STATS 207: Time Series Analysis
Autumn 2020

Lecture 17: Stochastic Volatility, Bayesian Analysis of State-Space Models.

Dr. Alon Kipnis

November 9th 2020

Genera Info

- HW4 is out. Due on Monday 11/16/2020.
- On Wednesday: Professor David Donoho will talk about "Bootstrap Reality Check & Technical Trading Rules"
- On Monday 11/16/2020: Prophet talk and workshop with Sean Taylor.

Final Assessment

- A mix of True/False and Fill-in-the-blacks questions + short explanations.
- Covers **all class material** until the end of this week.
- Can only **improve** your final grade.
- 2 hours limit. You can **Start** anytime between 11/16 (evening) - 11/20 (10:00pm).
- The **Stanford Honor Code** applies.

Final Assessment (cont'd)

- Examples for **True/False questions**:

  EXM 1: Second differencing transforms an $IMA(1,3)$ process into a stationary process. Explanation:

  _____.

  EXM 2: The Kalman smoother produces the best linear mean-squared error estimation of the unobserved state x_t at each time $1 \leq t \leq n$, given all the information $\{y_s,\ 1 \leq s \leq n\}$. Explanation:

  _____.

- Examples for **fill-in-the-blanks** questions:

  EXM 3: Consider an AR(1) model $x_t = -x_{t-1}/2 + w_t$, where (w_t) is white noise with $\mathrm{Var}(w_t) = 1$. The optimal linear estimator of x_t given $x_{t-1}, x_{t-2}, \ldots$ is _____. Explanation:

  _____.

  EXM 4: Let $f_y(\omega)$ denote the spectrum of the filtered standard white noise $y_t = (w_t + w_{t-1} + w_{t-2} + w_{t-3})/4$, where $\mathrm{Var}(w_t) = 1$. Then $f_y(\omega) =$ _____. Explanation:

  _____.

Outline

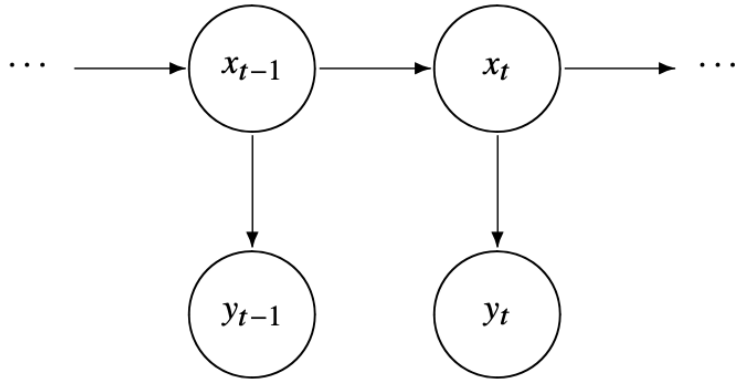
Stochastic Volatility

Bayesian Analysis of State-Space Models

State-Space Model (review)

- **State Equation**:

$$\boldsymbol{x}_t = \Phi \boldsymbol{x}_{t-1} + \Upsilon \boldsymbol{u}_t + \boldsymbol{w}_t,$$



where

  - $\boldsymbol{x}_t$, $\boldsymbol{w}_t$ have dimension p,
  - $\boldsymbol{u}_t$ has dimension r,
  - $\boldsymbol{w}_t \overset{iid}{\sim} \mathcal{N}(0, Q)$.

- **Observation Equation**:

$$\boldsymbol{y}_t = A_t \boldsymbol{x}_t + \Gamma \boldsymbol{u}_t + \boldsymbol{v}_t$$

  - $\boldsymbol{y}_t$, $\boldsymbol{v}_t$ have dimension q,
  - $\boldsymbol{v}_t \overset{iid}{\sim} \mathcal{N}(0, R)$.

- **Initial Conditions**: $\boldsymbol{x}_0 \sim \mathcal{N}(\mu_0, \Sigma_0)$.

Stochastic Volatility

Volatility Modeling

- P_t is the **price of an asset** at time t (e.g., AAPL, AAPL, GOOG, IBM, SPY).
- Financial **returns**:

$$\tilde{r}_t \equiv \frac{P_t - P_{t-1}}{P_{t-1}}$$

- Financial **"log-returns"**:

$$r_t = \nabla \log(P_t) = \log(P_t/P_{t-1}) \approx \tilde{r}_t.$$

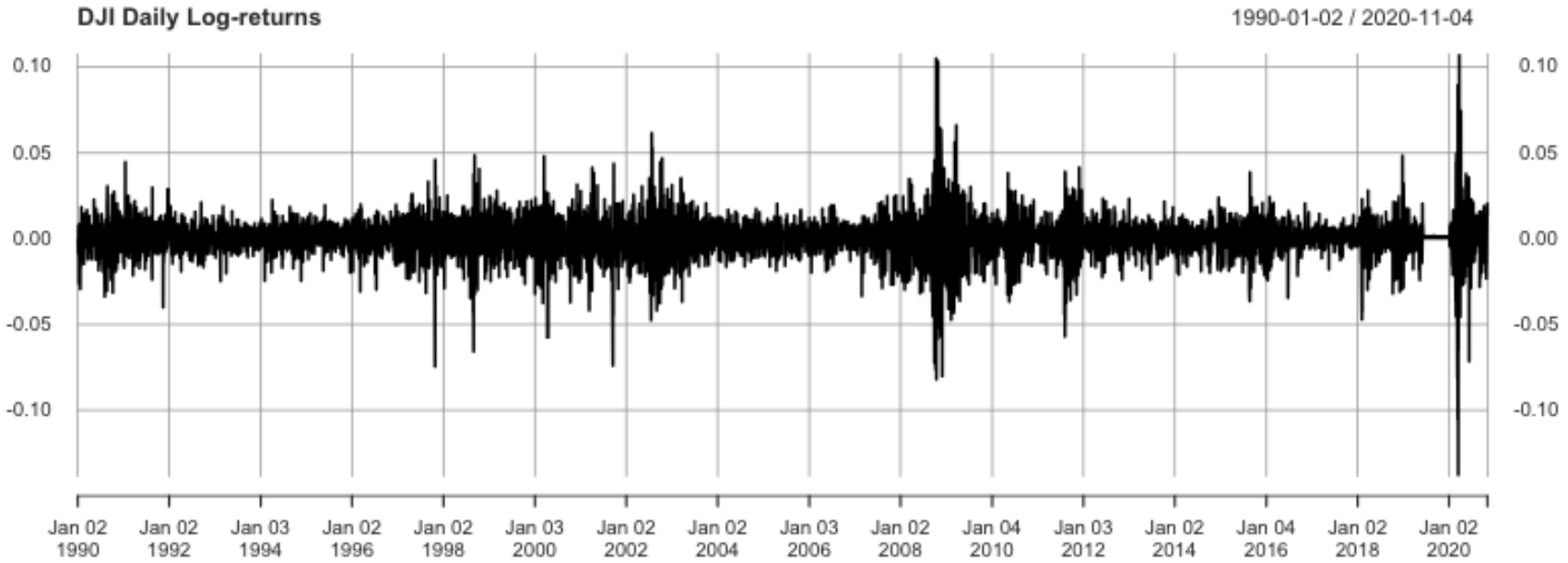
- **Return model**:

$$r_t = \beta \sigma_t \epsilon_t,$$

where:

  - (σ_t) is a **non-negative volatility process**.
  - $\epsilon_t \overset{iid}{\sim} N(0,1)$ is independent of σ_s for $s \leq t$.

Volatility Modeling – Motivation



- **Observation**: High-volatility days are **clumped together**.

Volatility Modeling

- **Stochastic volatility model** (Taylor 1982):

$$x_t = \phi x_{t-1} + w_t, \qquad x_t = \log \sigma_t^2$$
$$r_t = \beta \sigma_t \epsilon_t = \beta \exp(x_t/2)\epsilon_t$$

  where $w_t \overset{iid}{\sim} \mathcal{N}(0, \sigma_w^2)$.

- Compare to ARCH(1):

$$\sigma_t^2 = \alpha_0 + \alpha_1 r_{t-1}^2$$

Stochastic Volatility State-Space Model, I

- **Linearize observations**:

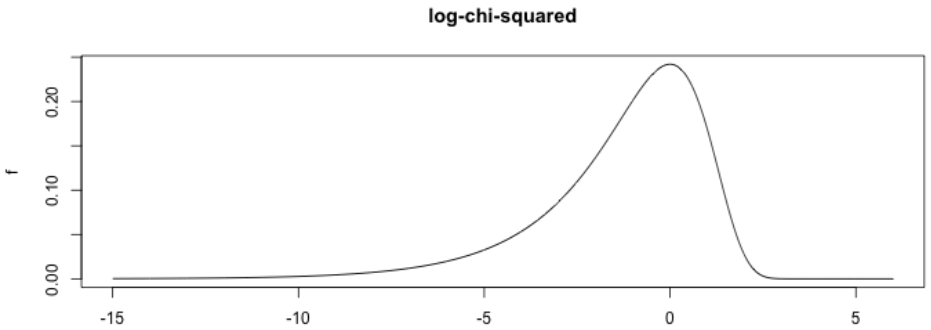
$$y_t = \log r_t^2 = \alpha + x_t + v_t, \qquad v_t = \log \epsilon_t^2, \qquad \alpha = \log \beta^2.$$

- Full **state dynamics**:

$$x_t = \phi_0 + \phi_1 x_{t-1} + w_t.$$

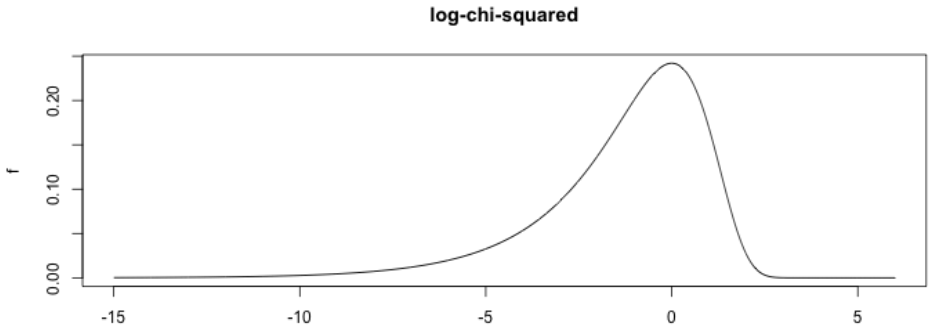
- Warning: Not a **Gaussian** state-space model because $v_t = \log \epsilon_t^2$ is log-chi-squared with 1 DoF:

$$f_{\log(\chi_1^2)}(v) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}(e^v - v)}, \qquad v \in \mathbb{R}.$$



$$\mathbb{E}\left[\log(\chi_1^2)\right] = -1.27, \qquad \mathrm{Var}(\log(\chi_1^2)) = \pi^2/2$$

Handling non-Gaussian Noise



- **Approximate** $\log(\chi_1^2)$ using a **Gaussian mixture** (Kim Shephard and Chib 1998):

$$v_t \approx \sum_{j=1}^{7} I_{t,j} z_{tj}, \qquad z_{tj} \overset{iid}{\sim} (\mu_j, \sigma_j^2), \qquad \Pr(I_{t,j} = 1) = \pi_j$$

  equivalently

$$\log(\chi_1^2) \overset{D}{\approx} \sum_{j=1}^{7} \pi_j \mathcal{N}(\mu_j, \sigma_j^2).$$

- This is a switching model!
- In the next example we use a **binary Gaussian mixture**:

$$\log(\chi_1^2) \overset{D}{\approx} \pi \mathcal{N}(0, \sigma_0^2) + (1 - \pi)\mathcal{N}(\mu_1, \sigma_1^2).$$

Stochastic Volatility State-Space Model, II

- **Approximate** $\log(\chi_1^2)$ using a **binary Gaussian mixture**:

$$\log(\chi_1^2) \overset{D}{\approx} \pi\mathcal{N}(0, \sigma_0^2) + (1-\pi)\mathcal{N}(\mu_1, \sigma_1^2).$$

- Can be written as a **switching state-space model**:
  - **State Dynamics**:

$$\begin{bmatrix} x_t \\ z_{t0} \\ z_{t1} \end{bmatrix} = \begin{bmatrix} \phi_1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_{t-1} \\ z_{t-1,0} \\ z_{t-1,1} \end{bmatrix} + \begin{bmatrix} \phi_0 \\ 0 \\ \mu_1 \end{bmatrix} + \begin{bmatrix} w_t \\ v_{0t} \\ v_{1t} \end{bmatrix}$$

  - **Observation Equation**:

$$y_t = A_t \boldsymbol{x}_t + [\alpha] + [0], \qquad A_t \in \{M_0, M_1\}$$

$$M_1 = \begin{bmatrix} 1 & 1 & 0 \end{bmatrix}, \qquad M_2 = \begin{bmatrix} 1 & 0 & 1 \end{bmatrix}.$$

  - **Covariance**:

$$Q = \mathrm{diag}(\sigma_w^2, \sigma_0^2, \sigma_1^2).$$

Stochastic Volatility State-Space Model – Filtering Equations

- Filtering of **Switching State-Space Model** applies:

$$x_{t+1}^{t} = \phi_0 + \phi_1 x_t^{t-1} + \sum_{j=0}^{1} \pi_{tj} K_{tj} \epsilon_{tj},$$

$$P_{t+1}^{t} = \phi_1^2 P_t^{t-1} + \sigma_w^2 - \sum_{j=0}^{1} \pi_{tj} K_{tj}^2 \Sigma_{tj},$$

$$\epsilon_{t0} = y_t - \alpha - x_t^{t-1} - \mu_0, \qquad \Sigma_{t0} = P_t^{t-1} + \sigma_0^2, \qquad K_{t0} = \phi_1 P_t^{t-1} / \Sigma_{t0},$$

$$\epsilon_{t1} = y_t - \alpha - x_t^{t-1} - \mu_1, \qquad \Sigma_{t1} = P_t^{t-1} + \sigma_1^2, \qquad K_{t1} = \phi_1 P_t^{t-1} / \Sigma_{t1}.$$

- **Filtered probabilities**:
  - $\pi_{t1} = \Pr(A_t = M_1 | y_{1:t})$, $\pi_{t0} = 1 - \pi_{t1}$, and

$$\pi_{t1} = \Pr(A_t = M_1 | y_{1:t}) = \frac{\pi_1 \mathbf{P}_1(t|t-1)}{\pi_0 \mathbf{P}_0(t|t-1) + \pi_1 \mathbf{P}_1(t|t-1)}.$$

  - π_0 and π_1 have been **specified a priori** (e.g. $\pi_0 = \pi_1 = .5$).
  - $\mathbf{P}_j(t|t-1) = f_{y_t|y_{1:t-1},A_t}(y_t|y_{1:t-1}, M_j)$, $j = 0, 1$.

Stochastic Volatility State-Space Model – Estimation

- **Parameters**:

$$\Theta = \left(\phi_0, \phi_1, \sigma_w^2, \sigma_0^2, \mu_1, \sigma_1^2, \alpha\right)',$$

- **Gaussian approximation** to conditional density of the observations:

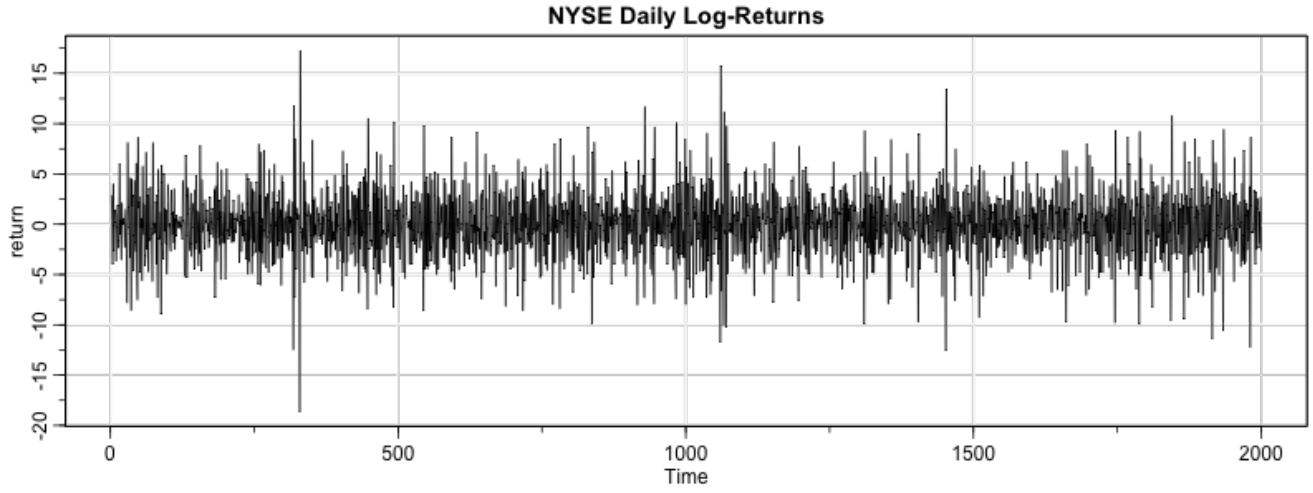
$$\mathbf{P}_j(t|t-1) = f_{y_t|y_{1:t-1},A_t}(y_t|y_{1:t-1}, M_j) \approx \mathcal{N}(y_t|x_t^{t-1} + \mu_j, \Sigma_{tj}).$$

- Log-likelihood:

$$\log L(y_{1:n}, \Theta) = \sum_{t=1}^{n} \log \left(\sum_{j=0}^{1} \pi_j \mathcal{N}(y_t|x_t^{t-1} + \mu_j, \Sigma_{tj}) \right)$$

- Solve using Newton-Raphson or EM (for EM need to write $L(y_{1:n}, x_{1:n}, \Theta)$).

Example 6.23: Analysis of NYSE Returns, I




```
y    = log(nyse^2)
num  =length(y)

# Initial Parameters
phi0=0; phi1=.95; sQ=.2; alpha=mean(y); sR0=1; mu1=-3; sR1=2
init.par = c(phi0,phi1,sQ,alpha,sR0,mu1,sR1)

# Innovations Likelihood
Linn = function(para){
  phi0=para[1]; phi1=para[2]; sQ=para[3]; alpha=para[4]
  sR0=para[5]; mu1=para[6]; sR1=para[7]
  sv = SVfilter(num,y,phi0,phi1,sQ,alpha,sR0,mu1,sR1)
  #  pi1 = 0.5, pi0 = 0.5, pit1 = 0.5, pit0 = 0.5
  return(sv$like) }
```

Example 6.23: Analysis of NYSE Returns, II

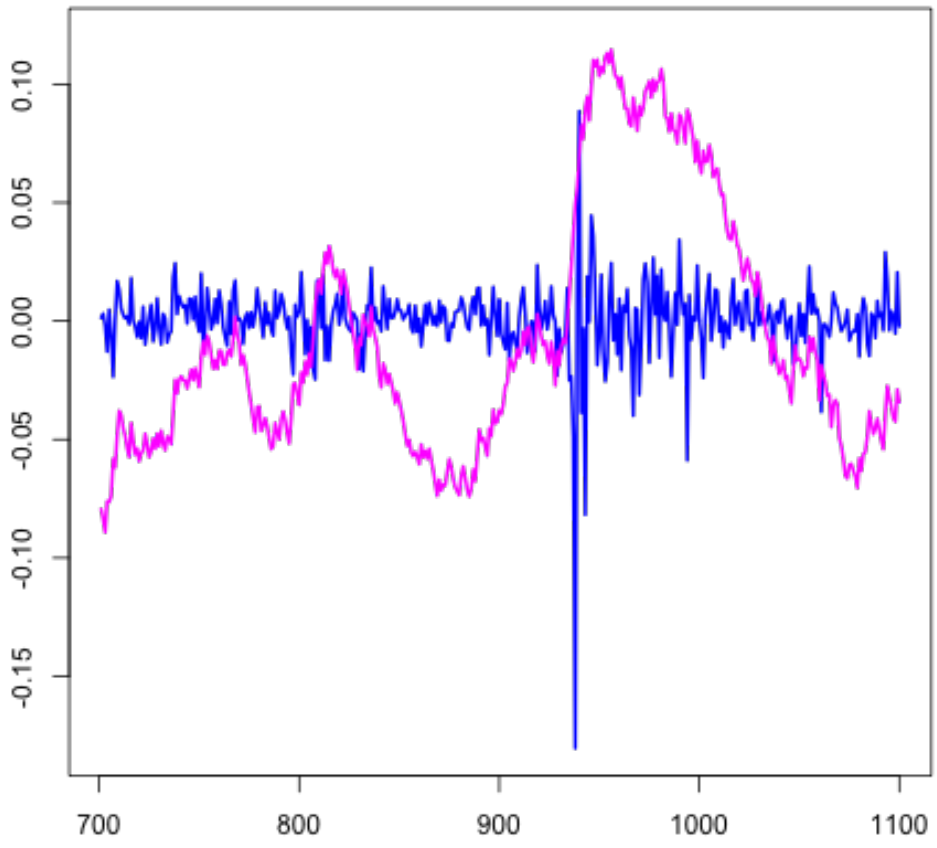
```
# Estimation
(est = optim(init.par, Linn, NULL, method="BFGS", hessian=TRUE,
             control=list(trace=1,REPORT=1)))
SE = sqrt(diag(solve(est$hessian)))
u = cbind(estimates=est$par, SE)
rownames(u)=c("phi0","phi1","sQ","alpha","sigv0","mu1","sigv1"); u
```

	estimates	SE
phi0	-0.005582453	0.019001014
phi1	0.988216120	0.007773915
sQ	0.093354667	0.027643077
alpha	-9.641761238	1.504936367
sigv0	1.217808393	0.064876895
mu1	-2.295367514	0.204367167
sigv1	2.682844415	0.104969139

```
# Graphics   (need filters at the estimated parameters)
phi0=est$par[1]; phi1=est$par[2]; sQ=est$par[3]; alpha=est$par[4]
sR0=est$par[5]; mu1=est$par[6]; sR1=est$par[7]
sv = SVfilter(num,y,phi0,phi1,sQ,alpha,sR0,mu1,sR1)
```

Example 6.23: Analysis of NYSE Returns, III

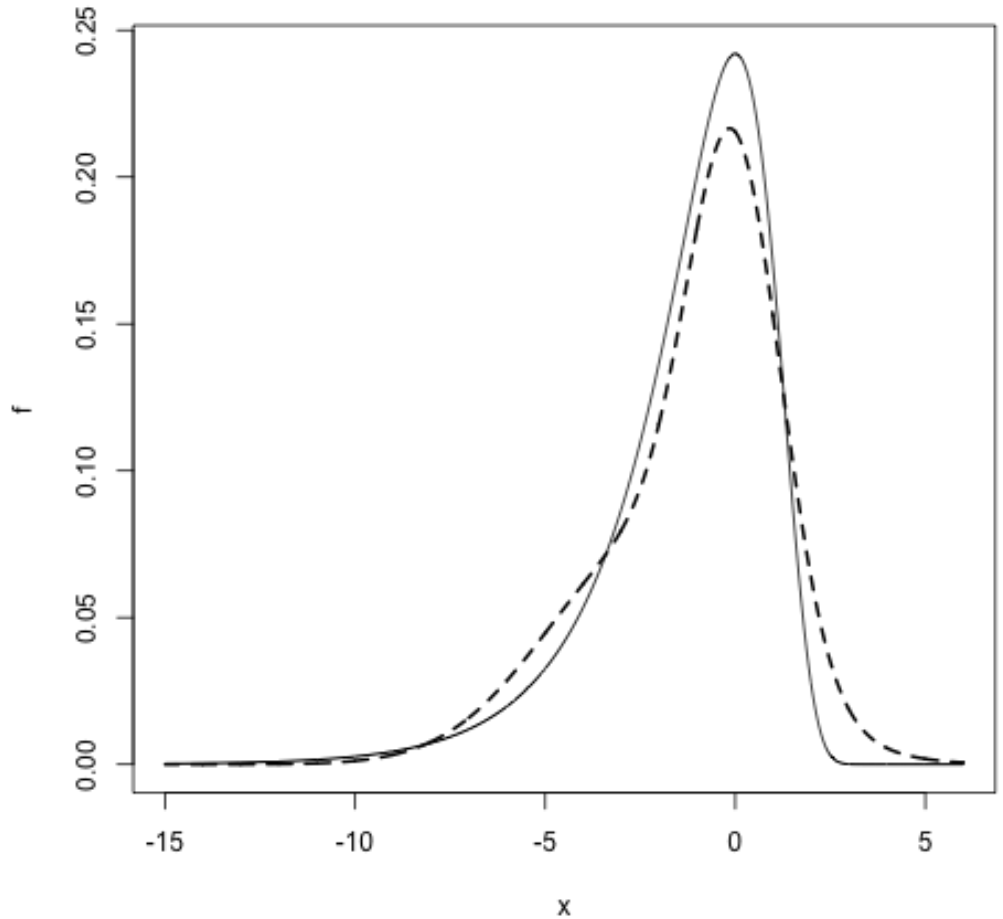
One-Step-Ahead Predicted log-volatility $\hat{x}_t^{t-1}$ (scaled to data range):



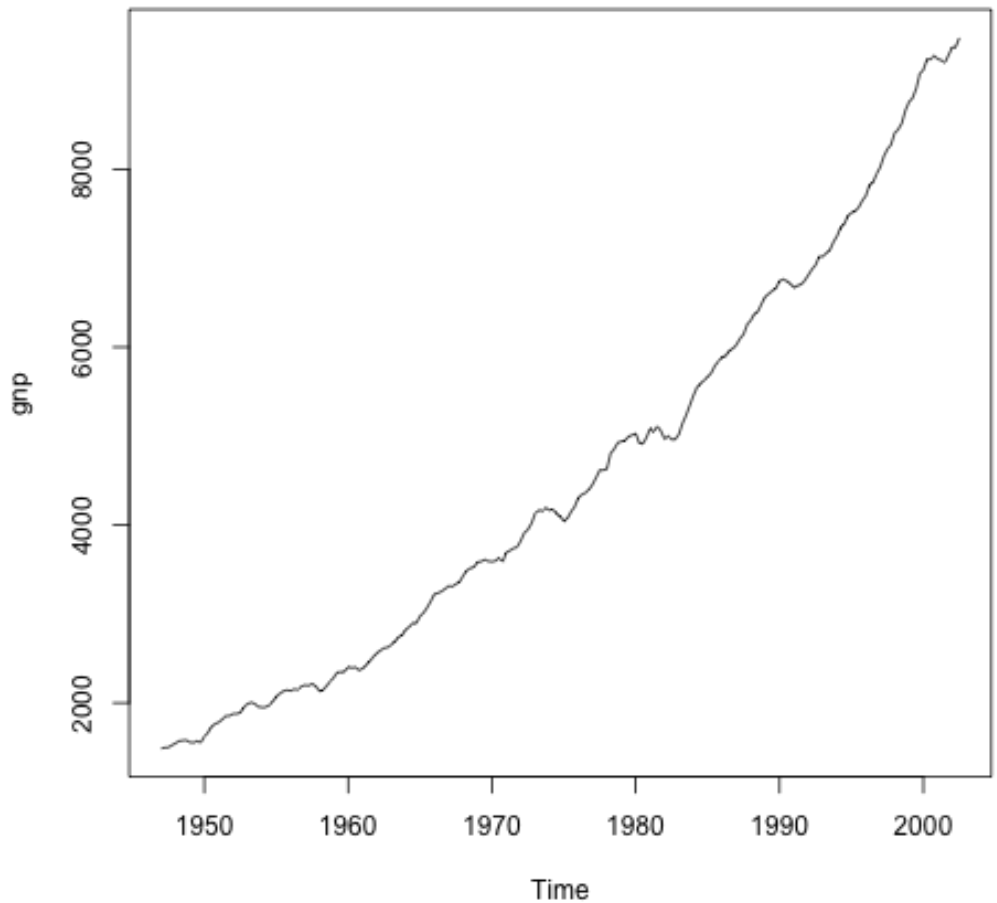
Example 6.23: Analysis of NYSE Returns, IV

Gaussian mixture approximation to log-chi-squared:

$$\log(\chi_1^2) \overset{D}{\approx} \pi\mathcal{N}(0, (1.28)^2) + (1-\pi)\mathcal{N}(-2.3, (2.68)^2)$$



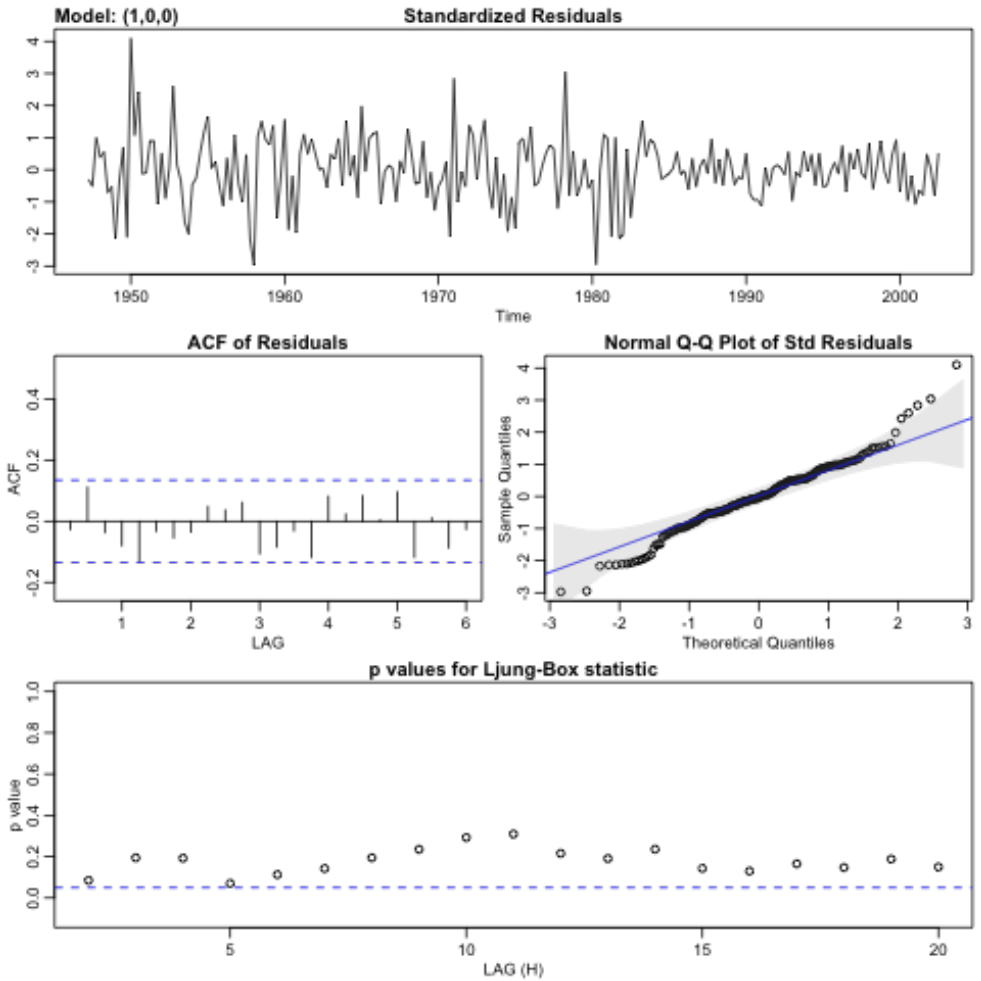
Example 6.24: Analysis of the U.S. GNP Growth Rate



Example 6.24: Analysis of the U.S. GNP Growth Rate

- Previously we fitted AR(1) to differenced log-return of GNP:

```
fit = sarima(diff(log(gnp)), 1, 0, 0)
```

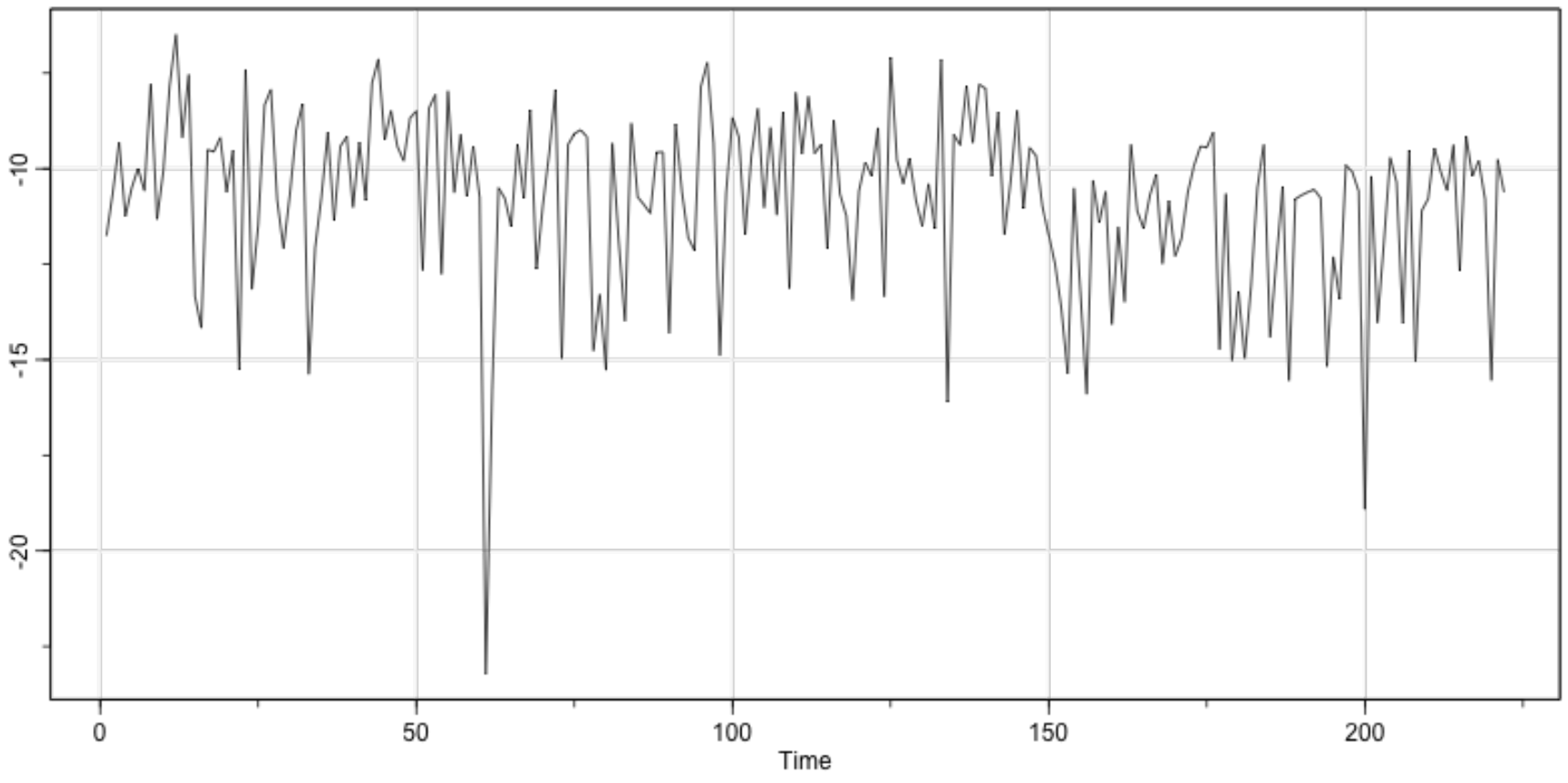


Fit SV model to residuals.

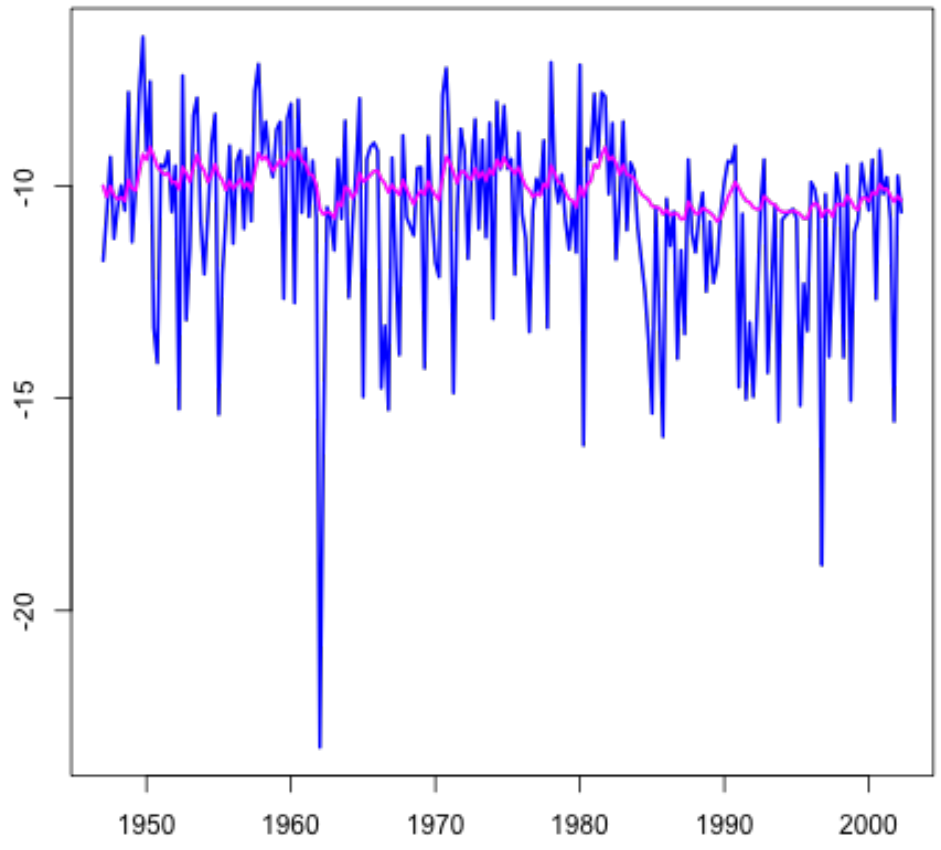
Example 6.24 (cont'd)

Log-Squared residuals:

```
y = as.matrix(log(resid(fit$fit)^2))
tsplot(y, ylab="")
```



Example 6.24 (cont'd)



Bayesian Analysis of State-Space Models

Bayesian Analysis of State-Space Models

- Starting point is a state-space model.
- **Goal**: Obtain **posterior densities**

$$\mathbf{P}(\Theta, \boldsymbol{x}_{0:n} | \boldsymbol{y}_{1:n})$$

  to estimate Θ and $\boldsymbol{x}_{0:n}$.
- **Strategy:** Use a **Markovian updating scheme** to sample from **intractable posterior** distributions.

Example 6.25: Gibbs Sampling for the Bivariate Normal

- Suppose we wish to **sample** from

$$\begin{bmatrix} X \\ Y \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix} \right), \qquad |\rho| < 1,$$

  but only have **conditional marginals**:

$$(X|Y = y) \sim \mathcal{N}(\rho y, 1 - \rho^2), \qquad (Y|X = x) \sim \mathcal{N}(\rho x, 1 - \rho^2).$$

- **Markov Chain**: Set $X^{(0)} = x_0$ and iterate:

$$x_0 \to Y^{(0)} \to X^{(1)} \to Y^{(1)} \to \ldots \to X^{(k)} \to Y^{(k)} \to \ldots,$$

  where

$$(Y^{(k)}|X^{(k)} = x_k) \sim \mathcal{N}(\rho x_k, 1 - \rho^2),$$
$$(X^{(k)}|Y^{(k-1)} = y_{k-1}) \sim \mathcal{N}(\rho y_{k-1}, 1 - \rho^2).$$

- We have

$$\begin{bmatrix} X^{(k)} \\ Y^{(k)} \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \rho^{2k} x_0 \\ \rho^{2k+1} x_0 \end{bmatrix}, \begin{bmatrix} 1 - \rho^{4k} & \rho(1 - \rho^{4k}) \\ \rho(1 - \rho^{4k}) & 1 - \rho^{4k} \end{bmatrix} \right).$$

Gibbs Sampler for State-Space Models

- Property 6.1: **Gibbs Sampler** for State-Space Models
  Iterate:
  (I) Draw $\Theta' \sim \mathbf{P}(\Theta|\boldsymbol{x}_{0:n}, \boldsymbol{y}_{1:n})$
  (II) Draw $\boldsymbol{x}'_{0:n} \sim \mathbf{P}(\boldsymbol{x}_{0:n}|\Theta', \boldsymbol{y}_{1:n})$.
- For (I), use

$$\mathbf{P}(\Theta|\boldsymbol{x}_{0:n}, \boldsymbol{y}_{1:n}) \propto \pi(\Theta)\mathbf{P}(\boldsymbol{x}_0|\Theta)\prod_{t=1}^{n}\mathbf{P}(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}, \Theta)\mathbf{P}(\boldsymbol{y}_t|\boldsymbol{x}_t, \Theta).$$

  Here $\pi(\Theta)$ is the **prior** on Θ (often depends on "hyperparameters").
- For (II), use the Markov structure to write

$$\mathbf{P}(\boldsymbol{x}_{0:n}|\boldsymbol{y}_{1:n}, \Theta) = \mathbf{P}(\boldsymbol{x}_n|\boldsymbol{y}_{1:n}, \Theta)\mathbf{P}(\boldsymbol{x}_{n-1}|\boldsymbol{x}_n, \boldsymbol{y}_{1:n-1}, \Theta)\cdots\mathbf{P}(x_0|\boldsymbol{x}_1, \Theta)$$

  ("forward-filtering, backward-sampling")

Forward-Filtering, Backward Sampling

$$\mathbf{P}(\boldsymbol{x}_{0:n}|\boldsymbol{y}_{1:n},\Theta) = \mathbf{P}(\boldsymbol{x}_n|\boldsymbol{y}_{1:n},\Theta)\mathbf{P}(\boldsymbol{x}_{n-1}|\boldsymbol{x}_n,\boldsymbol{y}_{1:n-1},\Theta)\cdots\mathbf{P}(x_0|\boldsymbol{x}_1,\Theta)$$

- Because $\boldsymbol{x}_{1:n}$ and $\boldsymbol{y}_{1:n}$ has a **joint MV normal** distribution,

$$\mathbf{P}(\boldsymbol{x}_t|\boldsymbol{x}_{t+1},\boldsymbol{y}_{1:t},\Theta) = \mathcal{N}(\boldsymbol{m}_t, V_t), \qquad t = n-1, n-2, \ldots, 0,$$

$$\boldsymbol{m}_t = \mathbb{E}\left[\boldsymbol{x}_t|\boldsymbol{x}_t,\boldsymbol{y}_{1:t-1},\Theta\right], \qquad V_t = \mathrm{Var}(\boldsymbol{x}_t|\boldsymbol{x}_{t+1},\boldsymbol{y}_{1:t},\Theta).$$

- In fact,

$$\boldsymbol{m}_t = \boldsymbol{x}_t^t + J_t(\boldsymbol{x}_t - \boldsymbol{x}_{t+1}^t), \qquad V_t = P_t^t - J_t P_{t+1}^t J_t'$$

  where $J_t = P_t^t \Phi [P_{t+1}^t]^{-1}$ (from the **Kalman Smoother** equations).
- **Algorithm** (for step (II)):
  1. Sample $\boldsymbol{x}_n \sim \mathcal{N}(\boldsymbol{x}_n^n, P_n^n)$.
  2. For $t = n-1, n-2, \ldots, 0$, sample $\boldsymbol{x}_t \sim \mathcal{N}(\boldsymbol{m}_t, V_t)$.

Example 6.26: Random Walk Hidden by Noise

- Model

$$y_t = x_t + v_t, \qquad x_t = x_{t-1} + w_t.$$

$w_t \overset{iid}{\sim} \mathcal{N}(0, \sigma_w^2)$, $v_t \overset{iid}{\sim} \mathcal{N}(0, \sigma_v^2)$.

- $\Theta = \left(\sigma_w^2, \sigma_v^2\right)$
- **Gibbs Sampling**: Iterate
  - (I) Draw $\Theta^{(j)} \sim \mathbf{P}(\Theta | x_{0:n}, y_{1:n})$
  - (II) Draw $x_{0:n}^{(j)} \sim \mathbf{P}(x_{0:n} | \Theta^{(j)}, y_{1:n})$.
- For (I), take $\pi(\Theta | x_{0:n}, y_{1:n})$ from a **conjugate family**:

$$\sigma_w^2 | x_{0:n}, y_{1:n} \sim \mathrm{IG}\left(\frac{a_0 + n}{2}, \frac{b_0 + \sum_{t=1}^n (x_t - x_{t-1})^2}{2}\right)$$

$$\sigma_v^2 | x_{0:n}, y_{1:n} \sim \mathrm{IG}\left(\frac{c_0 + n}{2}, \frac{d_0 + \sum_{t=1}^n (y_t - x_t)^2}{2}\right)$$

a_0, b_0, c_0 and d_0 represents initial believes on σ_w^2 and σ_v^2.

Example 6.24 (cont'd)

Sample $\boldsymbol{x}_n \sim \mathcal{N}(\boldsymbol{x}_n^n, P_n^n)$; $\boldsymbol{x}_t \sim \mathcal{N}(\boldsymbol{m}_t, V_t)$ for $t = n-1, n-2, \ldots, 0$:

```
ffbs = function(y,V,W,m0,C0){
  n  = length(y);  a  = rep(0,n);  R  = rep(0,n)
  m  = rep(0,n);   C  = rep(0,n);  B  = rep(0,n-1)
  H  = rep(0,n-1); mm = rep(0,n);  CC = rep(0,n)
  x  = rep(0,n); llike = 0.0
  for (t in 1:n){ # forward filtering
    if(t==1){a[1] = m0; R[1] = C0 + W
      }else{ a[t] = m[t-1]; R[t] = C[t-1] + W }
    f       = a[t]
    Q       = R[t] + V
    A       = R[t]/Q
    m[t]    = a[t]+A*(y[t]-f)
    C[t]    = R[t]-Q*A**2
    B[t-1]  = C[t-1]/R[t]
    H[t-1]  = C[t-1]-R[t]*B[t-1]**2
    llike   = llike + dnorm(y[t],f,sqrt(Q),log=TRUE) }
  mm[n] = m[n]; CC[n] = C[n]
  x[n]  = rnorm(1,m[n],sqrt(C[n]))
  for (t in (n-1):1){ # backward sampling
    mm[t] = m[t] + C[t]/R[t+1]*(mm[t+1]-a[t+1])
    CC[t] = C[t] - (C[t]^2)/(R[t+1]^2)*(R[t+1]-CC[t+1])
    x[t]  = rnorm(1,m[t]+B[t]*(x[t+1]-a[t+1]),sqrt(H[t]))  }
return(list(x=x,m=m,C=C,mm=mm,CC=CC,llike=llike))   }
```

Example 6.24 (cont'd)

```
# Simulate states and data
set.seed(1); W = 0.5; V = 1.0
n  = 100; m0 = 0.0; C0 = 10.0; x0 = 0
w  = rnorm(n,0,sqrt(W))
v  = rnorm(n,0,sqrt(V))
x  = y = rep(0,n)
x[1] = x0   + w[1]
y[1] = x[1] + v[1]
for (t in 2:n){
  x[t] = x[t-1] + w[t]
  y[t] = x[t] + v[t]   }
#
run = ffbs(y,V,W,m0,C0)
m   = run$m; C = run$C; mm = run$mm
CC  = run$CC; L1 = m-2*C; U1  = m+2*C
L2  = mm-2*CC; U2 = mm+2*CC
N   = 50
Vs  = seq(0.1,2,length=N)
Ws  = seq(0.1,2,length=N)
likes = matrix(0,N,N)
for (i in 1:N){
 for (j in 1:N){
   V   = Vs[i]
   W   = Ws[j]
   run = ffbs(y,V,W,m0,C0)
  likes[i,j] = run$llike  }  }
```

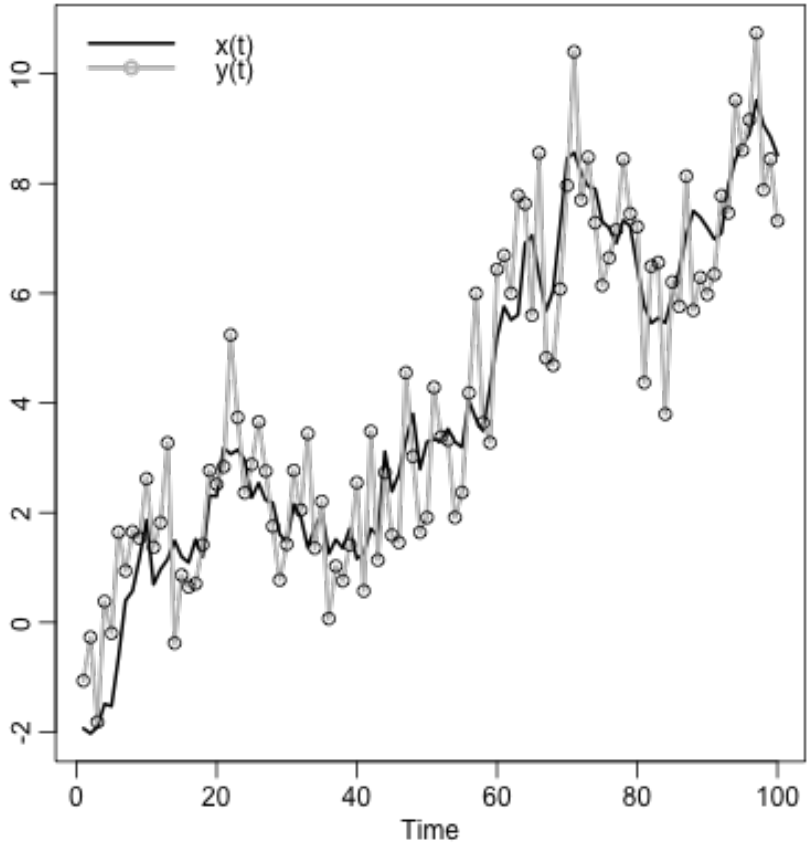
Example 6.24 (cont'd)

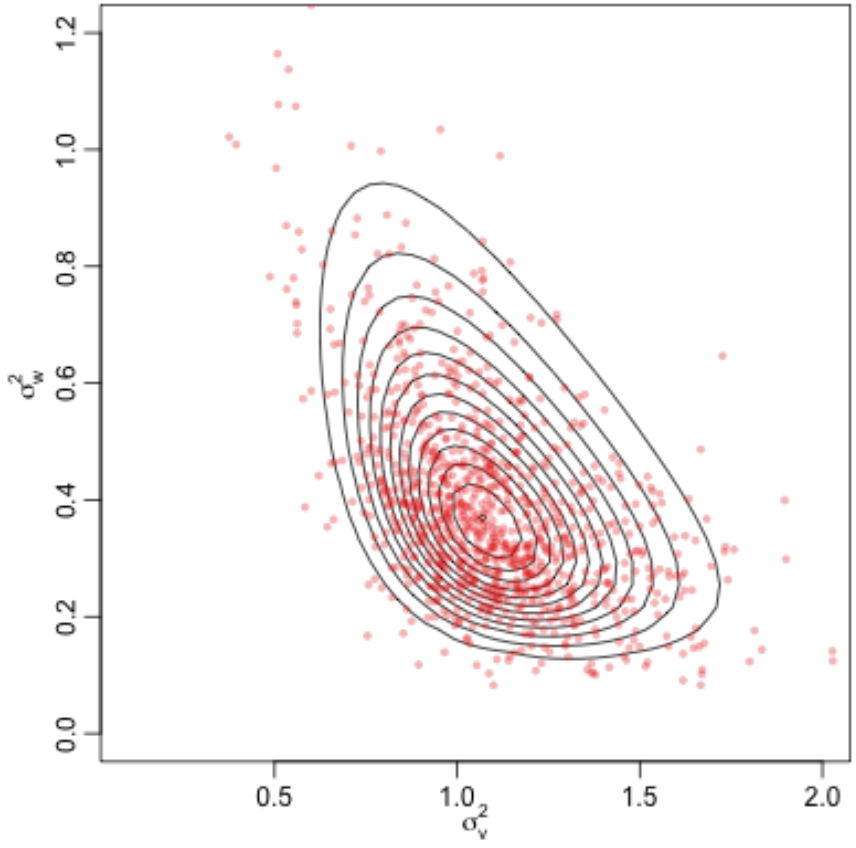
```
# Hyperparameters
a = 0.01; b = 0.01; c = 0.01; d = 0.01
# MCMC step
set.seed(90210)
burn  = 10;  M = 1000
niter = burn + M
V1    = V;  W1 = W
draws = NULL
all_draws = NULL
for (iter in 1:niter){
  run   = ffbs(y,V1,W1,m0,C0) # draw x[0:n]
  x     = run$x
  # draw Theta=(V1,W1)
  V1    = 1/rgamma(1,a+n/2,b+sum((y-x)^2)/2)
  W1    = 1/rgamma(1,c+(n-1)/2,d+sum(diff(x)^2)/2)
  draws = rbind(draws,c(V1,W1,x))    }
all_draws = draws[,1:2]
q025  = function(x){quantile(x,0.025)}
q975  = function(x){quantile(x,0.975)}
draws = draws[(burn+1):(niter),]
xs    = draws[,3:(n+2)]
lx    = apply(xs,2,q025)
mx    = apply(xs,2,mean)
ux    = apply(xs,2,q975)
```

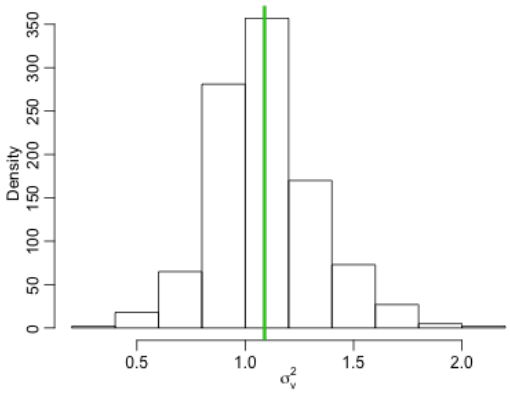
Example 6.24 (cont'd)

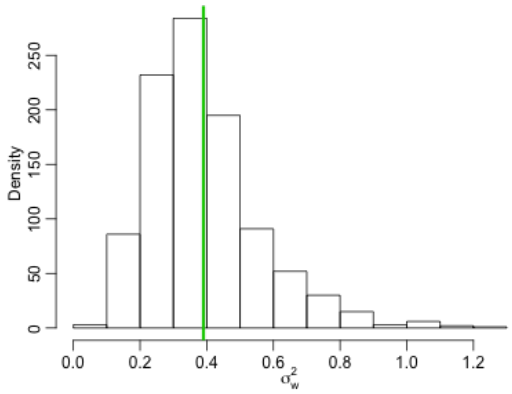
```
q025  = function(x){quantile(x,0.025)}
q975  = function(x){quantile(x,0.975)}
draws = draws[(burn+1):(niter),]
xs    = draws[,3:(n+2)]
lx    = apply(xs,2,q025)
mx    = apply(xs,2,mean)
ux    = apply(xs,2,q975)

##  plot of the data
par(mfrow=c(1,2), mgp=c(1.6,.6,0), mar=c(3,3.2,1,1))
ts.plot(ts(x), ts(y), ylab='', col=c(1,8), lwd=2)
points(y)
legend(0, 11, legend=c("x(t)","y(t)"), lty=1, col=c(1,8), lwd=2, bty="n", p
contour(Vs, Ws, exp(likes), xlab=expression(sigma[v]^2),
ylab=expression(sigma[w]^2), drawlabels=FALSE, ylim=c(0,1.2))
points(draws[,1:2], pch=16, col=rgb(.9,0,0,0.3), cex=.7)

# Histograms:
par(mfrow=c(1,2), mgp=c(1.6,.6,0), mar=c(3,3.2,1,1))
hist(draws[,1], ylab="Density",main="", xlab=expression(sigma[v]^2))
abline(v=mean(draws[,1]), col=3, lwd=3)
hist(draws[,2],main="", ylab="Density", xlab=expression(sigma[w]^2))
abline(v=mean(draws[,2]), col=3, lwd=3)
```

Example 6.24 (cont'd)









Example 6.24 (cont'd)