consider the data plotted in Figure 3.19, which are monthly yields of US bonds for $m = 17$ times-to-maturity from 3 to 120 months, from January 1985 to December 2000; for a detailed description, see Diebold and Li (2006). These authors used the Nelson and Siegel cross sectional model to fit the yield curve at time t. In fact, they regard such a model as a latent factor model (see

Section 3.3.6), with $\beta_{1,t}, \beta_{2,t}, \beta_{3,t}$ playing the role of latent dynamic factors (long-term, short-term and medium-term factors), also interpreted in terms of level, slope and curvature of the yield curve. In these terms, λ determines the maturity at which the loading on the medium-term factor, or curvature, achieves its maximum; taking 30-months maturity, they fix $\lambda = 0.0609$, which is the value we take here. Given λ, cross-sectional estimates of β_t at time t

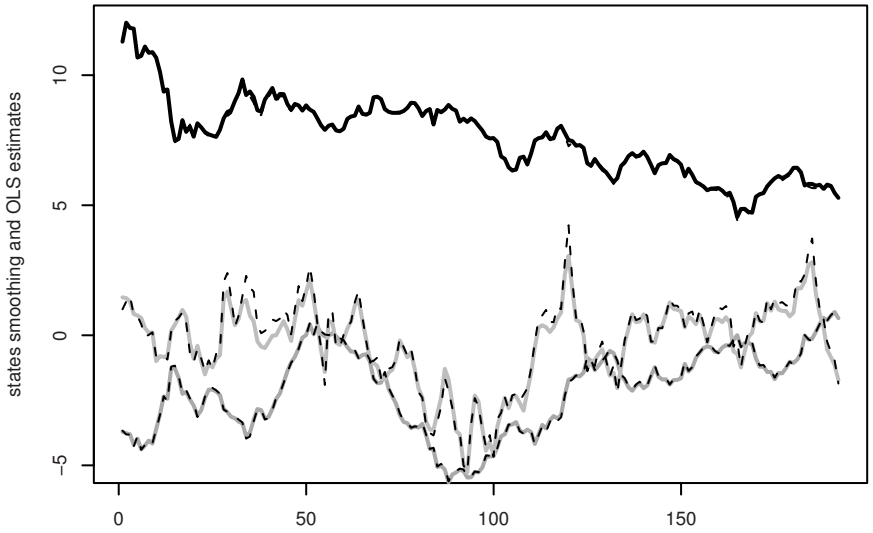

Fig. 4.19. MCMC smoothing estimates of the regression coefficients $\beta_{1,t}, \beta_{2,t}, \beta_{3,t}$ (solid lines: black, darkgray, gray respectively). The dashed lines are the month-by-month OLS estimates

are obtained by ordinary least squares (OLS), from the cross-sectional data $(y_{1,t}, \ldots, y_{m,t})$.

R code

```
> yields <- read.table("Datasets/yields.dat")
> y <- yields[1 : 192, 3 : 19]; y <- as.matrix(y)
> x <- c(3,6,9,12,15,18,21,24,30,36,48,60,72,84,96,108,120)
> p <- 3; m <- ncol(y); TT <- nrow(y)
> persp(x = x, z = t(y), theta = 40, phi = 30, expand = 0.5,
+       col = "lightblue", ylab = "time", zlab = "yield",
+       ltheta = 100, shade = 0.75, xlab = "maturity (months)")
> ### Cross-sectional model
> lambda <- 0.0609
> h2 <- function(x) {(1 - exp(-lambda * x)) / (lambda * x)}
> h3 <- function(x) {((1 - exp(-lambda * x)) / (lambda*x)) -
```

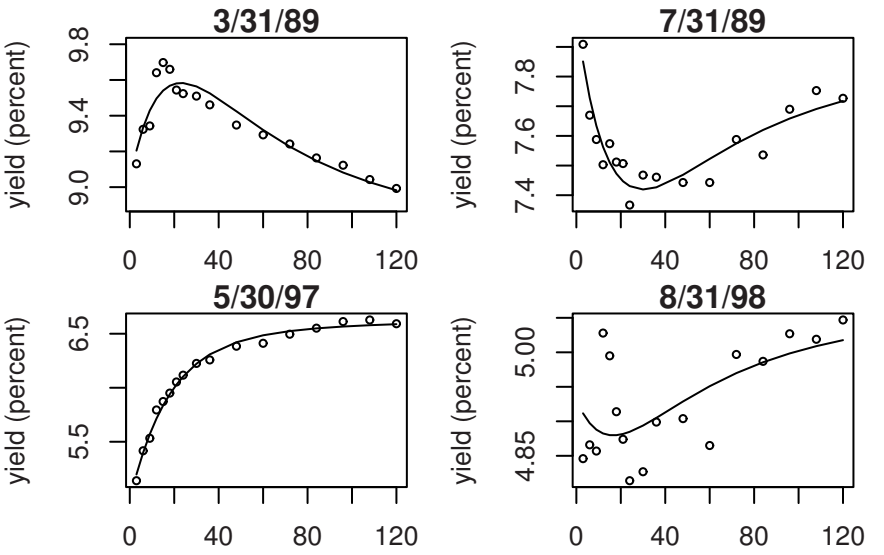

Fig. 4.20. Observed and OLS fitted yields curve, at selected dates

```
+                    exp(-lambda * x)}
> X <- cbind(rep(1, m), h2(x), h3(x))
> ## OLS estimates
> betahat <- solve(crossprod(X), crossprod(X, t(y)))
> nelsonSiegel <- function(x, beta) {
+       beta[1] + beta[2] * h2(x)+ beta[3] * h3(x)}
> month <- 51
> plot(x, y[month, ], xlab = "maturity(months)",
+      ylab = "yield (percent)", ylim = c(8.9,9.8),
+      main = "yield curve on 3/31/89")
> lines(x, nelsonSiegel(x, betahat[, month]))
```

Month-by-month OLS estimates over time are plotted in Figure 4.19 (dashed lines). A look at the autocorrelation function of the OLS estimates $\hat{\beta}_t$ and at the estimated residual variance gives a feeling of their evolution over time (plots not shown).

R code

```
> acf(t(betahat))
> yfit <- t(X %*% betahat)
> res <- (y - yfit)^2
> s2 <- rowSums(res) / (m - p)
> ts.plot(sqrt(s2))
```

The fit of the data is quite good, and the model adapts to the different shapes of the yield curve over time; see, e.g., Figure 4.20.

However, a dynamic estimate of the yield curve would give a more complete understanding of the problem. To this aim, as discussed in Section 3.3.5, a DLM can be used, introducing a state equation to describe the evolution of the regression coefficients. For example, Diebold et al. (2006) consider a VAR(1) states dynamics, also including the effects of macroeconomic variables. Petrone and Corielli (2005) propose a DLM where the state equation is derived from no-arbitrage constraints imposed on the yield curve evolution. In these papers, estimation of constant unknown parameters in the DLM matrices is obtained by MLE. Instead, here we illustrate Bayesian inference on the unknown parameters and the states of the model. For simplicity, we model $\beta_{1,t}, \beta_{2,t}, \beta_{3,t}$ as independent AR(1) processes. More specifically, the DLM we estimate is

$$\begin{aligned} Y_t &= F\theta_t + v_t, \qquad v_t \sim \mathcal{N}(0, V), \\ \theta_t &= G\theta_{t-1} + w_t, \quad w_t \sim \mathcal{N}(0, W), \end{aligned} \tag{4.25}$$

where $Y_t = (Y_{1,t}, \cdots, Y_{m,t})'$, $\theta_t = (\beta_{1,t}, \beta_{2,t}, \beta_{3,t})'$,

$$\begin{aligned} F &= \begin{bmatrix} 1 & h_2(x_1) & h_3(x_1) \\ 1 & h2(x_2) & h_3(x_2) \\ \vdots & \vdots & \vdots \\ 1 & h_2(x_m) & h_3(x_m) \end{bmatrix}, \\ G &= \text{diag}(\psi_1, \psi_2, \psi_3) \\ V &= \text{diag}(\phi_{y,1}^{-1}, \cdots, \phi_{y,m}^{-1}), \\ W &= \text{diag}(\phi_{\theta,1}^{-1}, \phi_{\theta,2}^{-1}, \phi_{\theta,3}^{-1}). \end{aligned}$$

Note that, while we assumed homoscedastic residuals in the cross sectional model for at each t, in the DLM above we allow different variances $\phi_{y,i}^{-1}, i = 1, \ldots, m$ for the yields at different times-to-maturity, although they are taken as time-invariant for simplicity.

As the prior, we assume that the AR parameters in the matrix G are i.i.d. Gaussian

$$\psi_j \overset{indep}{\sim} \mathcal{N}(\psi_0, \tau_0), \qquad j = 1, 2, 3,$$

with $\psi_0 = 0$ and $\tau_0 = 1$. We do not restrict the ψ_j to lie in the stationarity region. For the unknown variances, we use a d-Inverse-Gamma prior as in Section 4.5.1, so that the precisions have independent Gamma densities

$$\begin{aligned} \phi_{y,i} &\sim \mathcal{G}(\alpha_{y,i}, b_{y,i}), \qquad i = 1, 2, \cdots, m \\ \phi_{\theta,j} &\sim \mathcal{G}(\alpha_{\theta,j}, b_{\theta,j}), \qquad j = 1, \ldots, p. \end{aligned}$$

In the implementation below, we use $\alpha_{y,i} = 3, b_{y,i} = 0.01$, $i = 1, \ldots, m$ and $\alpha_{\theta,j} = 3, b_{\theta,j} = 1$, $j = 1, \ldots, p$. These choices correspond to prior guesses

$\mathrm{E}(\phi_{y,i}^{-1}) = 0.005$, $\mathrm{Var}(\phi_{y,i}^{-1}) = 0.005^2$, and $\mathrm{E}(\phi_{\theta,j}^{-1}) = 0.5$, $\mathrm{Var}(\phi_{\theta,j}^{-1}) = 0.5^2$ on the observation and state evolution variances.

The joint posterior of the states and model parameters is approximated by Gibbs sampling. Sampling the states can be done using the FFBS algorithm. The full conditionals for the parameters are as follows.

- The full conditional of $\phi_{\theta,j}$ is

$$\phi_{\theta,j}|\ldots \sim \mathcal{G}\left(\alpha_{\theta,j} + \frac{T}{2}, b_{\theta,j} + \frac{1}{2}SS_{\theta,j}\right), \qquad j = 1,2,3$$

  with $SS_{\theta,j} = \sum_{t=1}^{T}(\theta_{j,t} - (G\theta_{t-1})_j)^2$;
- the full conditional of the precision ϕ_{y_i} is

$$\phi_{y,i}|\ldots \sim \mathcal{G}\left(\alpha_{y,i} + \frac{T}{2}, b_{y,i} + \frac{1}{2}SS_{y,i}\right), \qquad i = 1,\cdots,m$$

  with $SS_{y,i} = \sum_{t=1}^{T}(y_{i,t} - (F\theta_t)_i)^2$;
- the full conditional of the AR parameters ψ_j is, by the theory of linear regression with a Normal prior,

$$\psi_j|\ldots \sim \mathcal{N}(\psi_{j,T}, \tau_{j,T}), \qquad j = 1,2,3$$

  where

$$\psi_{j,T} = \tau_{j,T}\left[\phi_{\theta,j}\sum_{t=1}^{T}\theta_{j,t-1}\theta_{j,t} + \frac{1}{\tau_{j,0}}\psi_0\right]$$

$$\tau_{j,T} = \left[\frac{1}{\tau_0} + \phi_{\theta,j}\sum_{t=1}^{T}\theta_{j,t-1}^2\right]^{-1}.$$

R code

```
> mod <- dlm(m0 = rep(0, p), C0 = 100 * diag(p),
+            FF = X, V = diag(m), GG = diag(p), W = diag(p))
> ##  Prior hyperparameters
> psi0 <- 0; tau0 <- 1
> shapeY <- 3; rateY <- .01
> shapeTheta <- 3; rateTheta <- 1
> ## MCMC
> MC <- 10000
> gibbsTheta <- array(NA, dim = c(MC, TT + 1, p))
> gibbsPsi <- matrix(NA, nrow = MC, ncol = p)
> gibbsV <- matrix(NA, nrow = MC, ncol = m)
> gibbsW <- matrix(NA, nrow = MC, ncol = p)
> ## Starting values: as specified by mod
> set.seed(3420)
> phi.init <- rnorm(3, psi0, tau0)
```

```
> V.init <- 1 / rgamma(1, shapeY, rateY)
> W.init <- 1 / rgamma(1, shapeTheta, rateTheta)
> mod$GG <- diag(phi. init)
> mod$V <- diag(rep(V.init, m))
> mod$W <- diag(rep(W.init, p))
> ## Gibbs sampler
> for (i in 1 : MC)
+ {
+     ## generate the states by FFBS
+     modFilt <- dlmFilter(y, mod, simplify = TRUE)
+     theta <- dlmBSample(modFilt)
+     gibbsTheta[i, , ] <- theta
+     ## generate the W_j
+     theta.center <- theta[-1, ] -
+         t(mod$GG %*% t(theta[-(TT + 1), ]))
+     SStheta <- apply((theta.center)^2, 2, sum)
+     phiTheta <- rgamma(p, shape =shapeTheta + TT / 2,
+                        rate = rateTheta + SStheta / 2)
+     gibbsW[i, ] <- 1 / phiTheta
+     mod$W <- diag(gibbsW[i, ])
+     ## generate the V_i
+     y.center <- y - t(mod$FF %*% t(theta[-1, ]))
+     SSy <- apply((y.center)^2, 2, sum)
+     gibbsV[i, ] <- 1 / rgamma(m, shape = shapeY + TT / 2,
+                               rate = rateY + SSy / 2)
+     mod$V <- diag(gibbsV[i, ])
+     ## generate the AR parameters psi_1, psi_2, psi_3
+     psi.AR <- rep(NA, 3)
+     for (j in 1 : p)
+     {
+         tau <- 1 / ((1 / tau0) + phiTheta[j] *
+                     crossprod(theta[-(TT + 1), j]))
+         psi <- tau * (phiTheta[j] * t(theta[-(TT + 1), j]) %*%
+                       theta[-1, j] + psi0 / tau0)
+         psi.AR[j] <- rnorm(1, psi, sd = sqrt(tau))
+     }
+     gibbsPsi[i, ] <- psi.AR
+     mod$GG <- diag(psi.AR)
+ }
```

We generate a sample of size 10000 and discard the first 1000 draws as burn in. Diagnostic plots (not shown) indicate convergence of the MCMC chain. Below are the MCMC approximations of the posterior expectations of the AR parameters and unknown variances, with their Monte Carlo standard errors.

R code

```
> burn <- 1000
> round(mcmcMean(gibbsPsi[-(1 : burn), ]), 4)
    V.1      V.2      V.3
  0.9949   0.9883   0.9135
 (0.0000) (0.0001) (0.0003)
> round(mcmcMean(sqrt(gibbsW[-(1 : burn), ])), 4)
    V.1      V.2      V.3
  0.3137   0.3220   0.6493
 (0.0112) (0.0115) (0.0258)
> round(mcmcMean(sqrt(gibbsV[-(1 : burn), ])), 4)
    V.1      V.2      V.3      V.4      V.5      V.6
  0.1450   0.0631   0.0600   0.0851   0.0932   0.0755
 (0.0094) (0.0055) (0.0030) (0.0041) (0.0040) (0.0032)
    V.7      V.8      V.9      V.10     V.11     V.12
  0.0565   0.0520   0.0284   0.0469   0.0626   0.0807
 (0.0027) (0.0021) (0.0017) (0.0023) (0.0026) (0.0031)
    V.13     V.14     V.15     V.16     V.17
  0.0873   0.0665   0.0490   0.0503   0.0837
 (0.0032) (0.0027) (0.0026) (0.0030) (0.0034)
```

The MCMC smoothing estimates of $\theta_t = (\beta_{1,t}, \beta_{2,t}, \beta_{3,t})$ are plotted in Figure 4.19. The results are quite close to the OLS month-by-month estimates, also shown in the plot. However, this is not always the case. Roughly speaking, while the cross-sectional OLS estimates minimize the residual sum of squares at each t, in the DLM the minimization is subject to the constraints implied by the state equation. In fact, a poor specification of the state equation might result in an unsatisfactory fit of the cross-sectional data. Further developments of this example include a more thoughtful specification of the state equation, time varying observation and/or evolution variances, and the inclusion of macroeconomic variables.

4.6.3 Factor models

In this example, we use a factor model (Section 3.3.6) to extract a common stochastic trend from multiple integrated time series. The basic idea is to explain fluctuations of various markets or common latent factors that affect a set of economic or financial variables simultaneously. The data are the federal funds rate (short rate) and 30-year conventional fixed mortgage rate (long rate), obtained from Federal Reserve Bank of St. Louis[2] (see Chang et al. (2005)). The series are sampled at weekly intervals over the period April 7, 1971 through September 8, 2004. We will work with the natural logarithm of one plus the interest rate; the transformed data are plotted in Figure 4.21.

[2] Source: `http://research.stlouisfed.org/fred2/`

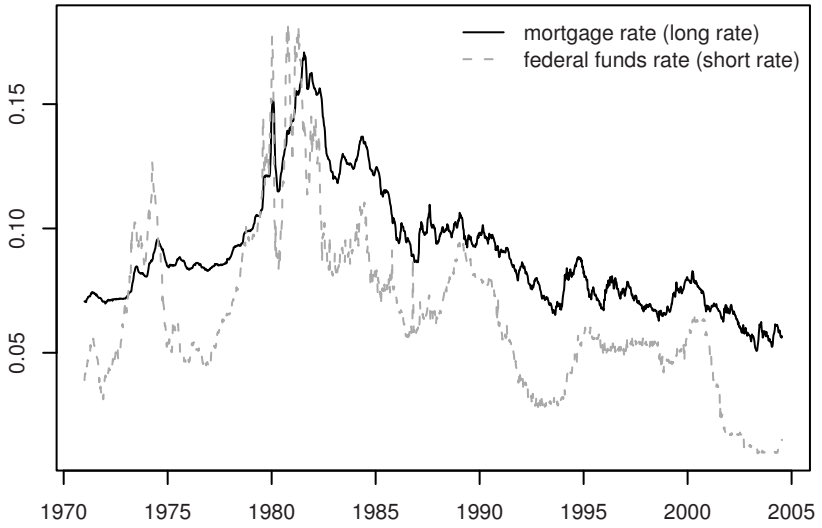

Fig. 4.21. Log of one plus the federal funds rate (FF) and the 30-year mortgage rate (WMORTG). Weekly data, 1971-2004

R code

```
> interestRate <- read.table("interestRates.dat",
+                            col.names = c("Long", "Short"))
> y <- log(1 + interestRate / 100)
> y <- ts(y, frequency = 52, start = 1971)
> ts.plot(y, lty = c(1, 2), col = c(1, "darkgray"))
> legend("topright", legend = c("mortgage rate (long rate)",
+        "federal funds rate (short rate)"),
+        col = c(1, "darkgray"), lty = c(1,2), bty = "n")
```

To extract a common stochastic trend in the bivariate time series $(Y_t = (Y_{1,t}, Y_{2,t}) : t \geq 1)$, we assume the following factor model:

$$\begin{cases} Y_t = A\mu_t + \mu_0 + v_t, & v_t \sim \mathcal{N}(0, V), \\ \mu_t = \mu_{t-1} + w_t, & w_t \sim \mathcal{N}(0, \sigma_\mu^2), \end{cases} \tag{4.26}$$

where the 2×1 matrix A is set to be $A = \begin{bmatrix} 1 & \alpha \end{bmatrix}'$ to ensure identifiability and $\mu_0 = \begin{bmatrix} 0 & \bar{\mu} \end{bmatrix}'$. The latent variable μ_t is interpreted as the common stochastic trend, here simply modeled as a random walk. In the usual DLM notation the model can be written in the form

$$\begin{aligned} Y_t &= F\theta_t + v_t, \\ \theta_t &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \theta_{t-1} + \begin{bmatrix} w_t \\ 0 \end{bmatrix}, \end{aligned} \tag{4.27}$$

with

$$\theta_t = \left[\mu_t, \bar{\mu}\right], \quad F = \begin{bmatrix} 1 & 0 \\ \alpha & 1 \end{bmatrix}, \quad V = \begin{bmatrix} \sigma_1^2 & \sigma_{12} \\ \sigma_{12} & \sigma_2^2 \end{bmatrix}, \quad W = \text{diag}(\sigma_\mu^2, 0).$$

We use a $\mathcal{N}(\alpha_0, \tau^2)$ prior for α and, as in Sections 4.5.1 and 4.5.2, independent Gamma and Wishart priors for the precision $1/\sigma_\mu^2$ and V^{-1},

$$\sigma_\mu^{-2} \sim \mathcal{G}(a, b), \quad V^{-1} \sim \mathcal{W}(\nu_0, S_0).$$

In the specific case illustrated below, we take $\alpha_0 = 0, \tau^2 = 16$; a and b such that $\text{E}(\sigma_\mu^{-2}) = 0.01$ with $\text{Var}(\sigma_\mu^{-2}) = 1$; $\nu_0 = (\delta + 1)/2 = 2$ and $S_0 = V_0/2$, where

$$V_0 = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 4 \end{bmatrix},$$

so that $\text{E}(V) = V_0$. The posterior distribution of the states and the parameters, given $y_{0:T}$, is proportional to

$$\begin{aligned} &\prod_{t=1}^{T} \mathcal{N}_2((y_{1,t}, y_{2,t}); (\mu_t, \alpha\mu_t + \bar{\mu})', V)\, \mathcal{N}(\bar{\mu}; m_{0,2} C_{0,22}) \\ &\mathcal{N}(\mu_t; \mu_{t-1}, \sigma_\mu^2)\, \mathcal{N}(\alpha; \alpha_0, \tau^2)\, \mathcal{G}(\sigma_\mu^{-2}; a, b)\, \mathcal{W}(V^{-1}; \nu_0, S_0), \end{aligned}$$

where $m_{0,2}$ and $C_{0,22}$ are the prior mean and variance for $\bar{\mu}$. The posterior is approximated via Gibbs sampling;

- the full conditional of α is $\mathcal{N}(\alpha_T, \tau_T^2)$, where

$$\begin{aligned} \tau_T^2 &= \frac{(1-\rho^2)\tau^2\sigma_2^2}{\tau^2 \sum_{t=1}^{T} \mu_t^2 + (1-\rho^2)\sigma_2^2} \\ \alpha_T &= \tau_T^2 \, \frac{\tau^2/\sigma_2 \sum_{t=1}^{T} (\frac{y_{2,t}-\bar{\mu}}{\sigma_2} - \rho \frac{y_{1,t}-\mu_t}{\sigma_1})\, \mu_t + \alpha_0(1-\rho^2)}{\tau^2\,(1-\rho^2)}, \end{aligned}$$

  with $\rho = \sigma_{12}/(\sigma_1\sigma_2)$;
- the full conditional of the precision σ_μ^{-2} is

$$\mathcal{G}\big(a + \frac{T}{2}, b + \frac{SS_\mu}{2}\big),$$

  where $SS_\mu = \sum_{t=1}^{T} (\mu_t - \mu_{t-1})^2$;
- the full conditional of V^{-1} is

$$\mathcal{W}\bigg(\frac{\delta + 1 + T}{2}, \frac{1}{2}(V_0 + \sum_{t=1}^{T} (y_t - F\theta_t)(y_t - F\theta_t)^{'})\bigg).$$

R code

```
> # Prior hyperparameters
> alpha0 <- 0; tau2 <- 16
> expSigmaMu<- 0.01; varSigmaMu<- 1
> a <- (expSigmaMu^2/varSigmaMu)+2; b <- expSigmaMu*(a-1)
> delta <- 3;
> V0=matrix(c(1, 0.5, 0.5, 4), byrow=T, nrow=2)
> # Gibbs sampling
> MC <- 10000
> n <- nrow(y)
> gibbsTheta <- array(0, dim=c(n+1,2,MC-1))
> gibbsV <-array(0, dim=c(2,2,MC))
> gibbsAlpha <- rep(0,MC)
> gibbsW <- rep(0,MC)
> # model and starting values for the MCMC
> mod <- dlmModPoly(2, dW=c(1,0), C0=100*diag(2))
> gibbsAlpha[1] <- 0
> mod$FF <- rbind(c(1,0), c(gibbsAlpha[1],1))
> mod$W[1,1] <- gibbsW[1] <- 1/rgamma(1, a, rate=b)
> mod$V <- gibbsV[,,1] <- V0 /(delta-2)
> mod$GG <- diag(2)
> # MCMC loop
> for(it in 1:(MC-1))
+ {
+ # generate state- FFBS
+ modFilt <- dlmFilter(y, mod, simplify=TRUE)
+    gibbsTheta[,,it] <- theta <- dlmBSample(modFilt)
+ # update alpha
+ rho <- gibbsV[,,it][1,2]/(gibbsV[,,it][1,1]*gibbsV[,,it][2,2])^.5
+ tauT <- (gibbsV[,,it][2,2]*(1-rho^2)*tau2) /
+         (tau2 * sum(theta[-1,1]^2)+(1-rho^2)*gibbsV[,,it][2,2])
+ alphaT <- tauT * ((tau2/(gibbsV[,,it][2,2])^.5) *
+           sum(((y[,2]-theta[-1,2])/gibbsV[,,it][2,2]^.5 -
+           rho* (y[,1]-theta[-1,1])/gibbsV[,,it][1,1]^.5) *
+           theta[-1,1])+ alpha0*(1-rho^2))/(tau2*(1-rho^2))
+ mod$FF[2,1] <- gibbsAlpha[it+1] <- rnorm(1, alphaT, tauT^.5)
+ # update sigma_mu
+ SSmu <- sum( diff(theta[,1])^2)
+ mod$W[1,1] <- gibbsW[it+1] <- 1/rgamma(1, a+n/2, rate=b+SSmu/2)
+ # update V
+ S <- V0 + crossprod(y- theta[-1,] %*% t(mod$FF))
+ mod$V <- gibbsV[,,it+1] <- solve(rwishart(df=delta+1+ n,
+            Sigma=solve(S)))
+ }
```

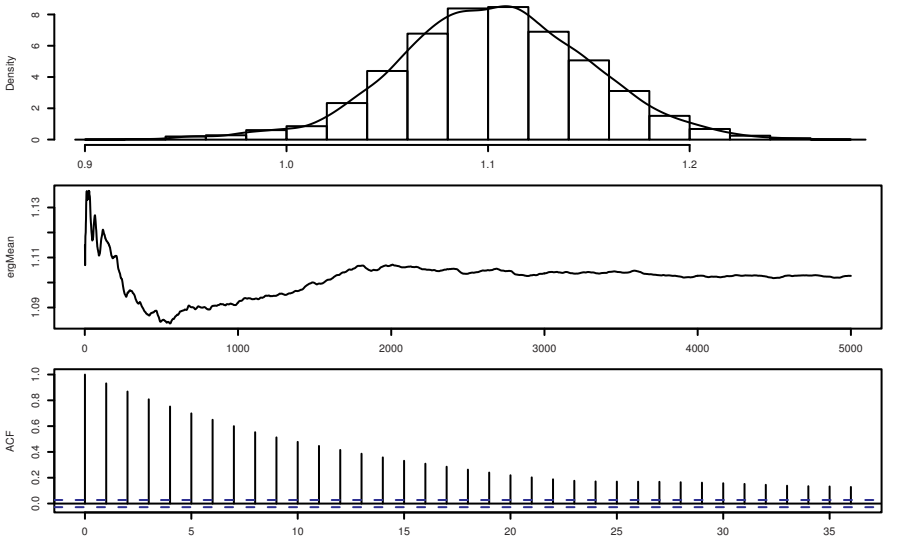

Fig. 4.22. MCMC estimate of the posterior distribution of α; running ergodic means; MCMC autocorrelation

We show the results for 10000 MCMC iterations, with a burn in of 5000 draws. Some diagnostic plots are summarized in Figures 4.22-4.23; the first panel in each figure shows the MCMC approximation of the posterior distribution of α and σ_μ.

The MCMC estimates of the parameters' posterior means are given in the display below, together with the estimated standard errors, obtained by the `dlm` function `mcmcMeans`, and the 5% and 95% quantiles of their marginal posterior distributions.

parameter	α	σ_μ	σ_1	σ_2	$\sigma_{1,2}$
posterior mean	1.1027	0.0084	0.0261	0.0512	0.00037
(st dev)	(0.0036)	(0.00001)	(0.00001)	(0.00001)	(0.0000)
5% quantile	1.0262	0.0079	0.0254	0.0498	0.0003
95% quantile	1.1806	0.0089	0.0269	0.0526	0.00043

Figure 4.24 shows the data and the posterior mean of the common stochastic trend, which results very close to the long rate.

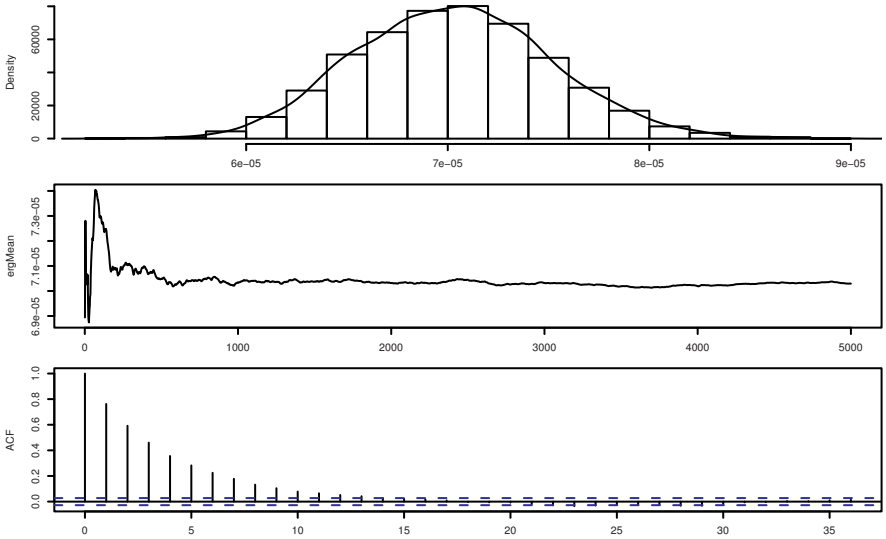

Fig. 4.23. MCMC estimate of the posterior distribution of σ_μ; running ergodic means; MCMC autocorrelation

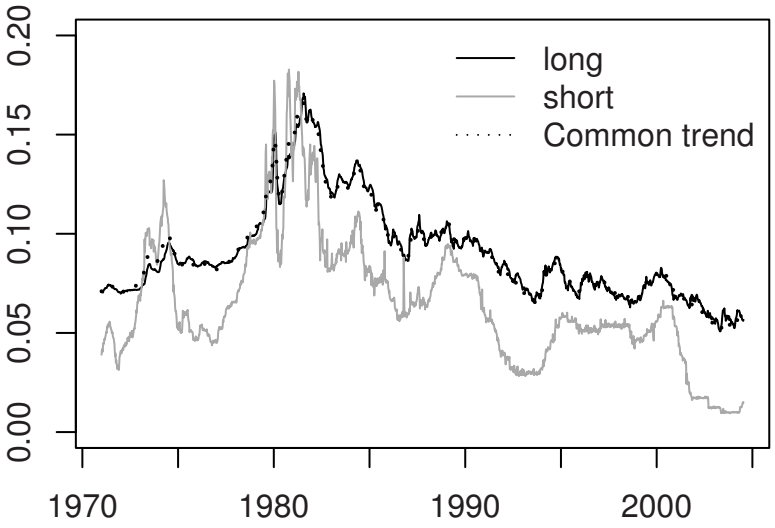

Fig. 4.24. Posterior mean of the common stochastic trend

Problems

4.1. In Chapter 2, we considered a random walk-plus-noise model for the Nile river data. There, we used the maximum likelihood estimates for the state and observation variances. Consider now Bayesian inference on the states and the unknown parameters of the model. Express conjugate priors for V and W and evaluate the posterior distribution of $(\theta_{0:T}, V, W|y_{1:T})$. Then, estimate the model using discount factors as in Sections 4.3.2 and 4.3.3, and compare the results.

4.2. Consider the DLM described in Section 4.3.1, with conjugate priors. Suppose for simplicity that θ_t is univariate. Compute the expression of $(1-\alpha)$ probability intervals for $\theta_t|y_{1:t}$. Discuss the results, comparing with the case where σ^2 is known.

4.3. Consider again the Nile data, which we modeled as a random walk plus noise (see Problem 4.1). In fact, assuming time-invariant variances is too restrictive for these data: we expect big changes caused by the dam's construction. The model described in Section 4.5.3 allows for outliers and structural breaks. Provide Bayesian estimates of this model for the Nile data (compute an MCMC approximation of the joint posterior of the states and unknown parameters, given $y_{1:T}$).

4.4. Show that the posterior distribution (4.20) is invariant for the hybrid sampler described in Algorithm 4.3.

4.5. A honest elicitation of a prior distribution is often difficult. At least, one should be aware of the role of prior hyperparameters, and the consequent sensitivity of inference to the prior assumptions (given the model, which in fact is also part of the prior assumptions). In Section 4.5.2, we consider an Inverse-Wishart prior on the random covariance matrix. Suppose that V is a (2×2) random matrix having an Inverse-Wishart distribution with parameters $(\alpha = n/2, B = \Sigma/2)$. Study the distribution for varying values of the parameters n and Σ, and plot the resulting marginal densities.

4.6. Study sensitivity to prior assumptions for the SUTSE model illustrated in Section 4.5.2.