FITTING CONDITIONAL HETEROSCEDASTIC MODELS

In a *conditional heteroscedastic model*, we have an observed data set with *time-varying variance*. Typically, these models can be considered as special forms of state space models.

Stochastic Volatility: The *stochastic volatility* model uses a stochastic state space form that allows the variance of the observed process to vary stochastically with time. The simplest form specifies that

$$Y_t = \exp\{X_t/2\}W_t \qquad \{W_t\} \sim \text{WN}(0, \sigma_W^2)$$

and then that

$$X_{t+1} = \phi X_t + V_t \qquad \{V_t\} \sim \text{WN}(0, \sigma_V^2)$$

that is, given $X_t = x_t$, Y_t has mean zero and variance $\exp\{x_t\}$, and $\{X_t\} \sim \text{AR}(1)$. We can reparameterize the model as follows:

$$Y_t = \exp\{X_t/2\}Z_t \qquad \{Z_t\} \sim \text{WN}(0, 1)$$

and

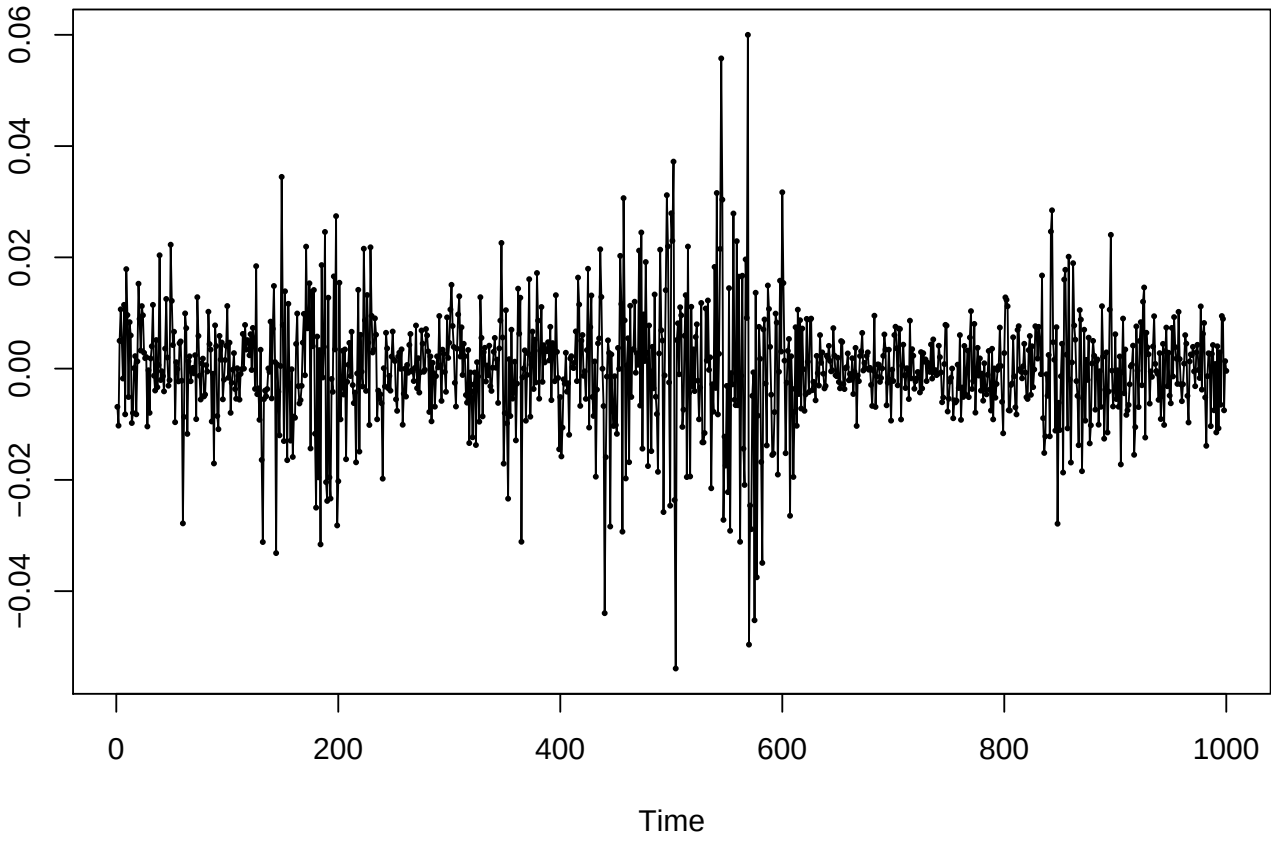
$$X_{t+1} = \mu + \phi(X_t - \mu) + V_t \qquad \{V_t\} \sim \text{WN}(0, \sigma_V^2).$$

where $\mu = 2\log\sigma_W$.

```
set.seed(24)
n<-1000;phi<-0.98;sigW<-0.01;sigV<-0.05
X<-arima.sim(n,model=list(ar=phi),rand.gen = function(n) sqrt(sigV) * rnorm(n))
Z<-rnorm(n)*sigW
Y<-exp(X/2)*Z
var(Y)

+ [1] 0.000108337

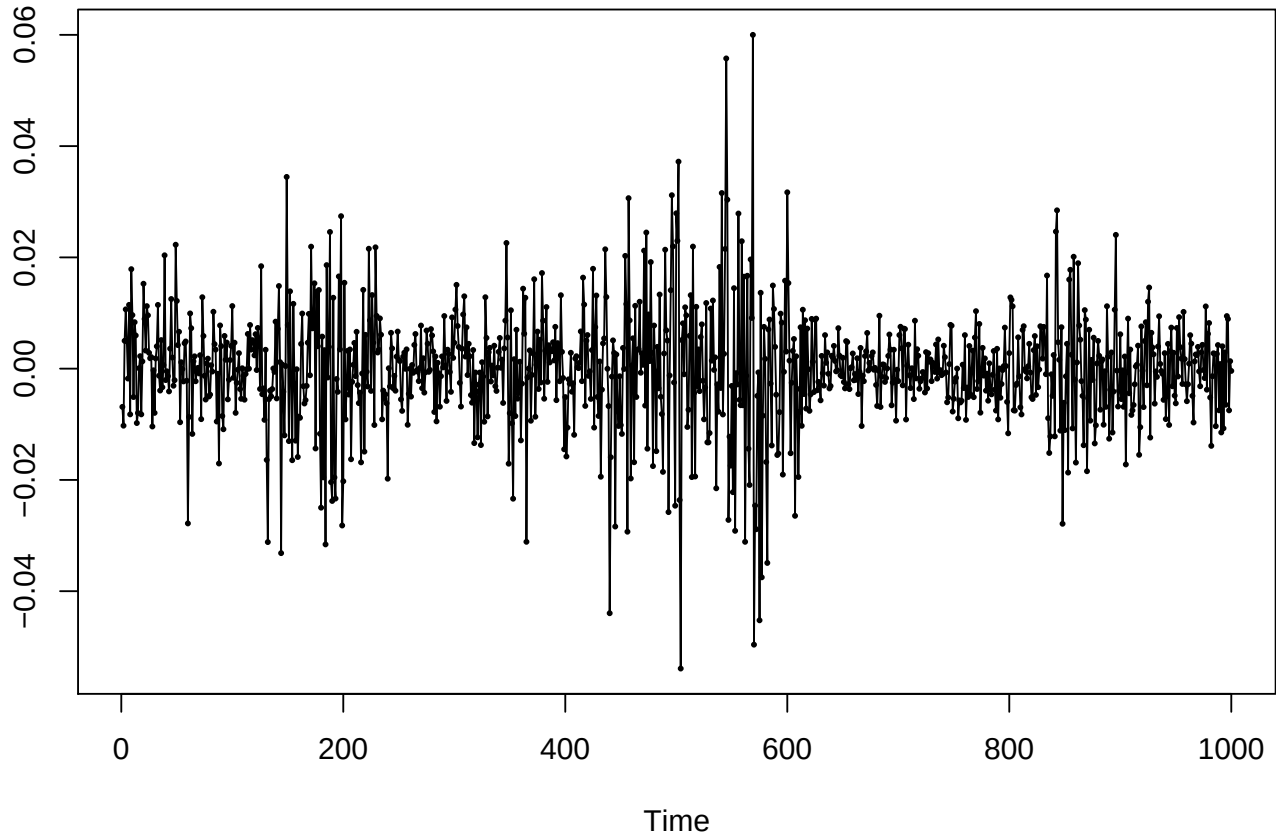
par(mar=c(4,3,2,0));plot(Y,type='l',ylim=range(Y));
points(Y,pch=19,cex=0.3)
```



```r
set.seed(24)
mu<-2*log(sigW)  #Reparameterization
X<-arima.sim(n,model=list(ar=phi),rand.gen = function(n) sqrt(sigV) * rnorm(n))
X<-X+mu
Z<-rnorm(n)
Y<-exp(X/2)*Z
var(Y)

+ [1] 0.000108337

par(mar=c(4,3,2,0));plot(Y,type='l',ylim=range(Y));
points(Y,pch=19,cex=0.3)
```



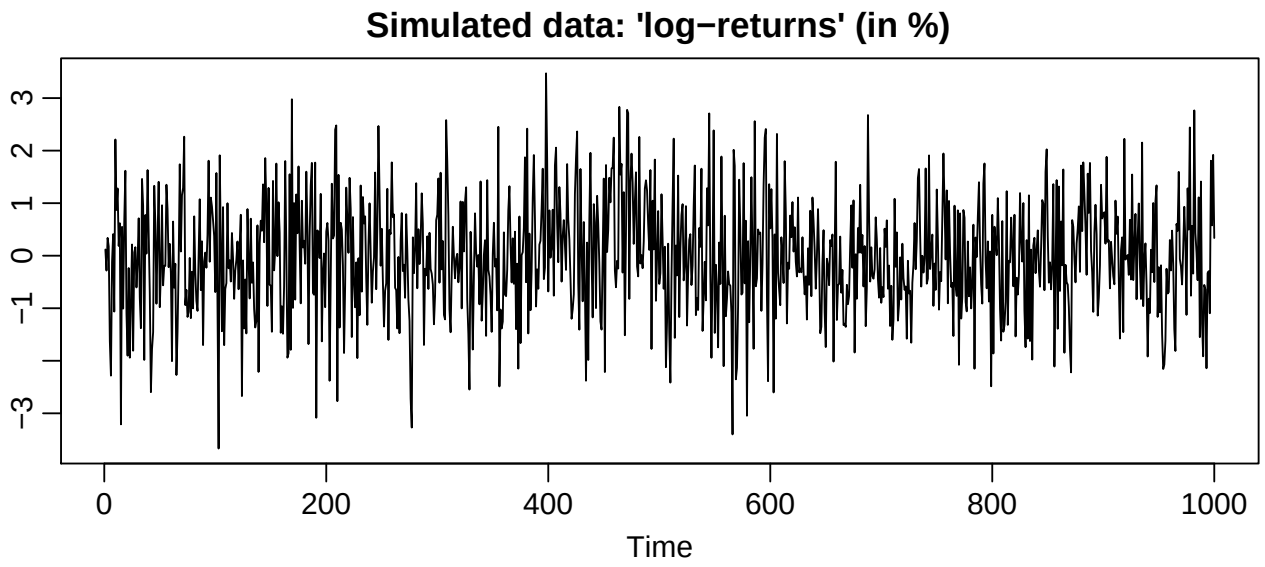
Inference for the SV model can be carried out using Bayesian methods and Markov chain Monte Carlo (MCMC) and the `stochvol` library.

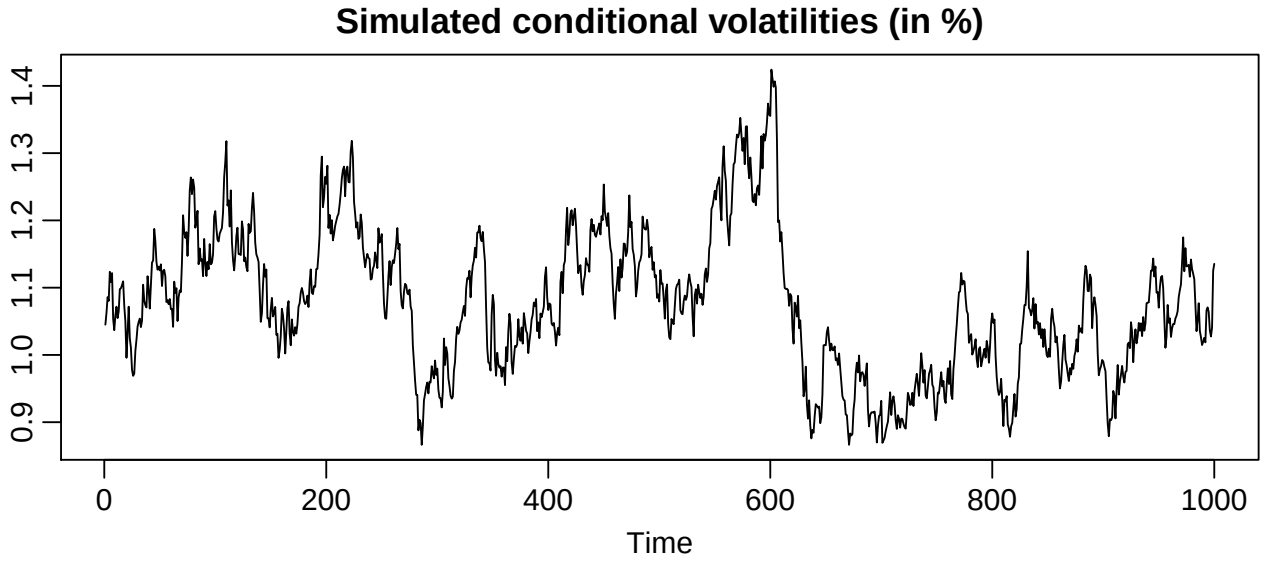
- `svsim` simulates a SV model
- `svsample` carries out MCMC analysis

```r
library(stochvol)
y.sim <- svsim(1000, mu = 2*log(sigW), phi = phi, sigma = sigV)
names(y.sim)

+ [1] "y"    "vol"  "vol0" "para"

plot(y.sim)
```



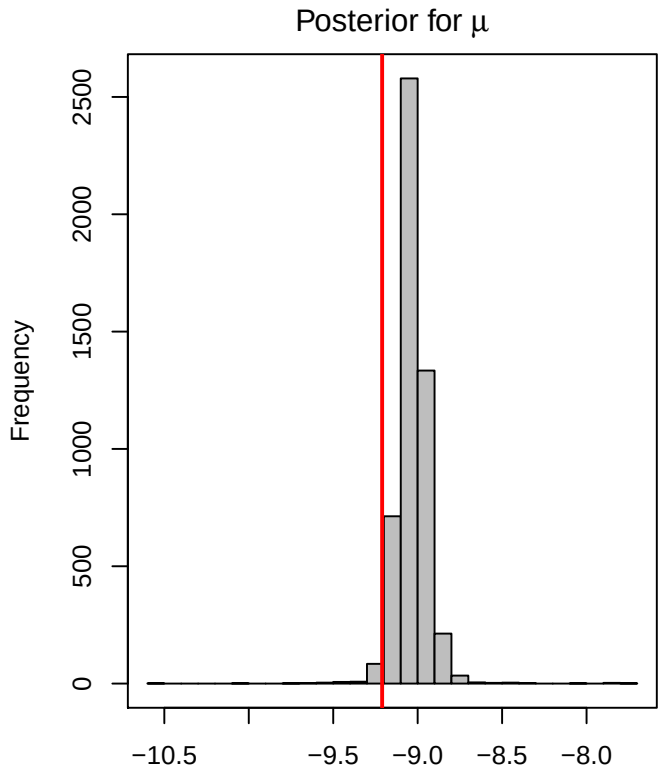


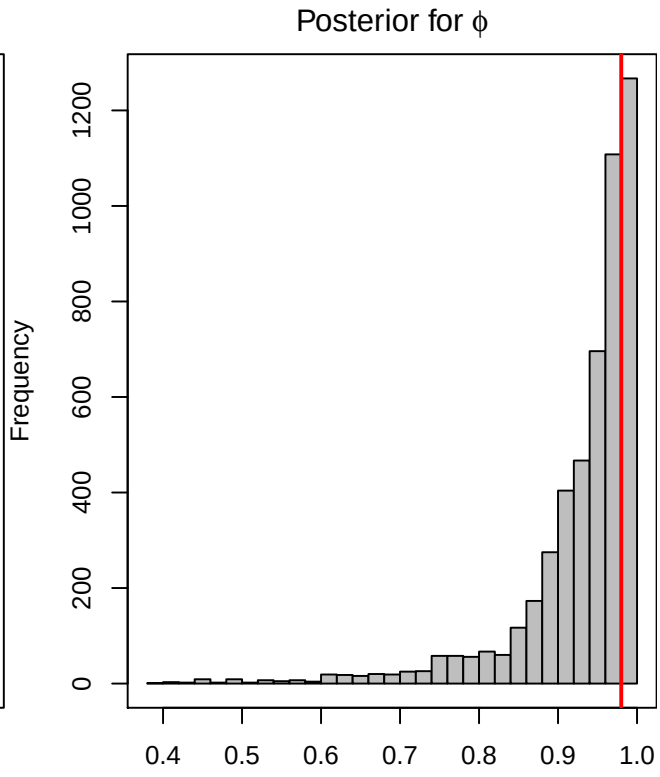
The algorithm draws 5000 samples from the *posterior distribution* for the three parameters of the model μ, ϕ and σ_V.

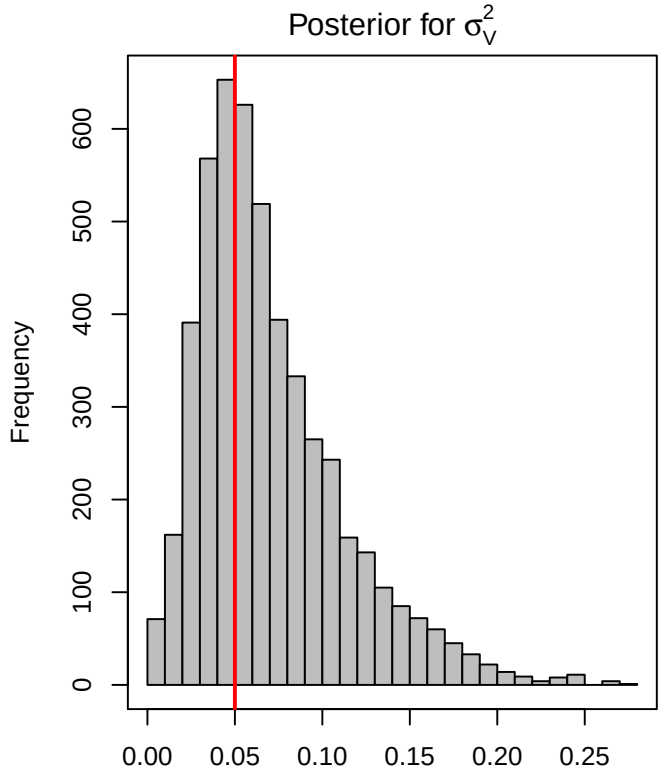
```
mcmc.draws <- svsample(y.sim$y, draws = 5000, burnin = 1000, priormu = c(0, 10),
                       priorphi = c(20, 1.2), priorsigma = 100)
names(mcmc.draws)

+ [1] "para"     "latent"   "latent0"  "tau"      "y"        "priors"
+ [7] "runtime"  "thinning" "summary"

par(mfrow=c(2,2),mar=c(4,4,2,0))
hist(mcmc.draws$para[,1],col='gray',nclass=30,xlab='',
          main=expression(paste('Posterior for ',mu)))
box();abline(v=mu,col='red',lwd=2)   #Mark the true values with red lines.
hist(mcmc.draws$para[,2],col='gray',nclass=30,xlab='',
          main=expression(paste('Posterior for ',phi)))
box();abline(v=phi,col='red',lwd=2)
hist(mcmc.draws$para[,3],col='gray',nclass=30,xlab='',
          main=expression(paste('Posterior for ',sigma[V]^2)))
box();abline(v=sigV,col='red',lwd=2)
```







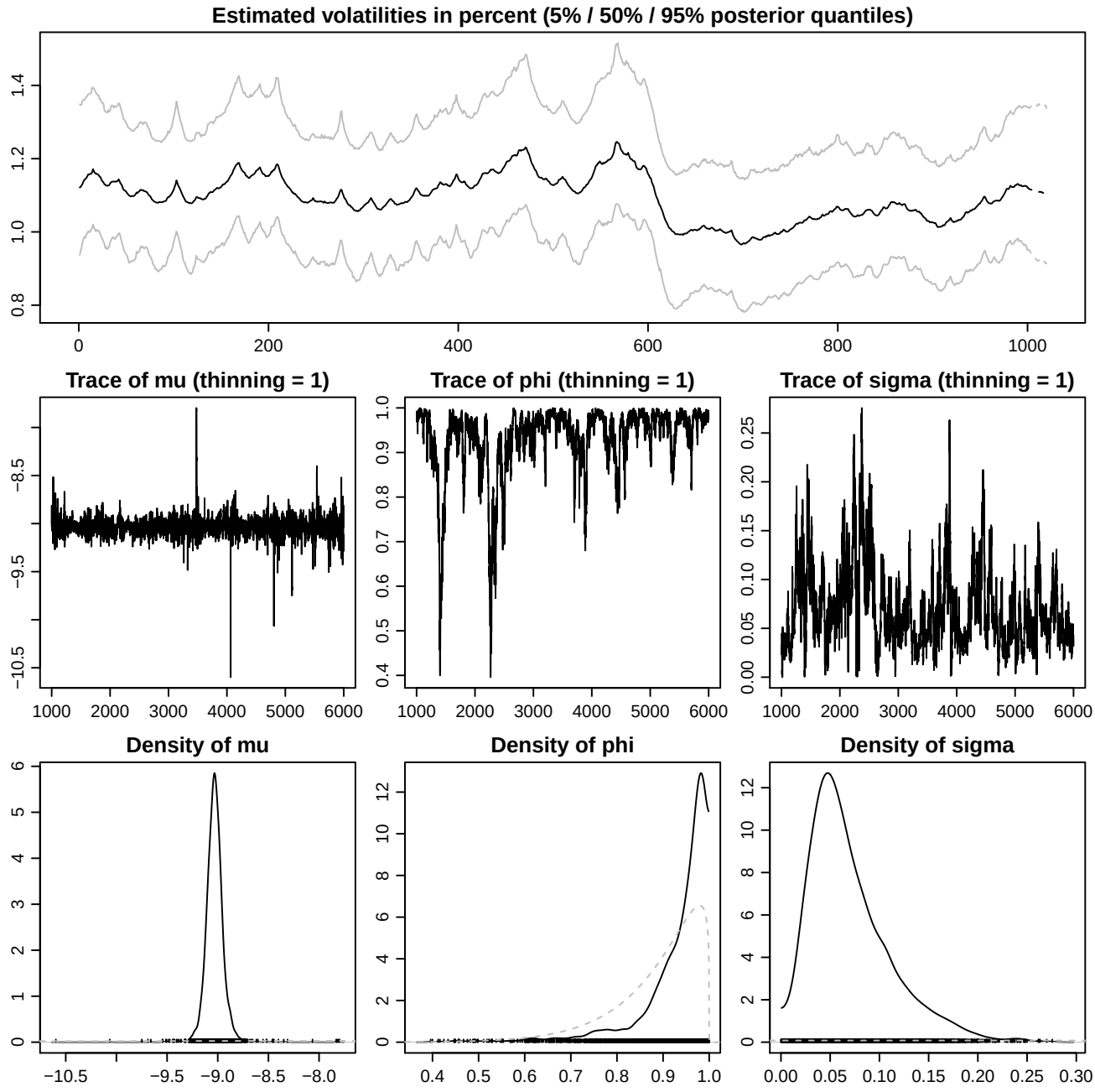
Parameter estimates can be obtained from the posterior distribution by summarizing the posterior samples:

```
apply(mcmc.draws$para,2,summary)

+                  mu       phi          sigma
+ Min.     -10.597414 0.3960318 0.000006423151
+ 1st Qu.   -9.078864 0.9098587 0.040904832253
+ Median    -9.031742 0.9568723 0.060747455084
+ Mean      -9.029947 0.9291611 0.070414505390
+ 3rd Qu.   -8.984730 0.9801732 0.090874061574
+ Max.      -7.797500 0.9997338 0.275599274894
```

We can also forecast using the model, with commands

```
fore <- predict(mcmc.draws, 20)
plot(mcmc.draws, forecast = fore)
```



ARCH/GARCH MODEL: The *ARCH* and *GARCH* models use *deterministic* state space forms that allows the variance of the observed process to vary with time. The ARCH(1) model specifies

$$Y_t = \sqrt{X_t} Z_t \qquad \{Z_t\} \sim \text{WN}(0,1)$$

and then that

$$X_{t+1} = \alpha_0 + \alpha_1 Y_t^2$$

whereas the GARCH(1,1) model specifies

$$X_{t+1} = \alpha_0 + \alpha_1 Y_t^2 + \beta_1 X_t$$

where, in both cases, given $X_t = x_t$, Y_t has mean zero and variance $\sqrt{x_t}$. The most general form of the GARCH model is the GARCH(p,q) model where

$$X_{t+1} = \alpha_0 + \sum_{j=1}^{p} \alpha_j Y_{t-j}^2 + \sum_{j=1}^{q} \beta_j X_{t-j}$$

```
set.seed(345)
n<-1000;al0<-0.01;al1<-0.95
X<-Y<-rep(0,n);X[1]<-al0;Y[1]<-rnorm(1)*sqrt(X[1])  #ARCH(1) MODEL
for(j in 2:n){
    X[j]<-al0+al1*Y[j-1]^2
    Y[j]<-rnorm(1)*sqrt(X[j])
}
```

GARCH models can be fitted using the `fGarch` library.

```
library(fGarch)
fit.10<-garchFit(Y~garch(1,0),trace=F)  #ARCH(1)
summary(fit.10)
```

```
+
+ Title:
+  GARCH Modelling
+
+ Call:
+  garchFit(formula = Y ~ garch(1, 0), trace = F)
+
+ Mean and Variance Equation:
+  data ~ garch(1, 0)
+ <environment: 0x0000000013cf8c10>
+  [data = fGarch::dem2gbp]
+
+ Conditional Distribution:
+  norm
+
+ Coefficient(s):
+         mu       omega      alpha1
+ -0.0015506   0.1465275   0.3708670
+
+ Std. Errors:
+  based on Hessian
+
+ Error Analysis:
+         Estimate  Std. Error  t value            Pr(>|t|)
+ mu     -0.001551    0.009362   -0.166               0.868
+ omega   0.146527    0.006397   22.905 <0.0000000000000002 ***
+ alpha1  0.370867    0.043667    8.493 <0.0000000000000002 ***
+ ---
```

```
+ Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
+
+ Log Likelihood:
+  -1206.588    normalized:  -0.61124
+
+ Description:
+  Tue Dec 05 14:14:32 2017 by user: dsteph5
+
+
+ Standardised Residuals Tests:
+                                 Statistic p-Value
+  Jarque-Bera Test   R    Chi^2  672.3254  0
+  Shapiro-Wilk Test  R    W      0.9629643 0
+  Ljung-Box Test     R    Q(10)  9.398438  0.4947532
+  Ljung-Box Test     R    Q(15)  15.97803  0.3835132
+  Ljung-Box Test     R    Q(20)  20.70954  0.4144026
+  Ljung-Box Test     R^2  Q(10)  145.6556  0
+  Ljung-Box Test     R^2  Q(15)  182.0525  0
+  Ljung-Box Test     R^2  Q(20)  207.4677  0
+  LM Arch Test       R    TR^2   107.0691  0
+
+ Information Criterion Statistics:
+      AIC      BIC      SIC     HQIC
+ 1.225519 1.234012 1.225515 1.228640
```

```
fit.11<-garchFit(Y~garch(1,1),trace=F)  #GARCH(1,1)
summary(fit.11)
```

```
+
+ Title:
+  GARCH Modelling
+
+ Call:
+  garchFit(formula = Y ~ garch(1, 1), trace = F)
+
+ Mean and Variance Equation:
+  data ~ garch(1, 1)
+ <environment: 0x0000000013efe6b8>
+  [data = fGarch::dem2gbp]
+
+ Conditional Distribution:
+  norm
+
+ Coefficient(s):
+         mu       omega      alpha1       beta1
+ -0.0061903   0.0107614   0.1531341   0.8059737
+
+ Std. Errors:
+  based on Hessian
+
+ Error Analysis:
+         Estimate  Std. Error  t value             Pr(>|t|)
+ mu     -0.006190    0.008462   -0.732             0.464448
+ omega   0.010761    0.002838    3.793             0.000149 ***
+ alpha1  0.153134    0.026422    5.796         0.0000000068 ***
+ beta1   0.805974    0.033381   24.144 < 0.0000000000000002 ***
+ ---
+ Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
+
+ Log Likelihood:
+  -1106.608    normalized:  -0.5605916
+
```

```
+ Description:
+  Tue Dec 05 14:14:32 2017 by user: dsteph5
+
+
+ Standardised Residuals Tests:
+                                 Statistic p-Value
+  Jarque-Bera Test   R    Chi^2  1059.851  0
+  Shapiro-Wilk Test  R    W      0.9622848 0
+  Ljung-Box Test     R    Q(10)  10.12141  0.4299066
+  Ljung-Box Test     R    Q(15)  17.04349  0.3162711
+  Ljung-Box Test     R    Q(20)  19.29764  0.5025619
+  Ljung-Box Test     R^2  Q(10)  9.062556  0.5261773
+  Ljung-Box Test     R^2  Q(15)  16.07769  0.3769072
+  Ljung-Box Test     R^2  Q(20)  17.50715  0.6198388
+  LM Arch Test       R    TR^2   9.771217  0.6360237
+
+ Information Criterion Statistics:
+      AIC      BIC      SIC     HQIC
+ 1.125236 1.136559 1.125228 1.129396

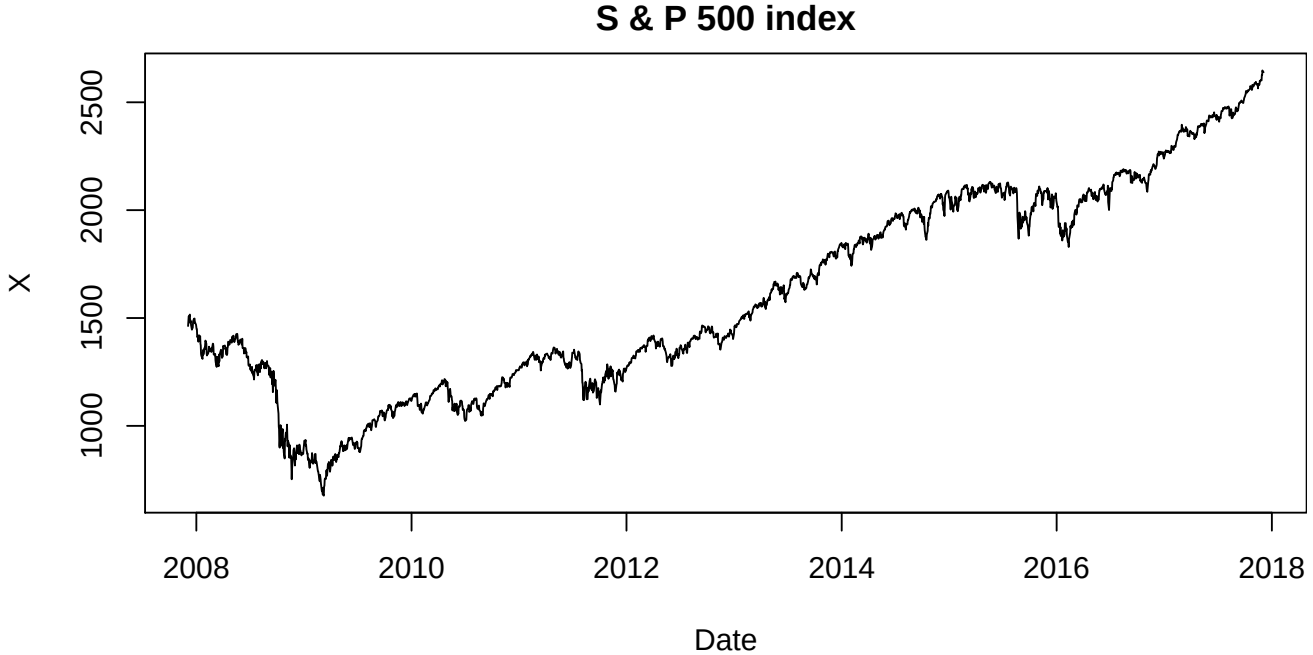
c(fit.10@fit$ics[1],fit.11@fit$ics[1])  #AIC comparison

+      AIC      AIC
+ 1.225519 1.125236
```

The ARCH(1) model is preferred. The parameter referred to as `omega` is α_0, and `mu` is the mean of the $\{Y_t\}$ series.

For an analysis of a real data set, we consider the S & P 500 index over the period 2007-2017.

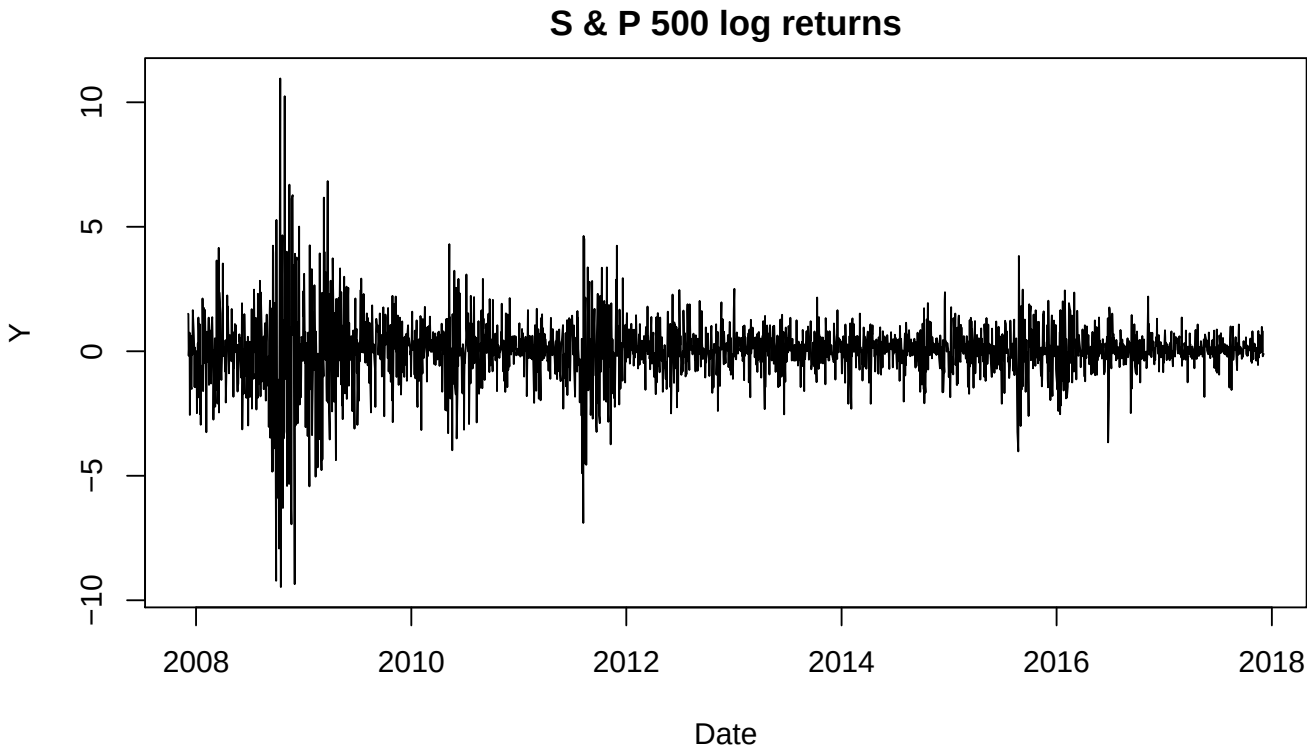
```
SP.2017<-read.csv('GSPC.csv',header=TRUE)
X<-SP.2017$Close
Tv<-as.Date(as.character(SP.2017$Date),'%d/%m/%Y')
par(mar=c(4,4,2,0))
plot(Tv,X,type='l',main='S & P 500 index',xlab='Date')
```



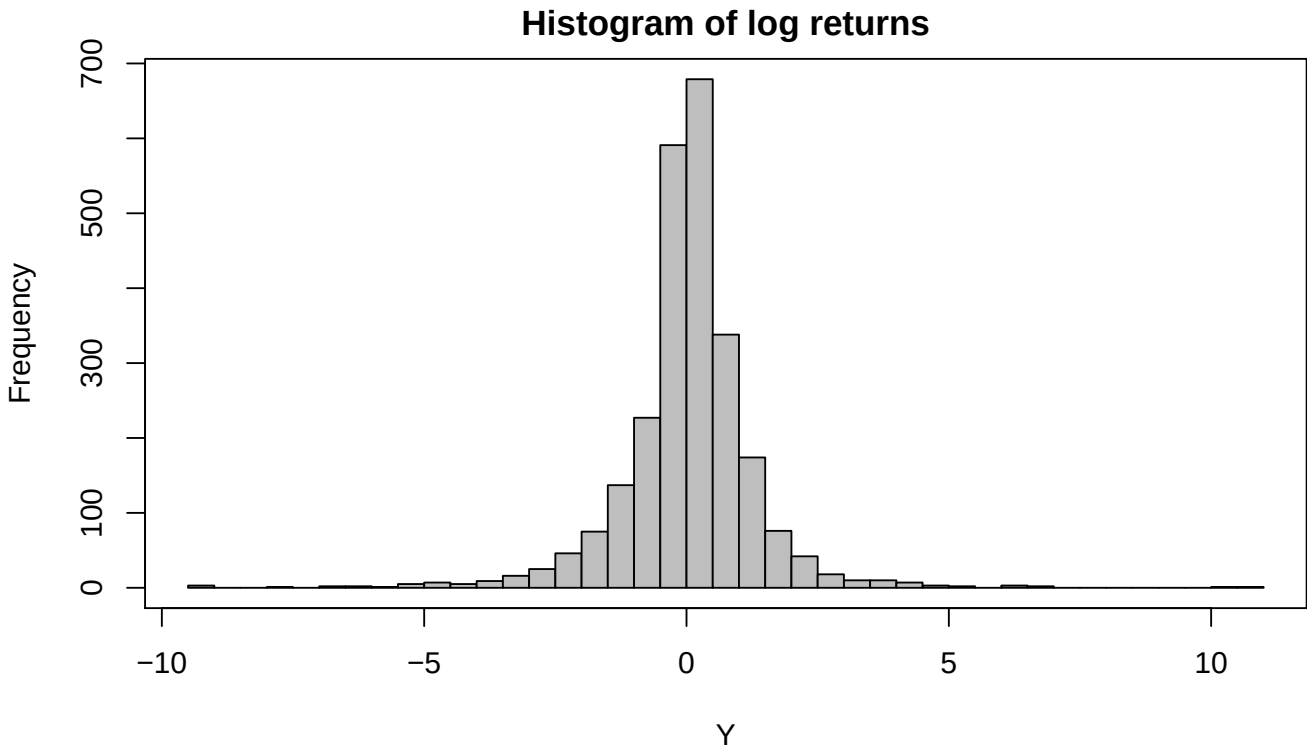
If we let X_t denote the index value, we then transform to the percentage log returns series

$$Y_t = \log(X_{t+1}/X_t) = 100(\log X_{t+1} - \log X_t)$$

```
Y<-100*diff(log(X))
par(mar=c(4,4,2,0))
plot(Tv[-1],Y,type='l',main='S & P 500 log returns',xlab='Date')
```

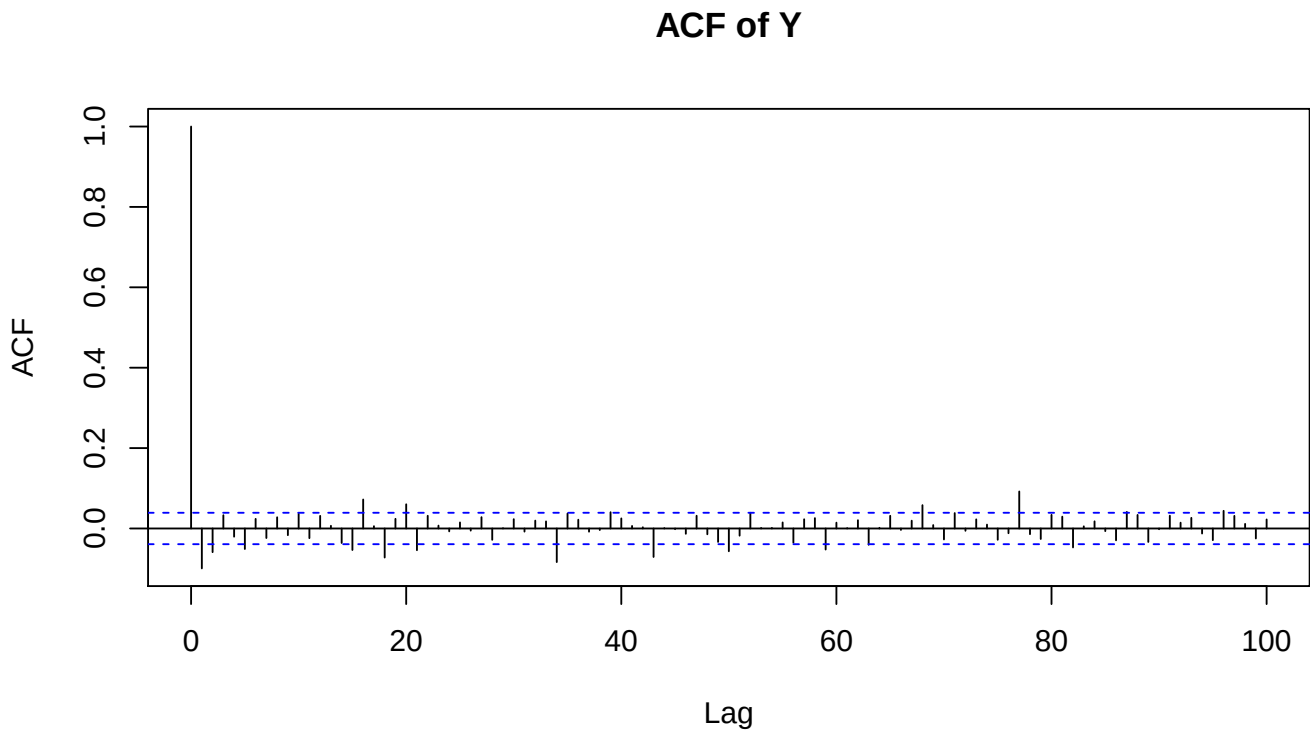



```
hist(Y,nclass=50,col='gray',main='Histogram of log returns');box()
```

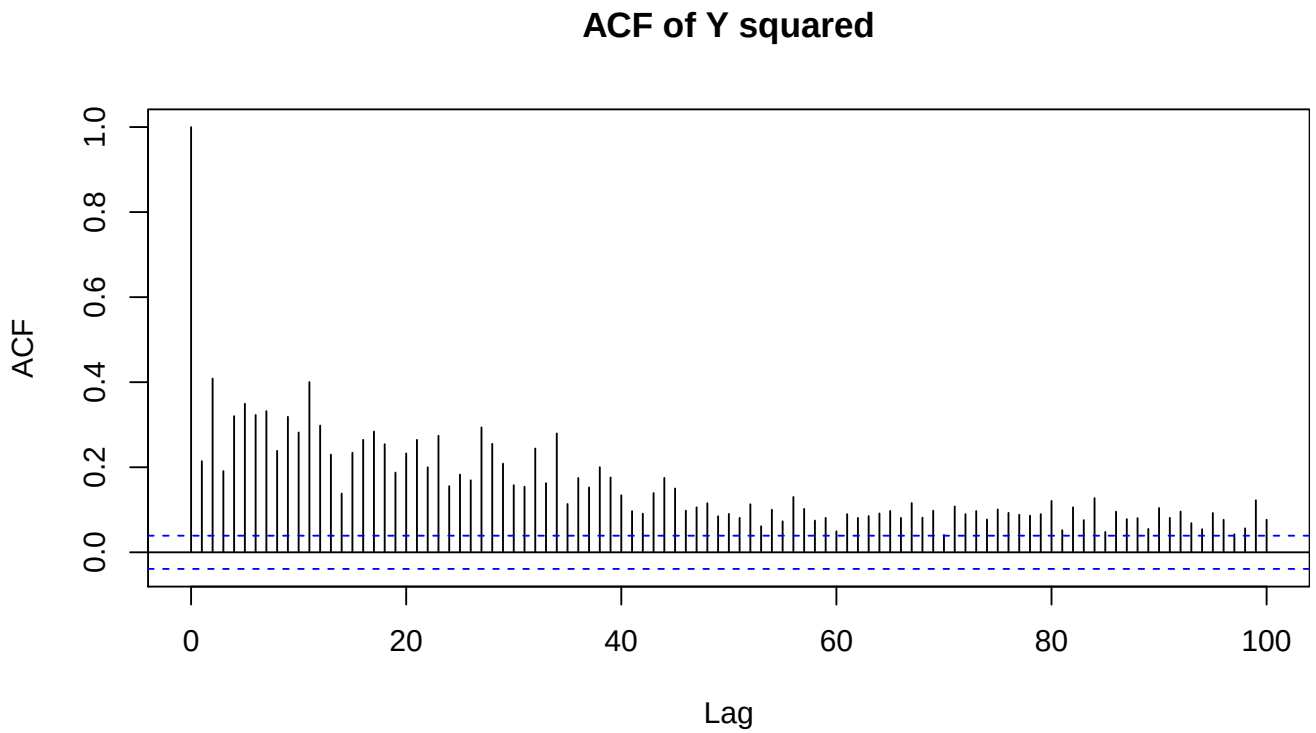


The histogram indicates a *leptokurtic* (or heavy-tailed) marginal distribution for the log returns.

```r
par(mar=c(4,4,4,0))
acf(Y,lag.max=100,main='ACF of Y')
```




```r
acf(Y^2,lag.max=100,main='ACF of Y squared')
```



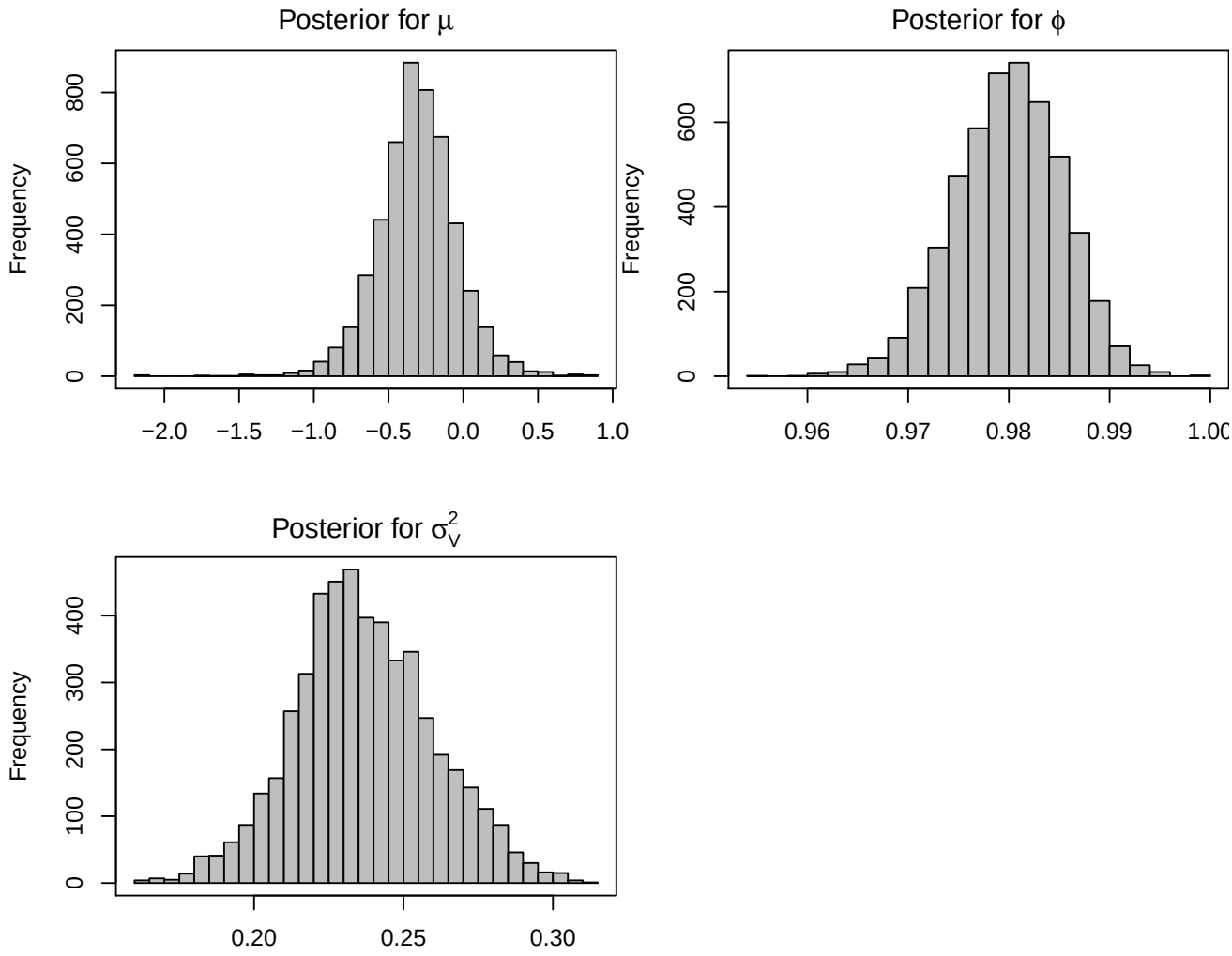
The log returns series is uncorrelated, buy the squared series has significant autocorrelation.

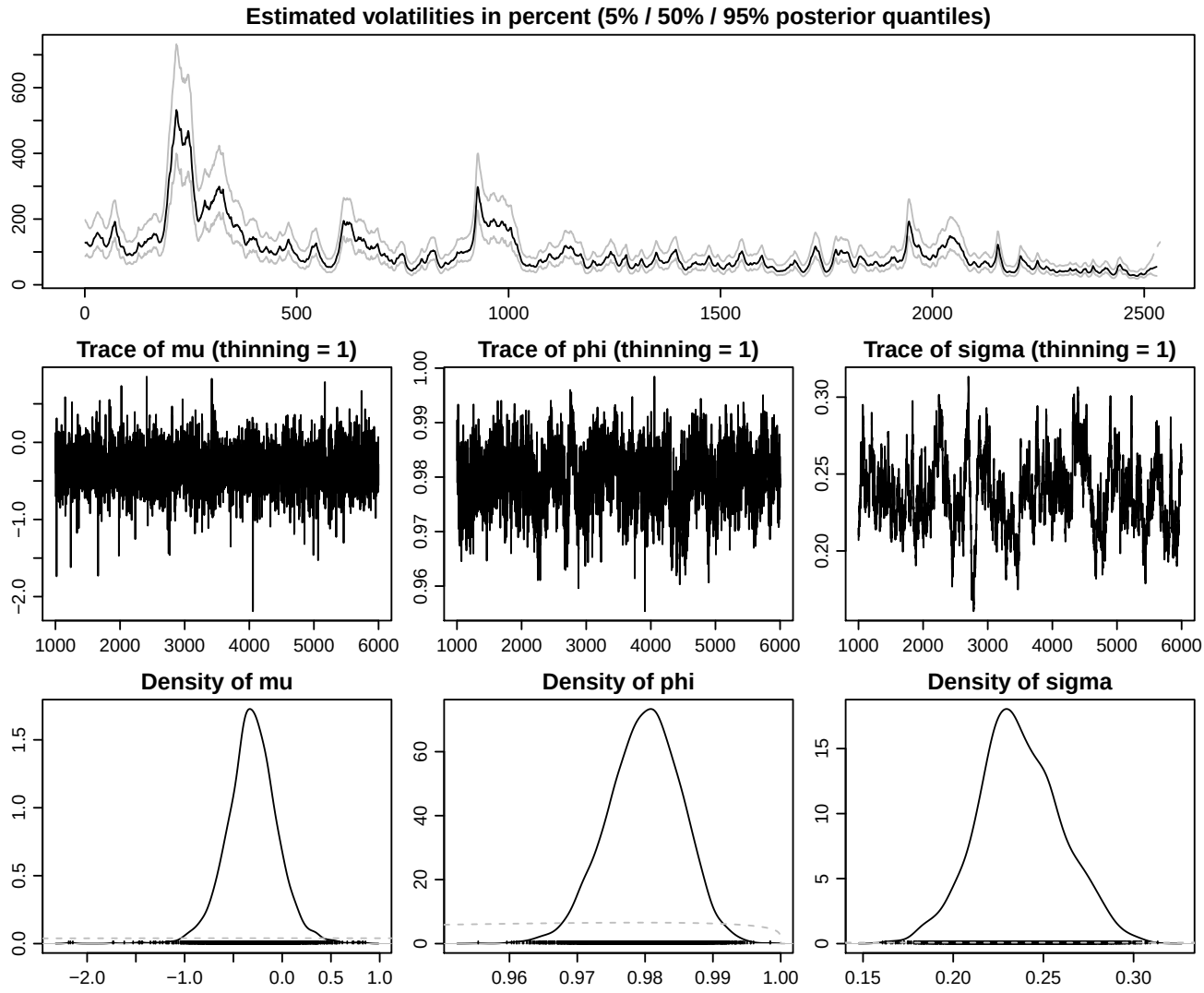
We can carry out an analysis using the stochastic volatility model.

```
mcmc.draws <- svsample(Y, draws = 5000, burnin = 1000, priormu = c(0, 10),
                       priorphi = c(20, 1.2), priorsigma = 100)
```

```
+ Warning in svsample(Y, draws = 5000, burnin = 1000, priormu = c(0, 10), :  Argument 'y' (data
vector) contains zeros.  I am adding an offset constant of size 0.000129092224403547 to do the
auxiliary mixture sampling.  If you want to avoid this, you might consider de-meaning the returns
before calling this function.
```

```
par(mfrow=c(2,2),mar=c(4,4,2,0))
hist(mcmc.draws$para[,1],col='gray',nclass=30,xlab='',
          main=expression(paste('Posterior for ',mu)));box()
hist(mcmc.draws$para[,2],col='gray',nclass=30,xlab='',
          main=expression(paste('Posterior for ',phi)));box()
hist(mcmc.draws$para[,3],col='gray',nclass=30,xlab='',
          main=expression(paste('Posterior for ',sigma[V]^2)));box()
fore <- predict(mcmc.draws, 20)
plot(mcmc.draws, forecast = fore)
```





The ARCH(1) and the GARCH(1,1) models can also be used.

```
fit.10<-garchFit(Y~garch(1,0),trace=F)   #ARCH(1)
summary(fit.10)

+
+ Title:
+  GARCH Modelling
+
+ Call:
+  garchFit(formula = Y ~ garch(1, 0), trace = F)
+
+ Mean and Variance Equation:
+  data ~ garch(1, 0)
+ <environment: 0x00000000131fd710>
+  [data = fGarch::dem2gbp]
+
+ Conditional Distribution:
+  norm
+
+ Coefficient(s):
+         mu       omega      alpha1
+ -0.0015506   0.1465275   0.3708670
```

```
+
+ Std. Errors:
+  based on Hessian
+
+ Error Analysis:
+         Estimate  Std. Error  t value            Pr(>|t|)
+ mu     -0.001551    0.009362   -0.166               0.868
+ omega   0.146527    0.006397   22.905 <0.0000000000000002 ***
+ alpha1  0.370867    0.043667    8.493 <0.0000000000000002 ***
+ ---
+ Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
+
+ Log Likelihood:
+  -1206.588    normalized:  -0.61124
+
+ Description:
+  Tue Dec 05 14:14:33 2017 by user: dsteph5
+
+
+ Standardised Residuals Tests:
+                                 Statistic p-Value
+  Jarque-Bera Test   R    Chi^2  672.3254  0
+  Shapiro-Wilk Test  R    W      0.9629643 0
+  Ljung-Box Test     R    Q(10)  9.398438  0.4947532
+  Ljung-Box Test     R    Q(15)  15.97803  0.3835132
+  Ljung-Box Test     R    Q(20)  20.70954  0.4144026
+  Ljung-Box Test     R^2  Q(10)  145.6556  0
+  Ljung-Box Test     R^2  Q(15)  182.0525  0
+  Ljung-Box Test     R^2  Q(20)  207.4677  0
+  LM Arch Test       R    TR^2   107.0691  0
+
+ Information Criterion Statistics:
+      AIC      BIC      SIC     HQIC
+ 1.225519 1.234012 1.225515 1.228640
```

```
fit.11<-garchFit(Y~garch(1,1),trace=F)  #GARCH(1,1)
summary(fit.11)
```

```
+
+ Title:
+  GARCH Modelling
+
+ Call:
+  garchFit(formula = Y ~ garch(1, 1), trace = F)
+
+ Mean and Variance Equation:
+  data ~ garch(1, 1)
+ <environment: 0x00000000141c1a58>
+  [data = fGarch::dem2gbp]
+
+ Conditional Distribution:
+  norm
+
+ Coefficient(s):
+         mu       omega      alpha1       beta1
+ -0.0061903   0.0107614   0.1531341   0.8059737
+
+ Std. Errors:
+  based on Hessian
+
+ Error Analysis:
+         Estimate  Std. Error  t value            Pr(>|t|)
```

```
+ mu     -0.006190    0.008462   -0.732             0.464448
+ omega   0.010761    0.002838    3.793             0.000149 ***
+ alpha1  0.153134    0.026422    5.796         0.0000000068 ***
+ beta1   0.805974    0.033381   24.144 < 0.0000000000000002 ***
+ ---
+ Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
+
+ Log Likelihood:
+  -1106.608    normalized:  -0.5605916
+
+ Description:
+  Tue Dec 05 14:14:33 2017 by user: dsteph5
+
+
+ Standardised Residuals Tests:
+                                 Statistic p-Value
+  Jarque-Bera Test   R    Chi^2  1059.851  0
+  Shapiro-Wilk Test  R    W      0.9622848 0
+  Ljung-Box Test     R    Q(10)  10.12141  0.4299066
+  Ljung-Box Test     R    Q(15)  17.04349  0.3162711
+  Ljung-Box Test     R    Q(20)  19.29764  0.5025619
+  Ljung-Box Test     R^2  Q(10)  9.062556  0.5261773
+  Ljung-Box Test     R^2  Q(15)  16.07769  0.3769072
+  Ljung-Box Test     R^2  Q(20)  17.50715  0.6198388
+  LM Arch Test       R    TR^2   9.771217  0.6360237
+
+ Information Criterion Statistics:
+      AIC      BIC      SIC     HQIC
+ 1.125236 1.136559 1.125228 1.129396

c(fit.10@fit$ics[1],fit.11@fit$ics[1])  #AIC comparison

+      AIC      AIC
+ 1.225519 1.125236
```

Here the GARCH(1,1) model is preferred.