AMS 223 Time Series Analysis
Homework 1
Oct 10 2013
Cheng-Han Yu

Textbook: Prado, R. and M. West (2010) *Time Series - Modeling, Computation and Inference*. New York: Chapman & Hall/CRC.

1 Homework Problems

1. **Chapter 1 Problem 2** Consider the AR(1) model $y_t = \phi y_{t-1} + \epsilon_t$, with $\epsilon_t \sim N(0, v)$.

   (a) Find the MLE of (ϕ, v) for the conditional likelihood.
   Solution:
   We know $p(y_1|\boldsymbol{\theta}) = N(0, v/(1-\phi)^2)$ and $p(y_t|y_{t-1}, \boldsymbol{\theta}) = N(y_t|\phi y_{t-1}, v)$, where $\boldsymbol{\theta} = (\phi, v)'$. Hence, the conditional likelihood conditional on y_1 is

$$
\begin{aligned}
p(y_{2:T}|y_1, \boldsymbol{\theta}) &= (2\pi v)^{-\frac{T-1}{2}} \exp\left(-\frac{\sum_{t=2}^{T}(y_t - \phi y_{t-1})^2}{2v}\right) \\
&\propto (v)^{-\frac{T-1}{2}} \exp\left(-\frac{\sum_{t=2}^{T}(y_t - \phi y_{t-1})^2}{2v}\right)
\end{aligned}
$$

   Hence, ther conditional log likelihood is $\log L := \log p(y_{2:T}|y_1, \boldsymbol{\theta}) \propto (-(T-1)/2)\log v - (\sum_{t=2}^{T}(y_t - \phi y_{t-1})^2/2v)$. To find the MLE of (ϕ, v), we solve the two first order conditions:

$$
\frac{\partial \log L}{\partial \phi} = \frac{(-2)\sum_{t=2}^{T}(y_t - \phi y_{t-1})(-y_{t-1})}{2v} \equiv 0 \tag{1}
$$

$$
\frac{\partial \log L}{\partial v} = \frac{-(T-1)}{2v} - \frac{(-1)\sum_{t=2}^{T}(y_t - \phi y_{t-1})^2}{2v^2} \equiv 0 \tag{2}
$$

   By (1), $\sum_{t=2}^{T}(y_t - \phi y_{t-1})y_{t-1} = 0$. Hence,

$$
\hat{\phi}_{cMLE} = \left(\sum_{t=2}^{T} y_{t-1}^2\right)^{-1}\left(\sum_{t=2}^{T} y_t y_{t-1}\right)
$$

   . Then by (2), we conclude that

$$
\hat{v}_{cMLE} = \frac{\sum_{t=2}^{T}(y_t - \phi y_{t-1})^2}{T-1}
$$

   .
   To check second order sufficient conditions, we have

$$
\left.\frac{\partial^2 \log L}{\partial \phi^2}\right|_{(\hat{\phi}_{cMLE}, \hat{v}_{cMLE})} = \frac{-\sum_{t=2}^{T} y_{t-1}^2}{v} < 0
$$

and

$$\left.\frac{\partial^2 \log L}{\partial v^2}\right|_{(\hat{\phi}_{cMLE},\hat{v}_{cMLE})} = \frac{(T-1)}{2v^2} - \frac{(-1)\sum_{t=2}^{T}(y_t - \phi y_{t-1})^2}{v^3} = \frac{-(T-1)^3}{Q(\phi)^2} < 0$$

where $Q(\phi) = \sum_{t=2}^{T}(y_t - \phi y_{t-1})^2$. Also,

$$\left(\left.\frac{\partial^2 \log L}{\partial v \partial \phi}\right|_{(\hat{\phi}_{cMLE},\hat{v}_{cMLE})}\right)^2 - \left(\left.\frac{\partial^2 \log L}{\partial \phi^2}\right|_{(\hat{\phi}_{cMLE},\hat{v}_{cMLE})}\right)\left(\left.\frac{\partial^2 \log L}{\partial v^2}\right|_{(\hat{\phi}_{cMLE},\hat{v}_{cMLE})}\right) < 0$$

This guarantees that $(\hat{\phi}_{cMLE}, \hat{v}_{cMLE})$ is the MLE of (ϕ, v) for the conditional likelihood.
We create a AR(1) dataset of size 100 using $\phi = 0.9$, $v = 1$ with seed number 123456 in `R`. The conditional MLE for ϕ and v are $\hat{\phi}_{cmle} = 0.939$ and $\hat{v}_{cMLE} = 0.969$, respectively.

(b) Find the MLE of (ϕ, v) for the unconditional likelihood (1.17).

Solution:

The equation (1.17) in the text is $p(y_{1:n}|\boldsymbol{\theta}) = \frac{(1-\phi^2)^{1/2}}{(2\pi v)^{n/2}} \exp\left[-\frac{Q^*(\phi)}{2v}\right]$ with $Q^*(\phi) = y_1^2(1-\phi^2) + \sum_{t=2}^{n}(y_t - \phi y_{t-1})^2$.

Since this unconditional likelihood is a nonlinear complicated function, we can use the Newton-Raphson method to obtain the MLE. Before employeeing the Newton-Raphson method, we first calculate its gradient and Hessian matrix.

Let $g(\boldsymbol{\theta}) = \log p(y_{1:n}|\boldsymbol{\theta}) \propto \log(1-\phi^2) - n \log v - Q^*(\phi)/v$. The gradient and Hessian matrix are as follows.

$$\begin{aligned}
\frac{\partial g}{\partial \phi} &= \frac{-2\phi}{(1-\phi^2)} + (2/v)\left(y_1^2\phi + \sum_{t=2}^{n} y_y y_{t-1} - \phi \sum_{t=2}^{n} y_{t-1}^2\right) \\
\frac{\partial g}{\partial v} &= (-n/v) + \frac{Q^*(\phi)}{v^2} \\
\frac{\partial^2 g}{\partial \phi^2} &= \frac{-2(1+\phi^2)}{(1-\phi^2)^2} + (2/v)\left(y_1^2 - \sum_{t=2}^{n} y_{t-1}^2\right) \\
\frac{\partial^2 g}{\partial v^2} &= (n/v^2) - \frac{2Q^*(\phi)}{v^3} \\
\frac{\partial^2 g}{\partial \phi \partial v} &= (-2/v^2)\left(y_1^2\phi + \sum_{t=2}^{n} y_y y_{t-1} + \phi \sum_{t=2}^{n} y_{t-1}^2\right)
\end{aligned}$$

After having this information, we can now use the Newton Raphson iteration to get the MLE for (ϕ, v). The following shows the unconditional MLE estimators using the Newton-Raphson algorithm.

```
## Iteration = 2
## The MLE for (phi, v) = (0.937, 0.843)
## Iteration = 3
## The MLE for (phi, v) = (0.932, 0.942)
## Iteration = 4
## The MLE for (phi, v) = (0.931, 0.97)
## Iteration = 5
## The MLE for (phi, v) = (0.931, 0.972)
## Iteration = 6
## The MLE for (phi, v) = (0.931, 0.972)
```

The result above uses starting value $\boldsymbol{\theta}^{(0)} = (0.8, 0.8)$. Note that a good starting value $\boldsymbol{\theta}^{(0)}$ is important because the algorithm may not converge for values in regions where the Hessian is not positive definite.

(c) Assume that v is known. Find the MAP estimator of ϕ under a uniform prior $p(\phi) = U(\phi|-1, 1)$ for the conditional and unconditional likelihoods.
Solution:
Since $p(\phi) = U(\phi|-1, 1) = 1/2$ if $\phi \in (-1, 1)$, for conditional likelihood, the posterior is proportional to the conditional likelihood times an indicator function $I_{(-1,1)}(\phi)$. Hence, the MAP for conditional likelihood is

$$\hat{\phi}_{cMAP} = \frac{\sum_{t=2}^{T} y_{t-1}^2}{\sum_{t=2}^{T} y_t y_{t-1}}$$

if $\hat{\phi}_{cMAP} \in (-1, 1)$. If the optimal $\phi > 1$, $\hat{\phi}_{cMAP}$ should be 1, and if the optimal $\phi < -1$, $\hat{\phi}_{cMAP}$ should be -1 since conditional likelihood is unimodal (normal) distribution.

The unconditional likelihood is similar to the conditional one. But if the function we want to optimize is multimodal, it may be difficult to get the $\hat{\phi}_{MAP}$.

2. **Chapter 1 Problem 3** Show that the distributions of $(\phi|\mathbf{y}, \mathbf{F}))$ and $(v|\mathbf{y}, \mathbf{F})$ obtained for the AR(1) reference analysis are those given in Example 1.6.
Solution:
An AR(1) process can be written as a linear regression model if $\mathbf{y} = (y_2, \ldots, y_n)'$, $\mathbf{F} = (y_1, \ldots, y_{n-1})'$, $\phi = \boldsymbol{\beta}$, and $\boldsymbol{\epsilon} = (\epsilon_2, \ldots, \epsilon_n)'$ with $\boldsymbol{\epsilon} \sim N(0, v\mathbf{I}_{n-1})$. In this case, $\mathbf{y} = \mathbf{F}'\boldsymbol{\beta} + \boldsymbol{\epsilon}$ is an AR(1) process.

Notice that when using reference prior $p(\boldsymbol{\beta}, v) \propto 1/v$, one has the following results:

- $\boldsymbol{\beta}|\mathbf{F}, \mathbf{y} \sim T_{n-p}(\hat{\boldsymbol{\beta}}, s^2(\mathbf{F}\mathbf{F}')^{-1})$, where the location parameter $\hat{\boldsymbol{\beta}}$ is the MAP for $\boldsymbol{\beta}$, and scale parameter s^2 is an estimator for v.
- $v|\mathbf{y}, \mathbf{F} \sim IG\left(\frac{n-p}{2}, \frac{(n-p)s^2}{2}\right)$.

Here, for the AR(1) process, $p = 1$, $n = n - 1$, and $\boldsymbol{\beta} = \phi$. So $\phi|\mathbf{y}, \mathbf{F} \sim T_{n-1-1}(\hat{\phi}, s^2(\mathbf{F}\mathbf{F}')^{-1})$. Now we need to find the $\hat{\phi}_{MAP}$ and s^2.

Since $p(\phi, v|\mathbf{y}, \mathbf{F}) \propto p(\mathbf{y}|\mathbf{F}, \phi, v)/v$, for conditional likelihood, $\hat{\phi}_{MAP} = \hat{\phi}_{cMLE}$. Also, under conditional likelihood, Normal likelihood gives us $\hat{\phi}_{cMLE} = \hat{\phi}_{ols}$. Hence, $\hat{\phi}_{MAP} = \hat{\phi}_{ols}$, and $\hat{\phi}_{ols} = (\mathbf{F}\mathbf{F}')^{-1}\mathbf{F}\mathbf{y} = \left(\sum_{t=1}^{n-1} y_t^2\right)^{-1} \left(\sum_{t=2}^{n} y_t y_{t-1}\right)$. Thus, the mode $m(y_{1:n}) = \hat{\phi}_{MAP} = \hat{\phi}_{cMLE} = \hat{\phi}_{ols} = \frac{\sum_{t=2}^{n} y_t y_{t-1}}{\sum_{t=1}^{n-1} y_t^2}$.

Now we have to figure out what the scale is. Instead of using MLE for v, $R/(n-1)$, as the estimator for v, more usually, one uses the unbiased estimate of v, $s^2 = R/(n-1-1)$, where $R = (\mathbf{y} - \hat{\mathbf{y}})'(\mathbf{y} - \hat{\mathbf{y}}) = (\mathbf{y} - \mathbf{F}'\hat{\boldsymbol{\beta}})'(\mathbf{y} - \mathbf{F}'\hat{\boldsymbol{\beta}})$ in general least square models. Here,

$$\begin{aligned}
R = (\mathbf{y} - \hat{\mathbf{y}})'(\mathbf{y} - \hat{\mathbf{y}}) &= \mathbf{y}'\mathbf{y} - 2(\hat{\mathbf{y}}'\mathbf{y}) + \hat{\mathbf{y}}'\hat{\mathbf{y}} \\
&= \sum_{t=2}^{n} y_t^2 - 2\left(\frac{\sum_{t=2}^{n} y_t y_{t-1}}{\sum_{t=1}^{n-1} y_t^2}\right)\left(\sum_{t=2}^{n} y_t y_{t-1}\right) + \left(\frac{\sum_{t=2}^{n} y_t y_{t-1}}{\sum_{t=1}^{n-1} y_t^2}\right)^2 \left(\sum_{t=1}^{n-1} y_t^2\right) \\
&= \sum_{t=2}^{n} y_t^2 - 2\frac{\left(\sum_{t=2}^{n} y_t y_{t-1}\right)^2}{\sum_{t=1}^{n-1} y_t^2} + \frac{\left(\sum_{t=2}^{n} y_t y_{t-1}\right)^2}{\sum_{t=1}^{n-1} y_t^2} \\
&= \sum_{t=2}^{n} y_t^2 - \frac{\left(\sum_{t=2}^{n} y_t y_{t-1}\right)^2}{\sum_{t=1}^{n-1} y_t^2}
\end{aligned}$$

As a result,

$$
\begin{aligned}
C(y_{1:n}) &= R(\mathbf{FF}')^{-1} \\
&= \left(\sum_{t=2}^{n} y_t^2 - \frac{\left(\sum_{t=2}^{n} y_t y_{t-1}\right)^2}{\sum_{t=1}^{n-1} y_t^2} \right) \left(\sum_{t=1}^{n-1} y_t^2 \right)^{-1} \\
&= \frac{\sum_{t=2}^{n} y_t^2 \sum_{t=2}^{n} y_{t-1}^2 - \left(\sum_{t=2}^{n} y_t y_{t-1}\right)^2}{\left(\sum_{t=1}^{n-1} y_t^2\right)^2}
\end{aligned}
$$

Thus, we conclude that $(\phi|\mathbf{y}, \mathbf{F}) \sim t_{(n-2)}\left(m(y_{1:n}), \dfrac{C(y_{1:n})}{n-2}\right)$, and $(v|\mathbf{y}, \mathbf{F}) \sim IG\left(\frac{n-2}{2}, \frac{(n-2)s^2}{2}\right)$.

3. **Chapter 2 Problem 4** Show that the distributions of $(\phi|\mathbf{y}, \mathbf{F}))$ and $(v|\mathbf{y}, \mathbf{F}))$ obtained for the AR(1) conjugate analysis are those given in Example 1.7.
Solution:
From the textbook, we know that in general, when using conjugate priors, $(\boldsymbol{\beta}|y_{1:n}, \mathbf{F}, v) \sim N(\mathbf{m}, v\mathbf{C})$, with $\mathbf{m} = \mathbf{m_0} + \mathbf{C_0F}[\mathbf{F}'\mathbf{C_0F} + \mathbf{I}_n]^{-1}(\mathbf{y} - \mathbf{F}'\mathbf{m_0})$, and $\mathbf{C} = \mathbf{C_0} - \mathbf{C_0F}[\mathbf{F}'\mathbf{C_0F} + \mathbf{I}_n]^{-1}\mathbf{F}'\mathbf{C_0}$. Also, $(v|\mathbf{F}, y_{1:n}) \sim IG(n^*/2, d^*/2)$ with $n^* = n + n_0$ and $d^* = (\mathbf{y} - \mathbf{F}'\mathbf{m_0})'[\mathbf{F}'\mathbf{C_0F} + \mathbf{I_n}]^{-\mathbf{1}}(\mathbf{y} - \mathbf{F}'\mathbf{m_0}) + d_0$.

By comparing the parameters and coefficients, here, $\mathbf{m_0} = 0$, $\mathbf{C_0} = 1$, $p = 1$, $n = n - 1$ and the fact that $[\mathbf{DV^{-1}D}' + \mathbf{R^{-1}}]^{-1} = \mathbf{R} - \mathbf{RD}[\mathbf{D}'\mathbf{RD} + \mathbf{V}]^{-1}\mathbf{D}'\mathbf{R}$, we have

$$
\begin{aligned}
\mathbf{m} &= \mathbf{m_0} + \mathbf{F}[\mathbf{F}'\mathbf{C_0F} + \mathbf{I_{n-1}}]^{-\mathbf{1}}(\mathbf{y} - \mathbf{F}'\mathbf{0}) \\
&= \mathbf{F}(\mathbf{I} - \mathbf{IF}'[\mathbf{FF}' + \mathbf{1}]^{-\mathbf{1}}\mathbf{F})\mathbf{y} \\
&= \mathbf{Fy} - \mathbf{FF}'[\mathbf{FF}' + \mathbf{1}]^{-\mathbf{1}}\mathbf{Fy} \\
&= \sum_{t=1}^{n-1} y_t y_{t+1} - \left(\sum_{t=1}^{n-1} y_t^2 \Big/ \left(\sum_{t=1}^{n-1} y_t^2 + 1 \right) \right) \sum_{t=1}^{n-1} y_t y_{t+1} \\
&= \frac{(\sum_{t=1}^{n-1} y_t y_{t+1})(\sum_{t=1}^{n-1} y_t^2 + 1) - (\sum_{t=1}^{n-1} y_t^2)(\sum_{t=1}^{n-1} y_t y_{t+1})}{\sum_{t=1}^{n-1} y_t^2 + 1} \\
&= \frac{\sum_{t=1}^{n-1} y_t y_{t+1}}{\sum_{t=1}^{n-1} y_t^2 + 1}.
\end{aligned}
$$

$$
\begin{aligned}
\mathbf{C} = C &= 1 - \mathbf{F}[\mathbf{F}'\mathbf{C_0F} + \mathbf{I_{n-1}}]^{-\mathbf{1}}\mathbf{F}' \\
&= 1 - \mathbf{F}(\mathbf{I} - \mathbf{IF}'[\mathbf{FF}' + \mathbf{1}]^{-\mathbf{1}}\mathbf{F})\mathbf{F}' \\
&= 1 - \mathbf{FF}' - \mathbf{FF}'[\mathbf{FF}' + \mathbf{1}]^{-\mathbf{1}}\mathbf{FF}' \\
&= 1 - \frac{\mathbf{FF}'(\mathbf{FF}' + \mathbf{1}) - \mathbf{FF}'\mathbf{FF}'}{\mathbf{FF}' + \mathbf{1}} \\
&= 1 - \frac{\mathbf{FF}'}{\mathbf{FF}' + \mathbf{1}} \\
&= \frac{1}{\sum_{t=1}^{n-1} y_t^2 + 1}.
\end{aligned}
$$

Hence, $(\phi|\mathbf{y}, \mathbf{F}, v) \sim N(m, vC)$ with $m = \dfrac{\sum_{t=1}^{n-1} y_t y_{t+1}}{\sum_{t=1}^{n-1} y_t^2 + 1}$, $C = \dfrac{1}{\sum_{t=1}^{n-1} y_t^2 + 1}$. Also, $n^* = n - 1 + n_0$, and

$$
\begin{aligned}
d^* &= (\mathbf{y} - \mathbf{F}'\mathbf{0})'[\mathbf{F}'\mathbf{C_0}\mathbf{F} + \mathbf{I_{n-1}}]^{-1}(\mathbf{y} - \mathbf{F}'\mathbf{0}) + d_0 \\
&= \mathbf{y}'(\mathbf{I} - \mathbf{I}\mathbf{F}'[\mathbf{F}\mathbf{F}' + \mathbf{1}]^{-1}\mathbf{F})\mathbf{y} + d_0 \\
&= \mathbf{y}'\mathbf{y} - \mathbf{y}'\mathbf{F}'[\mathbf{F}\mathbf{F}' + \mathbf{1}]^{-1}\mathbf{F}\mathbf{y} + d_0 \\
&= \sum_{t=2}^{n} y_t^2 - \frac{\left(\sum_{t=1}^{n-1} y_t y_{t+1}\right)^2}{\sum_{t=1}^{n-1} y_t^2 + 1} + d_0.
\end{aligned}
$$

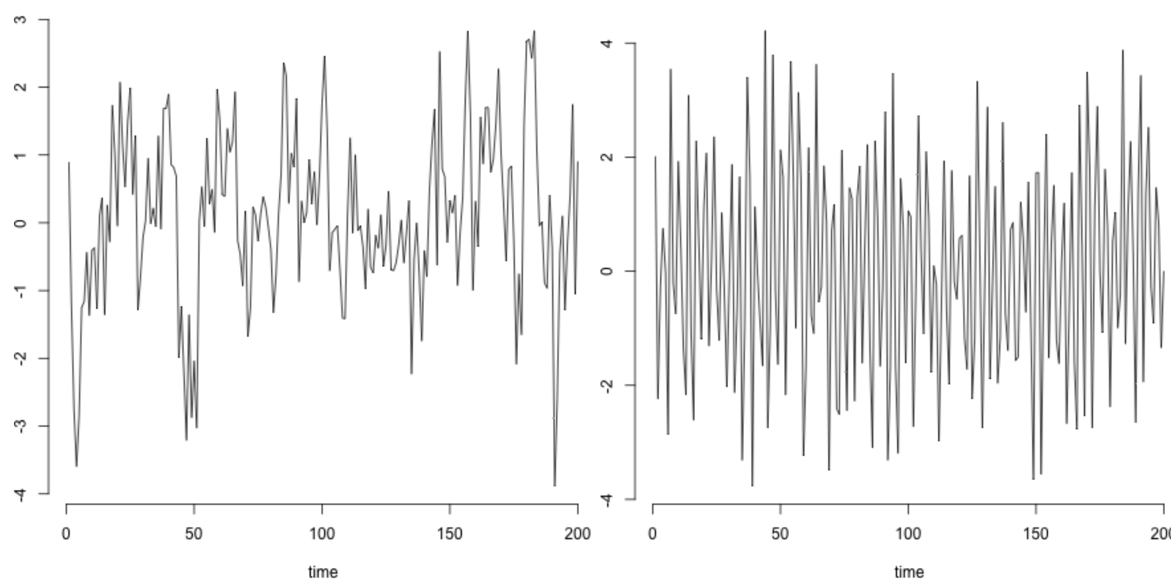
Hence, $(v|\mathbf{y}, \mathbf{F}) \sim IG(n^*/2, d^*/2)$ with $n^* = n - 1 + n_0$ and $d^* = \sum_{t=2}^{n} y_t^2 y - \frac{\left(\sum_{t=1}^{n-1} y_t y_{t+1}\right)^2}{\sum_{t=1}^{n-1} y_t^2 + 1} + d_0$.

4. **Chapter 2 Problem 5** Consider the following models:

$$
\begin{aligned}
y_t &= \phi_1 y_{t-1} + \phi_2 y_{t-2} + \epsilon_t && (3) \\
y_t &= a\cos(2\pi\omega_0 t) + b\sin(2\pi\omega_0 t) + \epsilon_t && (4)
\end{aligned}
$$

with $\epsilon \sim N(0, v)$.

(a) Sample 200 observations from each model using your favorite choice of the parameters. Make sure your choice of (ϕ_1, ϕ_2) in model (3) lies in the stationary region. That is, choose ϕ_1 and ϕ_2 such that $-1 < \phi_2 < 1$, $\phi_1 < 1 - \phi_2$, and $\phi_1 > \phi_2 - 1$.
Solution:



(a) Time Series of (3) (b) Time Series of (4)

Figure 1: 200 observations from (3) and (4)

Figure 1 shows 200 observations from model 3 and 4 with model 3 parameters $\phi_1 = 0.5$, $\phi_2 = 0.15$ and $v = 1$ and model 4 parameters $a = 1$, $b = 2$ and $\omega_0 = 0.3$.

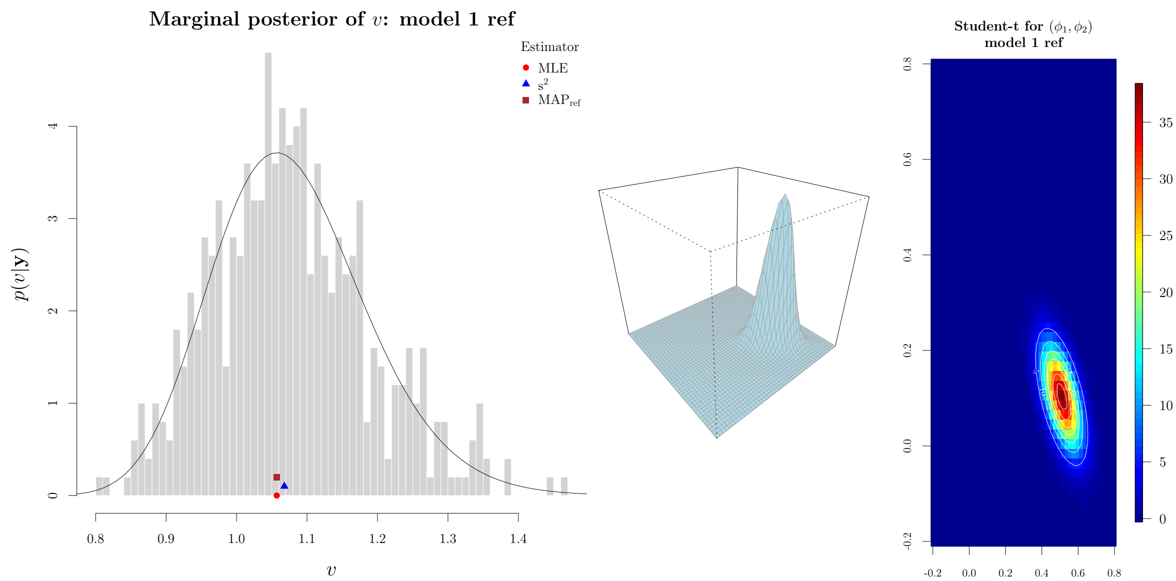
(b) Find the MLEs of the parameters in model (3) and (4). Use the conditional likelihood for model (3).
Solution:
Both models can be written in a matrix form of linear regression models, $\mathbf{y} = \mathbf{F}'\boldsymbol{\beta} + \boldsymbol{\epsilon}$ and

because of Normality, the MLEs for the regression coefficients are OLS estimates, which is of form $\hat{\boldsymbol{\beta}} = (\mathbf{F}\mathbf{F}')^{-1}\mathbf{F}\mathbf{y}$. The ML estimate for v is R/n, where $R = (\mathbf{y} - \mathbf{F}'\hat{\boldsymbol{\beta}})'(\mathbf{y} - \mathbf{F}'\hat{\boldsymbol{\beta}})$.
Hence, given the generated data set, the ML estimate for (ϕ_1, ϕ_2) are $(\hat{\phi}_{1,MLE}, \hat{\phi}_{2,MLE}) = (0.51, 0.102)$. The MLE for v of model (3) is $\hat{v}^{(1)}_{MLE} = 1.057$
For model (4), the ML estimates are $(\hat{a}_{MLE}, \hat{b}_{MLE}) = (0.994, 2.044)$ and $\hat{v}^{(2)}_{MLE} = 0.965$.

(c) Find the MAP estimators of the model parameters under the reference prior. Again, use the conditional likelihood for model (3).
Solution:

Under reference prior, OLS estimates for (ϕ_1, ϕ_2) and (a, b) are also MAPs. Hence $(\hat{\phi}_{1,MAP}, \hat{\phi}_{2,MAP}) = (0.51, 0.102)$ and $(\hat{a}_{MAP}, \hat{b}_{MAP}) = (0.994, 2.044)$. From Problem 2, we learn that $(v|\mathbf{y}, \mathbf{F}) \sim IG\left(\frac{n-2}{2}, \frac{(n-2)s^2}{2}\right)$. Hence the MAP for v is R/n, and so $\hat{v}^{(1)}_{MAP} = 1.057$ and $\hat{v}^{(2)}_{MAP} = 0.965$.

(d) Sketch $p(v|y_{1:n})$ and $p(\phi_1, \phi_2|y_{1:n})$ for model (3).
Solution: Please see Figure 2.



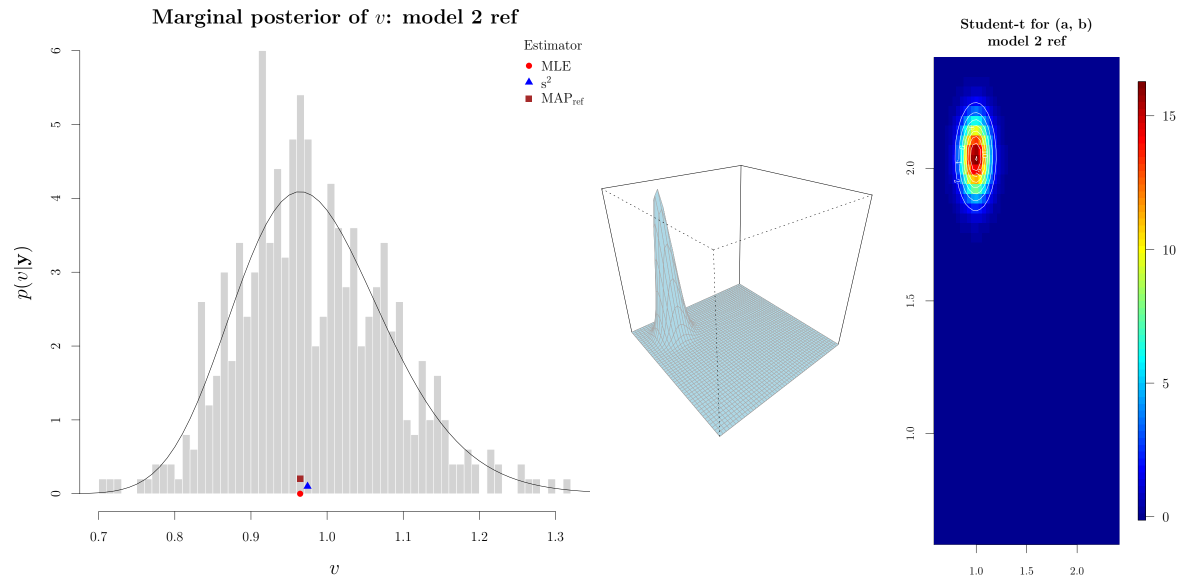
(a) Marginals of v (b) Marginals of (ϕ_1, ϕ_2)

Figure 2: Marginals of v and (ϕ_1, ϕ_2) of model (3) under reference prior

(e) Sketch $p(a, b|y_{1:n})$ and $p(a|y_{1:n})$ in model (4).
Solution:
Please see Figure 3.

(f) Perform a conjugate Bayesian analysis, i.e., repeat (c) to (e) assuming conjugate prior distributions in both models. Study the sensitivity of the posterior distributions to the choice of the hyperparameters in the prior.
Solution:

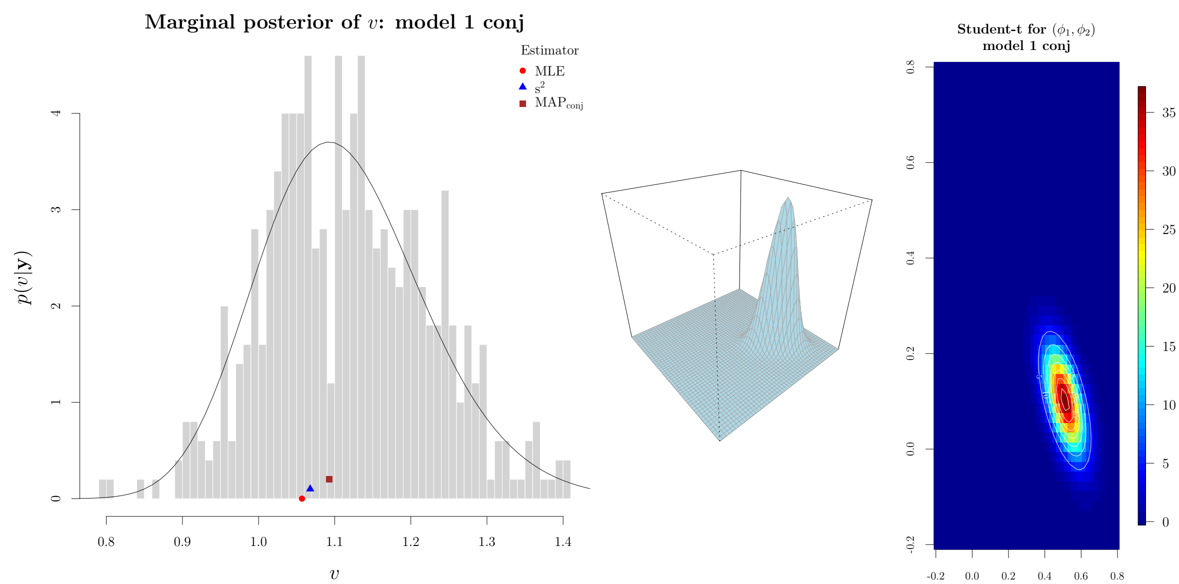
For model 1, with the prior $(\phi_1, \phi_2)' \sim N_2(\mathbf{m_0}, \mathbf{C_0})$, where $\mathbf{m_0} = (0.2, -0.5)$ and $\mathbf{C_0} = diag(3, 3)$ and $v \sim IG(n_0, d_0)$ where $n_0 = 10$ and $d_0 = 20$, we have $(\hat{\phi}_{1,MAPconj}, \hat{\phi}_{2,MAPconj}) = (0.51, 0.102)$ and $\hat{v}^{(1)}_{MAPconj} = 1.093$.
For model 2, we use prior $(a, b)' \sim N_2(\mathbf{m_0}, \mathbf{C_0})$, where $\mathbf{m_0} = (2, 3)$ and $\mathbf{C_0} = diag(1, 2)$ and $v \sim IG(n_0, d_0)$ where $n_0 = 10$ and $d_0 = 20$. Under this setting, we have $(\hat{a}_{1,MAPconj}, \hat{b}_{2,MAPconj}) = (1.004, 2.053)$ and $\hat{v}^{(2)}_{MAPconj} = 1.013$



(a) Marginals of v (b) Marginals of (a, b)

Figure 3: Marginals of v and (a, b) of model (4) under reference prior

Please see Figure 4 and Figure 5 for marginals under conjugate priors.



(a) Marginals of v (b) Marginals of (ϕ_1, ϕ_2)

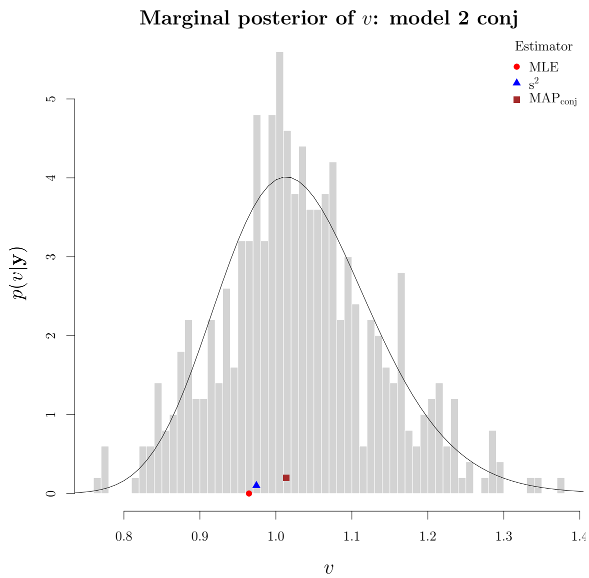
Figure 4: Marginals of v and (ϕ_1, ϕ_2) of model (3) under conjugate prior

Sensitivity analysis can be seen in Figure 6, 7, 8, 9 and 10.

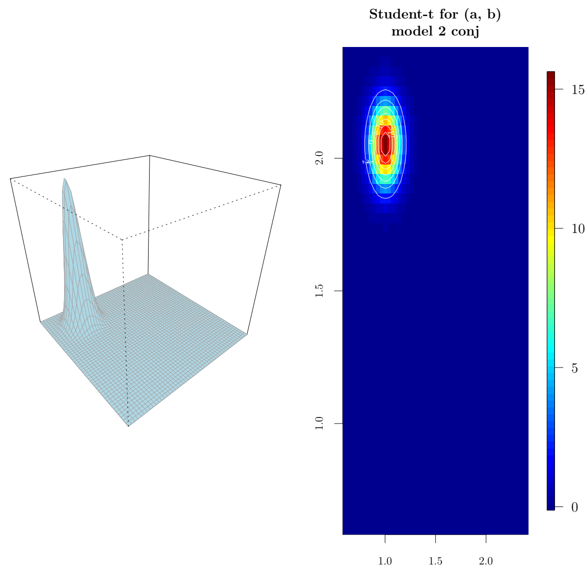
5. **Chapter 2 Problem 7** Sample 1000 observations from the model (1.1). Using a prior distribution of the form $p(\phi^{(i)}) = N(m, c)$, for some c and $i = 1, 2$, $p(\theta) = U(\theta| - a, a)$ and $p(v) = IG(\alpha_0, \beta_0)$, obtain samples from the joint posterior distribution by implementing a Metropolis-Hasting algorithm.
Solution:

First, we sample 1000 observations from the model (1.1). The y and δ series shown in Figure 11 look good.

We can derive full conditionals for ϕ_1, ϕ_2 and v, and perform Metropolis-Hastings step on θ. A



(a) Marginals of v



(b) Marginals of (a, b)

Figure 5: Marginals of v and (a, b) of model (4) under conjugate prior

Gaussian random walk proposal is used and the variance is tuned to have reasonable acceptance rate.

We use 21000 iterations, burn the first 1000 draws and thin the sequence by keeping every 10th draw and discarding the rest. Therefore, 2000 draws are stored for analysis.

Initial values are $\phi_1^{(0)} = 0.5$, $\phi_2^{(0)} = -0.5$, $\theta^{(0)} = -2$ and $v^{(0)} = 2$

Hyperparameters are chosen to be $m = 0$, $c = 1$, $a = 3$, $\alpha_0 = 3$ and $\beta_0 = 0.003$.

	effective size	post. mean	2.5% quantile	97.5% quantile
ϕ_1	1101.40	0.90	0.89	0.90
ϕ_2	2000.00	-0.32	-0.32	-0.32
v	2000.00	0.99	0.93	1.07
θ	967.63	-1.50	-1.51	-1.49

Table 1: Posterior Summary

Table 1 summarizes the posterior sample and Figure 12 shows trace plots, ACFs and histograms of the parameters from the 2000 draws.

the R package *knitr*
(Xie, 2016). It also relied on the R packages
ggplot2 (Plummer et al., 2015) and (Gandrud, 2016) and (Nychka et al., 2016).
The document can be completely reproduced from
source files available on GitHub at:
`https://GitHub.com/christophergandrud/Rep-Res-Examples`.

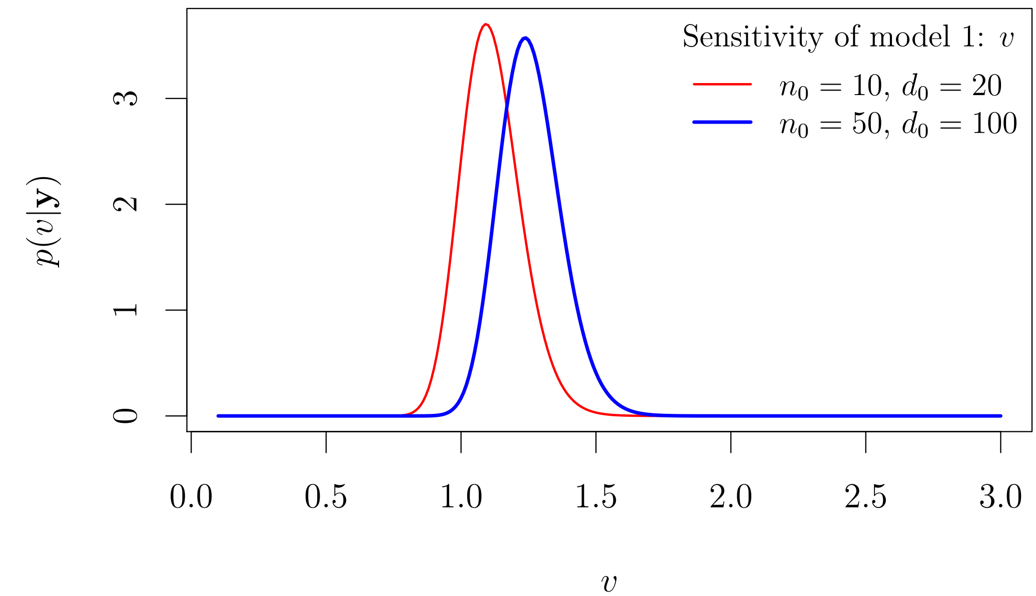


Figure 6: Marginal posterior of v of model 1, sensitivity.

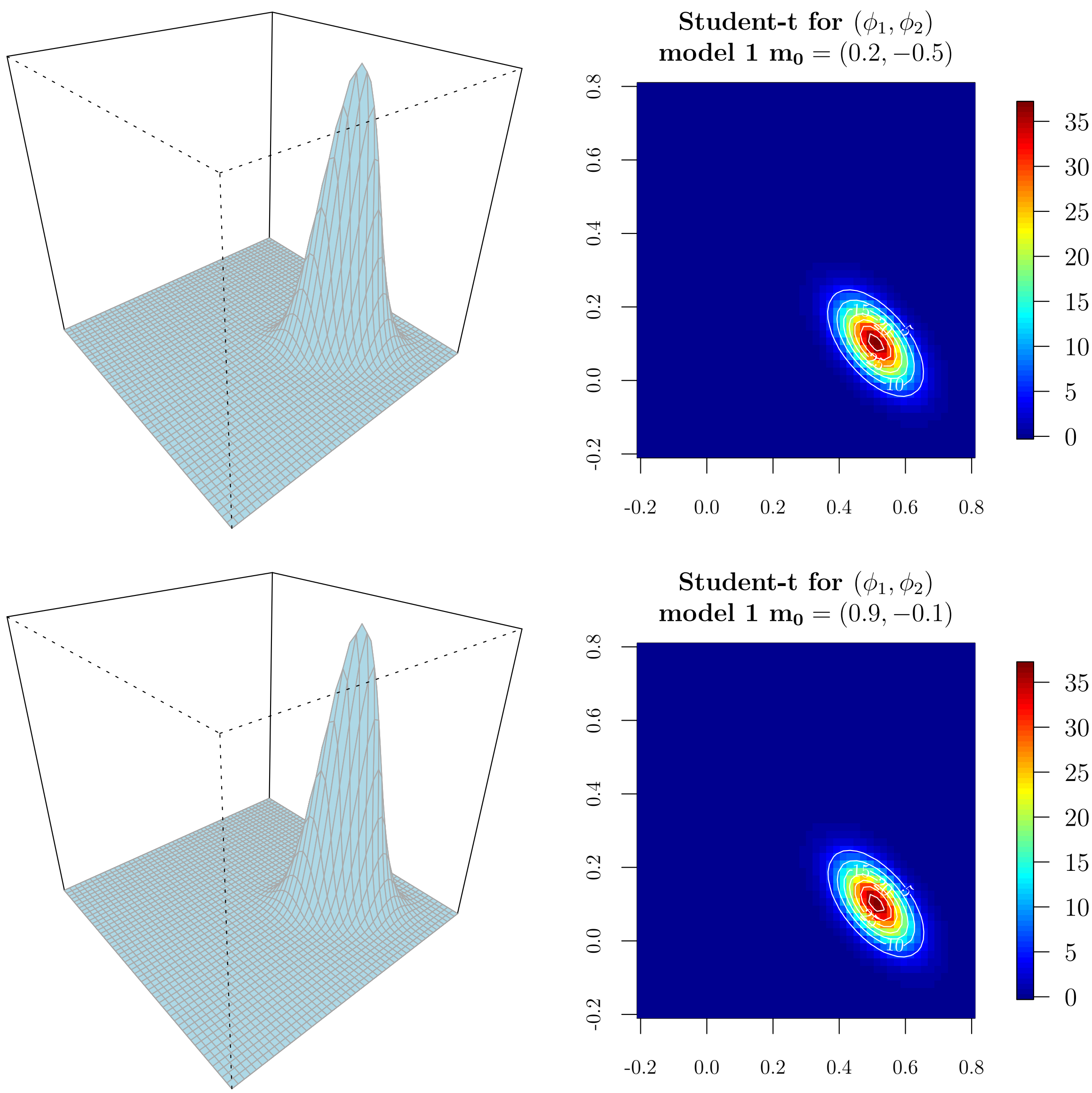


Figure 7: Marginal posterior of (ϕ_1, ϕ_2) of model 1 with same prior of v, sensitivity.

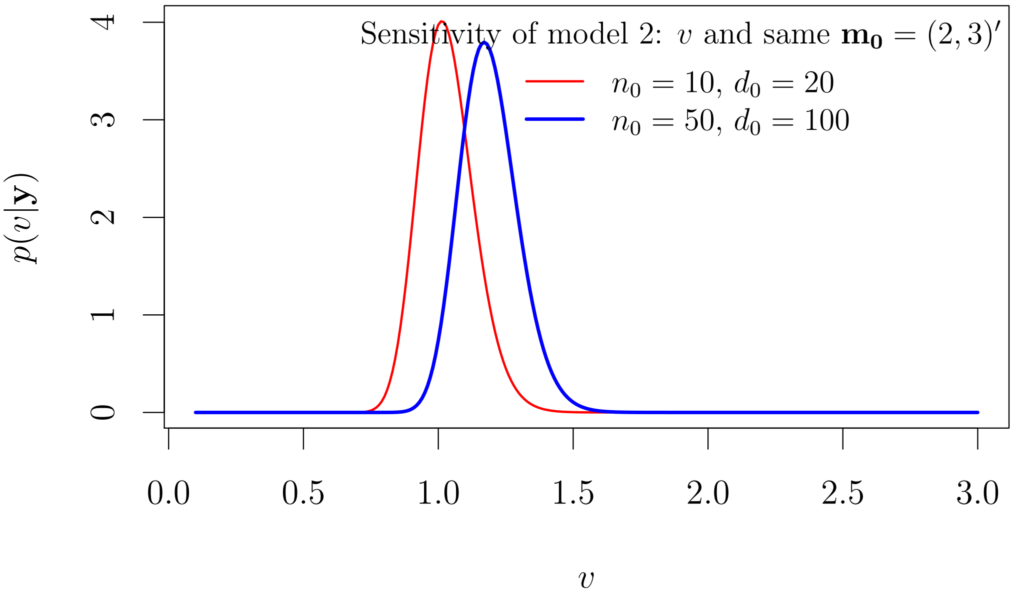


Figure 8: Marginal posterior of v of model 2, sensitivity.

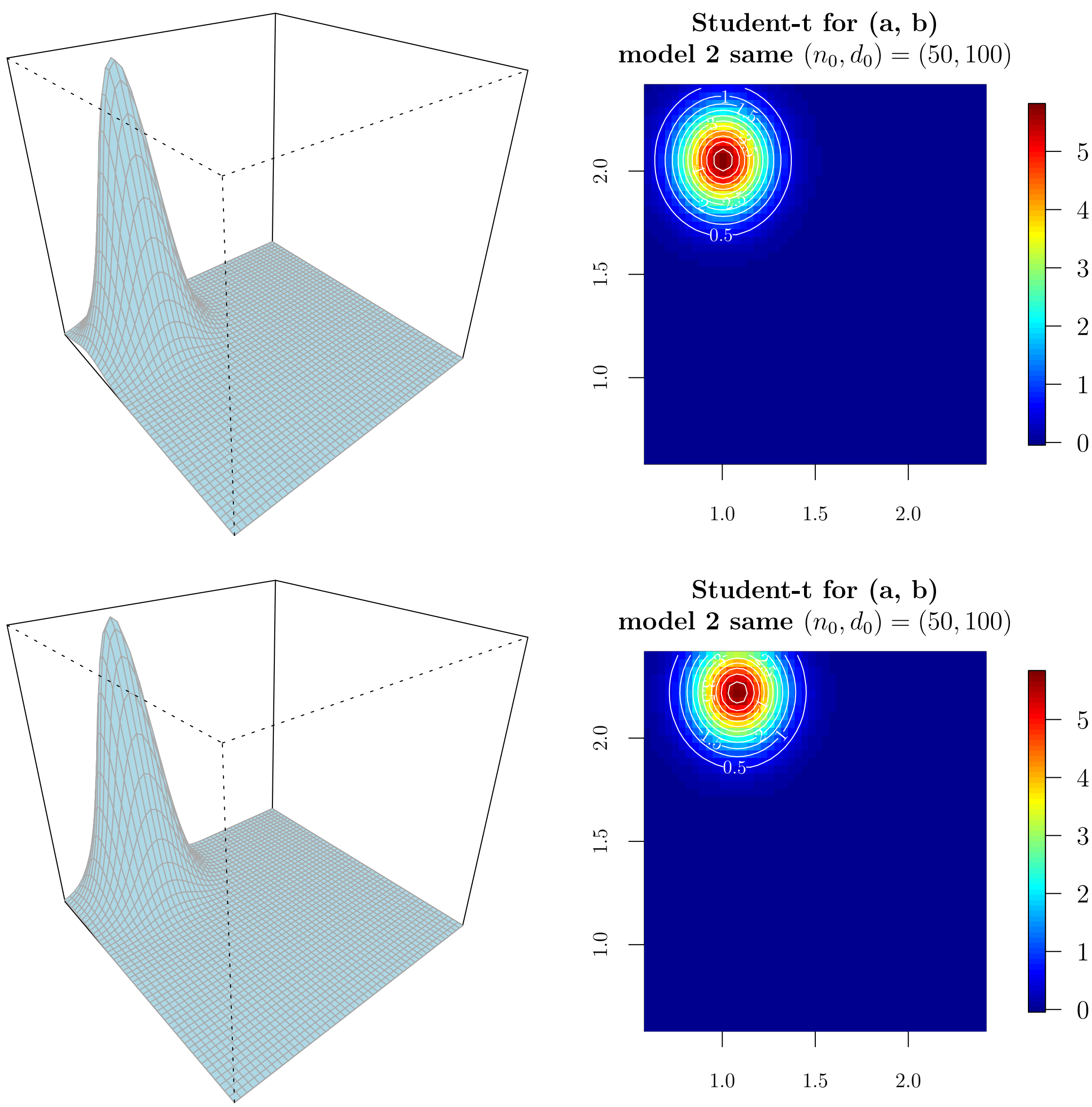


Figure 9: Marginal posterior of (a, b) of model 2 with the same prior on v and different $\mathbf{m_0}$s, sensitivity.

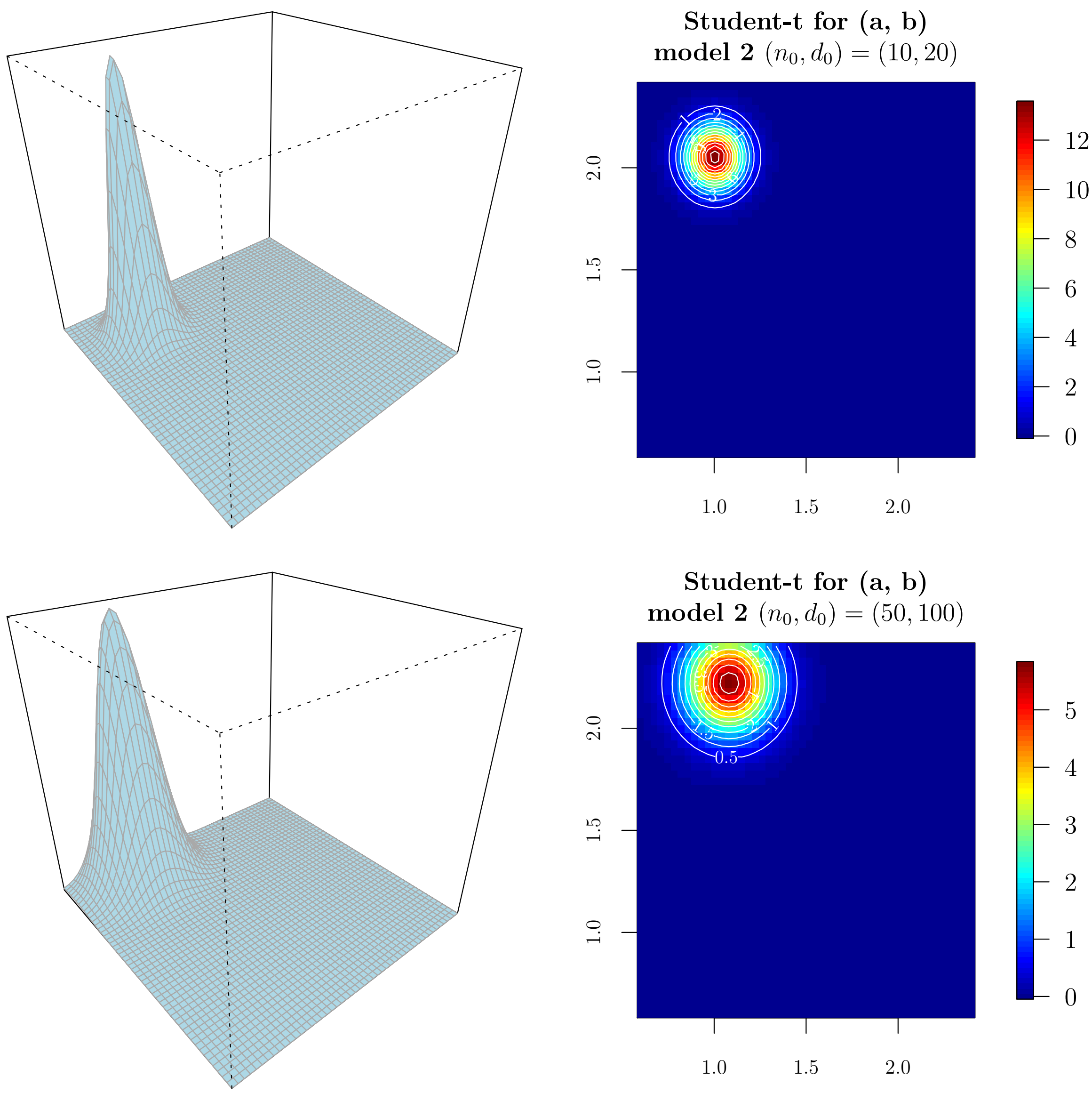


Figure 10: Marginal posterior of (a, b) of model 2 with different priors on v and different $\mathbf{m_0}$s, sensitivity.

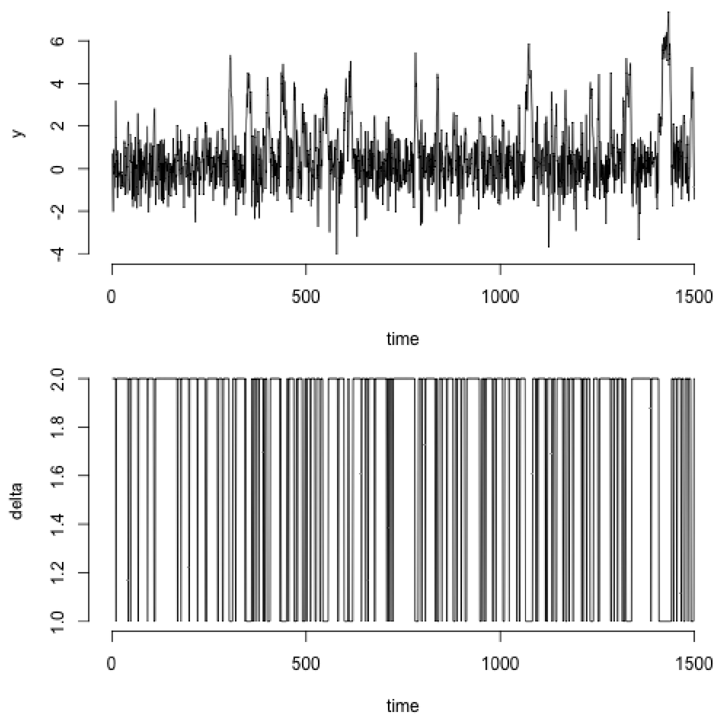


Figure 11: 1000 time series observations of y and δ from the model (1.1).

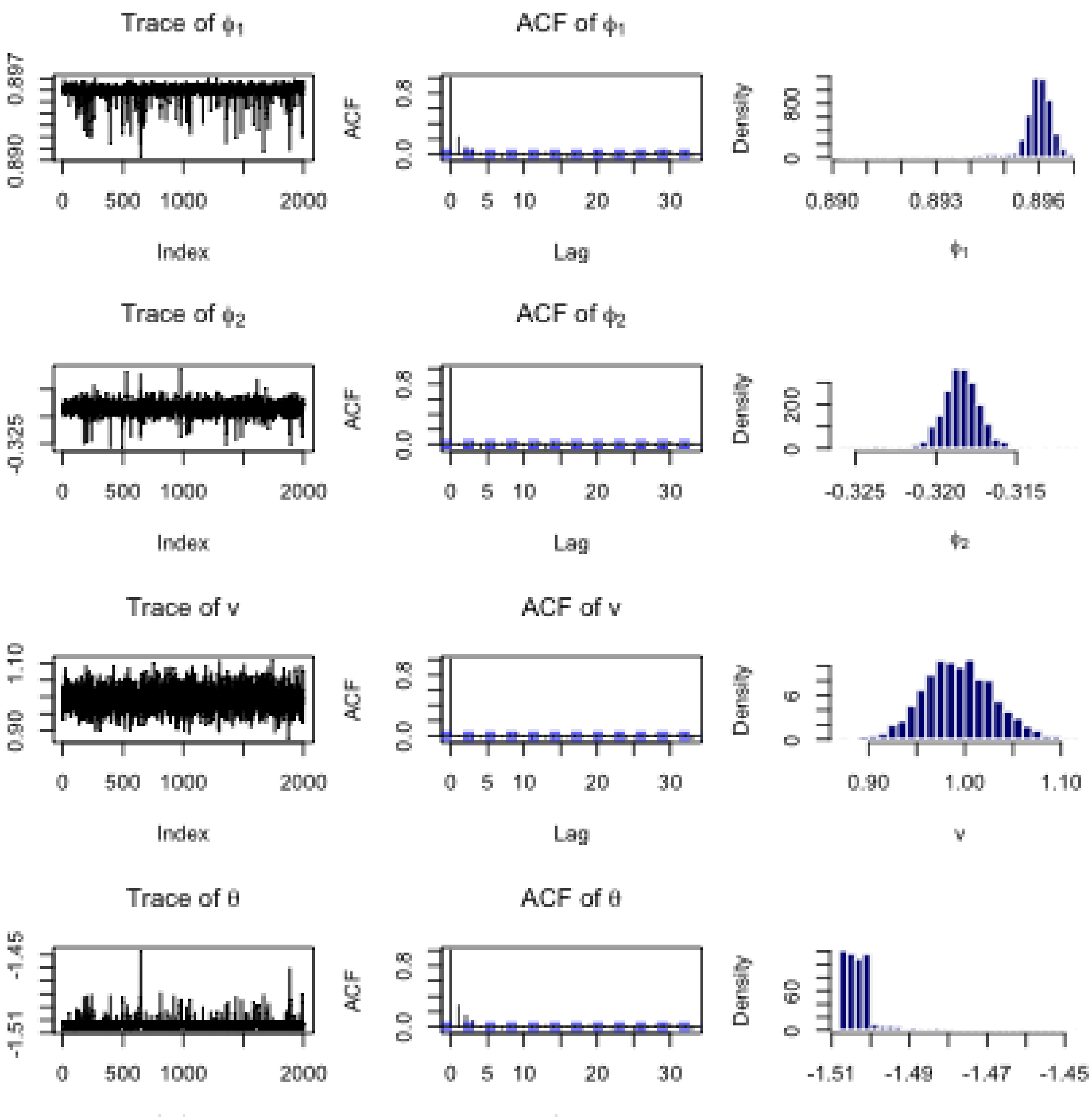


Figure 12: Trace plots, ACFs and histograms of ϕ_1, ϕ_2, v and θ.

2 Code

```
#### Global options
# options(replace.assign=TRUE,width=90)
# A penalty to be applied when deciding to print numeric values in fixed
# or exponential notation
options(scipen = 1, digits = 3, width = 55)

#### Set chunk options
# figure options
library(knitr)
opts_chunk$set(fig.path='figure/hw1_', fig.align='center', crop = TRUE,
               dev = 'png', out.width = "0.49\\linewidth")

# error message options
opts_chunk$set(warning = FALSE, message = FALSE, error = FALSE)
# other options:
# the width of source code
# changes in comments won't aff3ect the cache
opts_chunk$set(tidy.opts = list(width.cutoff = 65),
               cache.comments = FALSE, echo = FALSE)

#### Set hooks
# set margin like margin = c(2, 3, 1, 1)
knit_hooks$set(margin = function(before, options, envir) {
        if (before) {
                m <- options$margin
                if (is.numeric(m) && length(m) == 4L) {
                        par(mar = m)
                }
        } else NULL
})
# crop = TRUE to crop the white margin
knit_hooks$set(crop = hook_pdfcrop)
#### Set up aliases for chink options
set_alias(w = "fig.width", h = "fig.height")

#### Option templates
opts_template$set(
fig.large = list(fig.width = 7, fig.height = 5),
fig.small = list(fig.width = 3.5, fig.height = 3)
)

#### Set the theme
# thm <- knit_theme$get("bclear")
# thm$background
# thm$foreground
# knit_theme$set(thm)
# opts_chunk$set(background = "#f5f5f5")
```

```r
#### Load and cite R packages
# List of packages
PackageUsed <- c("knitr", "pscl", "coda", "parallel",
"doParallel", "tikzDevice", "mvtnorm", "fields", "repmis")
# Load packages
lapply(PackageUsed, library, character.only = TRUE)

#### Read external R scripts
read_chunk("../Analysis/AMS223_HW1_P1.R")
read_chunk('../Analysis/AMS223_HW1_P4.R')
read_chunk('../Analysis/AMS223_HW1_P5.R')
read_chunk('../Analysis/AMS_223_HW1_Q6.R')
# acceptance rate
accept_rate = mcmc_sample$accept / mcmc_sample$count

quan025 <- function(x) {
    quantile(x, prob = 0.025)
}
quan975 <- function(x) {
    quantile(x, prob = 0.975)
}

library(coda)
# draws <- cbind(Phi1, Phi2, V, Theta)[sampleidx, ]
colnames(draws) <- c("$\\phi_1$", "$\\phi_2$", "$v$", "$\\theta$")
result <- round(cbind(apply(draws, 2, effectiveSize),
                      apply(draws, 2, mean),
                      apply(draws, 2, quan025),
                      apply(draws, 2, quan975)), 4)
colnames(result) <- c("effective size", "post. mean", "2.5% quantile",
                      "97.5% quantile")
library(xtable)
print(xtable(result, caption = "Posterior Summary", label = "MCMCresult"),
      sanitize.rownames.function=function(x){x})
# for html output
# print(xtable(result, caption = "Posterior Summary", label = "MCMCresult"),
#       sanitize.rownames.function=function(x){x}, type = "html")
# AMS 223 Time Series
# Cheng-Han Yu, Dept of Applied Math and Statistics, UC Santa Cruz
# HW1 Time series and Baysian inference overview
# Problem 1: Chap 1 problem 2
phi <- 0.9
v <- 1
y0 <- 0.1
n <- 100
y <- rep(NA, n)
# y[1] <- y0
set.seed(123456)
for (i in 1:n) {
```

```
    if (i == 1) {
        y[i] <- phi * y0 + rnorm(1, 0, v)
    } else {
        y[i] <- phi * y[i - 1] + rnorm(1, 0, v)
    }
}
# (a) MLE for conditional likelihood can be derived analytically
ym1 <- y[-n]
yt <- y[-1]
phi_cmle <- sum(yt * ym1) / (sum(ym1 ^ 2))
v_cmle <- sum((yt - phi_cmle * ym1) ^ 2) / (n - 1)
# (b) MLE for the unconditional likelihood using Newton-Raphson method
# Simulation data from AR(1) with phi = 0.9, v = 1 and y0 = 0.1
# with sample size n

# Newton-Raphson iteration starting value
phi0 <- 0.8
v0 <- 0.8
theta0 <- c(phi0, v0)
# Hessian(theta0[1], theta0[2])
# solve(Hessian(theta0[1], theta0[2]))
# gradient of the objective function (1.17)
Qstar <- y[1] ^ 2 * (1 - phi ^ 2) + sum((yt - phi * ym1) ^ 2)
gradient <- function(phi, v) {
    dphi <- (-2 * phi / (1 - phi ^ 2)) + (2 / v) *
        (y[1] ^ 2 * phi + sum(yt * ym1) - phi * sum(ym1 ^ 2))
    dv <- (-n / v) + (1 / v ^ 2) * Qstar
    return (c(dphi, dv))
}

# Hessian of the objective function (1.17)
Hessian <- function(phi, v) {
    dphiphi <- (-2 * (1 + phi ^ 2) / ((1 - phi ^ 2) ^ 2)) +
        (2 / v) * (y[1] ^ 2 - sum(ym1 ^ 2))
    dvv <- (n / (v ^ 2)) - (2 * Qstar / (v ^ 3))
    dphiv <- (-2 / v ^ 2) * (y[1] ^ 2* phi + sum(yt * ym1) -
                                phi * sum(ym1 ^ 2))
    return (matrix(c(dphiphi, dphiv, dphiv, dvv), nrow = 2))
}

# Newton-Raphson iteration
theta1 <- theta0 - solve(Hessian(theta0[1], theta0[2])) %*%
                                gradient(theta0[1], theta0[2])
count <- 1
while (sum(theta1 - theta0) ^ 2 > 1e-8){
    theta0 <- theta1
    theta1 <- theta0 - solve(Hessian(theta0[1], theta0[2])) %*%
        gradient(theta0[1], theta0[2])
    count <- count + 1
```

```r
    cat("Iteration = ", count, "\n", sep = "")
    cat("The MLE for (phi, v) = (", theta1[1], ", ", theta1[2], ")", "\n",
        sep = "")
}
################################################################################
# AMS 223 Time Series
# Cheng-Han Yu, Dept of Applied Math and Statistics, UC Santa Cruz
# HW1 Time series and Baysian inference overview
# Problem 4: Chap 1 problem 5
################################################################################
# (a) Sample 200 obs
n <- 200
v <- 1
# choose model 1 parameters
phi1 <- 0.5
phi2 <- 0.15
# (a.1) sample 200 obs from model 1
# set.seed(1234)
y <- stats::arima.sim(n = n, model = list(ar = c(phi1, phi2)), sd = v)
# head(y)
plot(y, type = "l", xlab = "time", ylab = "", axes = F)
axis(1)
axis(2)

a <- 1
b <- 2
w0 <- 0.3
# (a.2) sample 200 obs from model 2
angle = 2 * w0 * pi * 1:n
x = a * cos(angle) + b * sin(angle) + rnorm(n, 0, v)
# head(x)
plot(x, type = "l", xlab = "time", ylab = "", axes = F)
axis(1)
axis(2)
# (b) Find MLE
# (b.1) MLE for model 1
f1 <- y[-c(1, n)]  # 1st col of F'
f2 <- y[-c(n - 1, n)]  # 2nd col of F'
Ft <- cbind(f1, f2)  # Ft = F' in the model
mle1 <- chol2inv(chol(crossprod(Ft))) %*% t(Ft) %*% y[-c(1, 2)]
R1 <- sum((y[3:n] - Ft %*% mle1) ^ 2)
mle_v1 <- R1 / (n - 2)  # sample size n-2
s2_1 <- R1 / (n - 2 - 2)  # 2 parameters phi1, phi2
# (b.2) MLE for model 2
# depend on Q4a2
Xt <- cbind(cos(angle), sin(angle))
mle2 <- chol2inv(chol(crossprod(Xt))) %*% t(Xt) %*% x

R2 <- sum((x - Xt %*% mle2) ^ 2)
```

```r
mle_v2 <- R2 / (n)  # sample size n

s2_2 <- R2 / (n - 2) # 2 parameters a, b
# (c) Find MAP
# (c.1) MAP for model 1
# (phi1, phi2) same as MLE
map1 <- mle1

# v is IG((n - 2 - 2) / 2), (n - 2 - 2) * s ^ 2 / 2).
# v_map = mode of IG = ((n - 2 - 2) * s ^ 2 / 2) / ((n - 2 - 2) / 2 + 1)
map_v1 <- ((n - 2 - 2) * s2_1 / 2) / ((n - 2 - 2) / 2 + 1)
# (c.2) MAP for model 2
# (phi1, phi2) same as MLE
map2 <- mle2

# v is IG((n - 2) / 2), (n - 2) * s ^ 2 / 2).
# v_map = (n - 2) * s ^ 2 / 2 /((n - 2) / 2 + 1)
map_v2 <- ((n - 2) * s2_2 / 2) / ((n - 2) / 2 + 1)
suppressMessages(library(pscl))
library(tikzDevice)
alpha1 <- (n - 4) / 2
beta1 <- (n - 4) * s2_1/2
v1 <- seq(0.1, 3, 0.01)
sample1 <- pscl::rigamma(500, alpha1, beta1)
hist_v <- function(samp, mdl, prior.type, alpha, beta, mle, s2, map) {
    par(mar = c(4, 4, 2, .1))
    hist(samp, prob = T,
         main = paste("Marginal posterior of $v$: model", mdl, prior.type),
         xlab = '$v$', ylab = '$p(v|\\mathbf{y})$', breaks = 50,
         col  ="lightgray", cex.lab = 1.5, cex.main = 1.5,
         border = "white")
    lines(v1, densigamma(v1, alpha, beta), type = 'l')
    points(mle, 0, pch = 16, col = "red")  # MLE
    points(s2, 0.1, pch = 17, col = "blue")   # s^2
    points(map, 0.2, pch = 15, col = "brown")  # MAP
    legend("topright", title = "Estimator", bty = "n",
           c("MLE", expression(s^2),
             substitute(paste("MAP"[prior.type]))),
           pch = c(16, 17, 15), col = c("red", "blue", "brown"))
}
hist_v(sample1, 1, "ref", alpha1, beta1, mle_v1, s2_1, map_v1)

library(mvtnorm)
library(fields)
den_coef <- function(k, map, Sigma, mdl, prior.type) {
    m <- length(k)
    mu <- as.vector(map)
    # Omega <- s2 * chol2inv(chol(crossprod(design_mat)))
    Z = matrix(NA, nrow = m, ncol = m)
```

```r
    for (i in 1:m) {
        for (j in 1:m) {
            Z[i, j] = dmvt(c(k[i], k[j]), delta = mu, sigma = Sigma, df = n - 4,
                           log = FALSE)
        }
    }
    # Z = outer(k, k, dmvt, delta = mu, sigma = Sigma, df = n - 4, log = FALSE)
    # does not work. Why?
    par(mar = rep(.05, 4))
    persp(k, k, Z, theta = -40, phi = 30, col = "lightblue", border = "darkgray",
          box = T, axes = F)
    par(mar = rep(3, 4))
    image.plot(k, k, Z, axes = T, xlab = "", ylab = "", cex.axis = 0.8)
    contour(k, k, Z, add = TRUE, col = "white")
    title(main = paste("Student-t for",
                       ifelse(mdl == 1, "$(\\phi_1, \\phi_2)$", "(a, b)"),
                       "\n model", mdl, prior.type), cex.main = 1)
}
k <- seq(-0.2, 0.8, length = 50)
Omega <- s2_1 * chol2inv(chol(crossprod(Ft)))
par(mfrow = c(1, 2))
den_coef(k, mle1, Omega, 1, "ref")
# dev.off()
# (e) model 2: sketch marginal posterior of v and (a, b)
# sketch v
alpha2 <- (n - 2) / 2
beta2 <- (n - 2) * s2_2 / 2
sample2 <- rigamma(500, alpha2, beta2)
hist_v(sample2, 2, "ref", alpha2, beta2, mle_v2, s2_2, map_v2)
# sketch (a, b)
k2 <- seq(0.6, 2.4, length = 50)
Omega2 <- s2_2 * chol2inv(chol(crossprod(Xt)))
den_coef(k2, mle2, Omega2, 2, "ref")
# (e) model 2: sketch marginal posterior of v and (a, b)
# sketch v
alpha2 <- (n - 2) / 2
beta2 <- (n - 2) * s2_2 / 2
sample2 <- rigamma(500, alpha2, beta2)
hist_v(sample2, 2, "ref", alpha2, beta2, mle_v2, s2_2, map_v2)

# sketch (a, b)
k2 <- seq(0.6, 2.4, length = 50)
Omega2 <- s2_2 * chol2inv(chol(crossprod(Xt)))
par(mfrow = c(1, 2))
den_coef(k2, mle2, Omega2, 2, "ref")
# (f) conjugate prior case: redo (c), (d), and (e)
# model 1
# set up conjugate priors
# (phi1, phi2) ~ N(m0, vC0), v ~ IG(n0/2, d0/2)
```

```r
m01 <- c(0.2, -0.5)
C01 <- diag(3, 2)
n01 <- 10
d01 <- 20

# redo(c) posterior of (phi1, phi2) ~ T(m, vC) df = nstar
# and v ~ IG(nstar/2, dstar/2)
Q1 <- Ft %*% C01 %*% t(Ft) + diag(1, n - 2)
e1 <- y[3:n] - Ft %*% m01
AA <- C01 %*% t(Ft) %*% chol2inv(chol(Q1))
m1 <- m01 + AA %*% e1
C1 <- C01 - AA %*% Ft %*% C01

nstar1 <- (n - 2) + n01
dstar1 <- t(e1) %*% chol2inv(chol(Q1)) %*% e1 + d01

map1_conj <- m1
map_v1_conj <- (dstar1 / 2) / ((nstar1 / 2) + 1)
# redo(d) sketch marginal posterior of v and (phi1, phi2)
# sketch v
sample1conj <- rigamma(500, nstar1 / 2, dstar1 / 2)
hist_v(sample1conj, 1, "conj", nstar1 / 2, dstar1 / 2, mle_v1,
       s2_1, map_v1_conj)

# sketch (phi1, phi2)
Omega <- as.vector(dstar1 / nstar1) * C1
par(mfrow = c(1, 2))
den_coef(k, map1_conj, Omega, 1, "conj")
# model 2
# set up conjugate priors
# (a, b) ~ N(m0, vC0), v ~ IG(n0/2, d0/2)
m02 <- c(2, 3)
C02 <- diag(1, 2)
n02 <- 10
d02 <- 20

# redo(c) posterior of (phi1, phi2) ~ T(m, vC) df = nstar
# and v ~ IG(nstar/2, dstar/2)
Q2 <- Xt %*% C02 %*% t(Xt) + diag(1, n)
e2 <- (x - Xt %*% m02)
BB <- C02 %*% t(Xt) %*% chol2inv(chol(Q2))
m2 <- m02 + BB %*% e2
C2 <- C02 - BB %*% Xt %*% C02

nstar2 <- n + n02
dstar2 <- t(e2) %*% chol2inv(chol(Q2)) %*% e2 + d02

map2_conj <- m2
map_v2_conj <- (dstar2 / 2) / ((nstar2 / 2) + 1)
```

```r
# redo(e) sketch marginal posterior of v and (a, b)
# sketch v
v2 <- seq(0.1, 3, 0.01)
sample2conj <- rigamma(500, nstar2/2, dstar2/2)
hist_v(sample2conj, 2, "conj", nstar2 / 2, dstar2 / 2,
       mle_v2, s2_2, map_v2_conj)

# sketch (a, b)
Omega2 <- as.vector(dstar2 / nstar2) * C2
par(mfrow = c(1, 2))
den_coef(k2, map2_conj, Omega2, 2, "conj")
# Sensitivity analysis
# model 1
# reset conjugate priors
m01new <- c(0.9, -0.1)
C01new <- diag(2, 2)
n01new <- 50
d01new <- 100

# redo(c)
Q1new <- Ft %*% C01new %*% t(Ft) + diag(1, n-2)
e1new <- (y[3:n] - Ft %*% m01new)
AAnew <- C01new %*% t(Ft) %*% chol2inv(chol(Q1new))
m1new <- m01new + AAnew %*% e1new
C1new <- C01new - AAnew %*% Ft %*% C01new

nstar1new <- (n - 2) + n01new
dstar1new <- t(e1new) %*% chol2inv(chol(Q1new)) %*% e1new + d01new

map1_conj_new <- m1new
# compare map
# map1_conj
# map1_conj_new
map_v1_conj_new <- (dstar1new / 2) / ((nstar1new / 2) + 1)
# map_v1_conj
# map_v1_conj_new
par(mfrow = c(1, 1), mar = c(4, 4, 1, 1))
plot(v1, densigamma(v1, nstar1 / 2, dstar1 / 2), type = 'l',
     xlab = '$v$', ylab = '$p(v|\\mathbf{y})$', lwd = 2, col = 2)
lines(v1, densigamma(v1, nstar1new / 2, dstar1new / 2), lwd = 3, col = 4)
legend("topright", title = "Sensitivity of model 1: $v$",
       c("$n_0 = 10$, $d_0 = 20$", "$n_0 = 50$, $d_0 = 100$"), lwd = c(2, 3),
       cex = 0.9, bty = "n", col = c(2, 4))
Omeganew <- as.vector(dstar1new / nstar1new) * C1new
par(mfrow = c(2, 2))
den_coef(k, map1_conj, Omega, 1, "$\\mathbf{m_0} = (0.2, -0.5)$")
den_coef(k, map1_conj_new, Omega, 1, "$\\mathbf{m_0} = (0.9, -0.1)$")
# model 2
# reset conjugate priors
```

```r
m02new <- c(2, 3) # = m02
m02new1 <- c(10, 20)
C02new <- diag(1, 2)
n02new <- 50
d02new <- 100

# redo(c)
Q2new <- Xt %*% C02new %*% t(Xt) + diag(1, n)
e2new <- (x - Xt %*% m02new)
e2new1 <- (x - Xt %*% m02new1)
BBnew <- C02new %*% t(Xt) %*% chol2inv(chol(Q2new))
m2new <- m02new + BBnew %*% e2new
m2new1 <- m02new1 + BBnew %*% e2new1
C2new <- C02new - C02new %*% t(Xt) %*% chol2inv(chol(Q2new)) %*% Xt %*% C02new

nstar2new <- (n) + n02new
nstar2new1 <- (n) + n02new

dstar2new <- t(e2new) %*% chol2inv(chol(Q2new)) %*% e2new + d02new
dstar2new1 <- t(e2new1) %*% chol2inv(chol(Q2new)) %*% e2new1 + d02new

map2_conj_new1 <- m2new1
map_v2_conj_new <- (dstar2new / 2) / ((nstar2new / 2) + 1)


par(mfrow = c(1, 1), mar = c(4, 4, 1, 1))
plot(v2, densigamma(v2, nstar2 / 2, dstar2 / 2), type = 'l',
     xlab = '$v$', ylab = '$p(v|\\mathbf{y})$', lwd = 2, col = 2)
lines(v2, densigamma(v2, nstar2new/2, dstar2new/2), lwd = 3, col = 4)
legend("topright", bty = "n",
       title = "Sensitivity of model 2: $v$ and same $\\mathbf{m_0} = (2, 3)'$",
       c("$n_0 = 10$, $d_0 = 20$", "$n_0 = 50$, $d_0 = 100$"), lwd = c(2, 3),
       cex = 0.9, col = c(2, 4))
mu <- as.vector(m2new)
munew <- as.vector(m2new1)
Omeganew <- as.vector(dstar2new / nstar2new) * C2
Omeganew1 <- as.vector(dstar2new1 / nstar2new1) * C2new

library(fMultivar)
Z = matrix(dmvst(K2, 2, mu, Omega, df = nstar2), length(k2))
Znew = matrix(dmvst(K2, 2, munew, Omeganew, df = nstar2new), length(k2))
par(mfrow = c(2, 2))
den_coef(k2, m2new, Omeganew1, 2, "same $(n_0, d_0) = (50, 100)$")
den_coef(k2, m2new1, Omeganew1, 2, "same $(n_0, d_0) = (50, 100)$")
par(mfrow = c(2, 2))
den_coef(k2, m2new, Omeganew, 2, "$(n_0, d_0) = (10, 20)$")
den_coef(k2, m2new1, Omeganew1, 2, "$(n_0, d_0) = (50, 100)$")
# AMS 223 Time Series HW1 Q5 Chap 1 problem 7
# Cheng-Han Yu, Dept of Statistics UC Santa Cruz
```

```r
# Time series and Baysian inference overview
rm(list = ls())
# Metropolis-Hasting algorithm for Threshold autoregressive (TAR) model
# model(1.1) true parameter values
true_phi1 <- 0.9
true_phi2 <- -0.3
true_v <- 1
true_theta <- -1.5

# sample size
n <- 1500

# generate data from model (1.1)
y0 <- 1  # arbitrary initial value
delta <- rep(0, n)  # indicator variable (1 from M1 and 2 from M2)
y <- rep(0, n)
x <- y0
for (i in 1:n) {
    if (x > -true_theta) {
        y[i] <- true_phi1 * x + rnorm(1, 0, true_v)
        delta[i] <- 1
        x <- y[i]
    } else {
        y[i] <- true_phi2 * x + rnorm(1, 0, true_v)
        delta[i] <- 2
        x <- y[i]
    }
}
par(mfrow = c(2, 1), mar = c(4, 4, 1, 1))
plot(y, type = 'l', axes = F, xlab = 'time')
axis(1); axis(2)
plot(delta, type = 'l', axes = F, xlab = 'time')
axis(1); axis(2)
# set yt and y_{t-1} vector
yt <- y[-1]
ym1 <- y[-n]

# choose prior parameters
c <- 1
a <- 3
alpha0 <- 3
beta0 <- 0.003

# stotrage
m <- 21000
Theta <- rep(NA, m)
Phi1 <- rep(NA, m)
Phi2 <- rep(NA, m)
V <- rep(NA, m)
```

```r
# initial values
Phi1[1] <- 0.5
Phi2[1] <- -0.5
Theta[1] <- -2
V[1] <- 2

# counting variable
accept <- 0
count <- 0

Q_fcn <- function(yt, ym1, phi1, phi2, theta) {
    cond <- as.numeric((theta + ym1) > 0)
    sum((yt - ym1 * ifelse(cond, phi1, phi2)) ^ 2)
}

update_phi1 <- function(yt, ym1, phi2, theta, v) {
    cond <- as.numeric((theta + ym1) > 0)
    idx_one <- which(cond == 1)
    sig2_phi1 <- c * v / (c * sum(ym1[idx_one] ^ 2) + v)
    mu_phi1 <- sig2_phi1 * sum(yt[idx_one] * ym1[idx_one]) / v
    rnorm(1, mu_phi1, sig2_phi1)
}

update_phi2 <- function(yt, ym1, phi1, theta, v) {
    cond <- as.numeric((theta + ym1) > 0)
    idx_zero <- which(cond == 0)
    sig2_phi2 <- c * v / (c * sum(ym1[idx_zero] ^ 2) + v)
    mu_phi2 <- sig2_phi2 * sum(yt[idx_zero] * ym1[idx_zero]) / v
    rnorm(1, mu_phi2, sig2_phi2)
}

update_v <- function(yt, ym1, phi1, phi2, theta) {
    alpha_v <- (n - 1) / 2 + alpha0
    beta_v <- Q_fcn(yt, ym1, phi1, phi2, theta) / 2 + beta0
    pscl::rigamma(1, alpha_v, beta_v)
}

lpost_theta <- function(yt, ym1, phi1, phi2, theta, v){
    -0.5 * Q_fcn(yt, ym1, phi1, phi2, theta) / v
}

library(pscl)

# phi2 <- Phi2[1]
# theta <- Theta[1]
# v <- V[1]
# for (i in 2:m) {
#     cat("iter:", i, "\r")
#
```

```r
#     # sample phi1
#     phi1 <- update_phi1(yt, ym1, phi2, theta, v)
#
#     # sample phi2
#     phi2 <- update_phi2(yt, ym1, phi1, theta, v)
#
#     # sample v
#     v <- update_v(yt, ym1, phi1, phi2, theta)
#
#     # use random walk proposal newtheta = theta + N(0, 1) to update theta
#     new.theta <- Theta[i - 1] + rnorm(1, 0, .02)
#     if (new.theta < -a || new.theta > a) {
#         theta <- Theta[i - 1]
#     } else {
#         count <- count + 1
#         u <- runif(1)
#         if (log(u) < (lpost_theta(yt, ym1, phi1, phi2, new.theta, v)
#                      - lpost_theta(yt, ym1, phi1, phi2, Theta[i - 1], v))) {
#             theta <- new.theta
#             accept <- accept + 1
#         }
#     }
#
#     # store results
#     Phi1[i] <- phi1
#     Phi2[i] <- phi2
#     Theta[i] <- theta
#     V[i] <- v
# }
# draws <- MCMCalgo(yt, ym1, m = 11000)
# burnning and thining
# burn <- 1000
# thin <- 10
# sampleidx = seq(from = (burn + thin), to = m, by = thin)

# trace plots, ACF and histograms
# par(mfrow = c(4, 3), mar = c(4, 4, 4, 1))
# plot(draws[, "Theta"][sampleidx], type = 'l', ylab = "",
#      main = expression(paste("Trace of ", theta)))
# # abline(h = true_theta, col = 2, lwd = 2)
# acf(draws[, "Theta"][sampleidx], main = expression(paste("ACF of ", theta)))
# hist(draws[,"Theta"][sampleidx], freq = F, breaks = 30, main = "",
#      xlab = expression(theta), col = "navy", border = FALSE)

# main M-H alogorithm
MCMCalgo <- function(yt, ym1, init_phi2, init_theta, init_v, m) {
    # stotrage
    Theta <- rep(NA, m)
    Phi1 <- rep(NA, m)
```

```r
    Phi2 <- rep(NA, m)
    V <- rep(NA, m)

    # counting variable
    accept <- 0
    count <- 0

    phi2 <- init_phi2
    theta <- init_theta
    v <- init_v

    for (i in 1:m) {
        # cat("iter:", i, "\r")

        # sample phi1
        phi1 <- update_phi1(yt, ym1, phi2, theta, v)

        # sample phi2
        phi2 <- update_phi2(yt, ym1, phi1, theta, v)

        # sample v
        v <- update_v(yt, ym1, phi1, phi2, theta)

        # use random walk proposal newtheta = theta + N(0, 1) to update theta
        new.theta <- theta + rnorm(1, 0, .025)
        if (new.theta < -a || new.theta > a) {
            theta <- theta
        } else {
            count <- count + 1
            u <- runif(1)
            if (log(u) < (lpost_theta(yt, ym1, phi1, phi2, new.theta, v)
                          - lpost_theta(yt, ym1, phi1, phi2, theta, v))) {
                theta <- new.theta
                accept <- accept + 1
            }
        }

        # store results
        Phi1[i] <- phi1
        Phi2[i] <- phi2
        Theta[i] <- theta
        V[i] <- v
    }
    return(list(Phi1 = Phi1, Phi2 = Phi2, V = V, Theta = Theta,
                accept = accept, count = count))
}

mcmc_sample <- MCMCalgo(yt, ym1, init_phi2 = rnorm(1, 0, 0.25),
                  init_theta = runif(1, -3, 3), init_v = rigamma(1, 3, 1),
```

```r
                  m = 21000)
# burnning and thining
burn <- 1000
thin <- 10
sampleidx = seq(from = (burn + thin), to = m, by = thin)

post.sample <- function(data, sampleidx) {
    Phi1 <- data$Phi1[sampleidx]
    Phi2 <- data$Phi2[sampleidx]
    V <- data$V[sampleidx]
    Theta <- data$Theta[sampleidx]
    draws <- cbind(Phi1, Phi2, V, Theta)
    colnames(draws) <- c("phi_1", "phi_2", "v", "theta")
    return(draws)
}

draws <- post.sample(mcmc_sample, sampleidx)
par(mfrow = c(4, 3), mar = c(4, 4, 4, 1))
plot(draws[, "phi_1"], type = 'l', ylab = "",
     main = expression(paste("Trace of ", phi[1])))
# abline(h = true_phi1, col = 2, lwd = 2)
acf(draws[, "phi_1"], main = expression(paste("ACF of ", phi[1])))
hist(draws[, "phi_1"], freq = F, breaks = 30, main = "",
     xlab = expression(phi[1]), col = "navy", border = FALSE)

plot(draws[, "phi_2"], type = 'l', ylab = "",
     main = expression(paste("Trace of ", phi[2])))
# abline(h = true_phi2, col = 2, lwd = 2)
acf(draws[, "phi_2"], main = expression(paste("ACF of ", phi[2])))
hist(draws[, "phi_2"], freq = F, breaks = 30, main = "",
     xlab = expression(phi[2]), col = "navy", border = FALSE)

plot(draws[, "v"], type = 'l', ylab = "",
     main = expression(paste("Trace of ", v)))
# abline(h = true_v, col = 2, lwd = 2)
acf(draws[, "v"], main = expression(paste("ACF of ", v)))
hist(draws[, "v"], freq = F, breaks = 30, main = "",
     xlab = expression(v), col = "navy", border = FALSE)

plot(draws[, "theta"], type = 'l', ylab = "",
     main = expression(paste("Trace of ", theta)))
# abline(h = true_theta, col = 2, lwd = 2)
acf(draws[, "theta"], main = expression(paste("ACF of ", theta)))
hist(draws[, "theta"], freq = F, breaks = 30, main = "",
     xlab = expression(theta), col = "navy", border = FALSE)
library(coda)
library(parallel)
library(doParallel)
detectCores()
```

```
cl <- makeCluster(2, type = "FORK")
registerDoParallel(cl)
getDoParWorkers()

MCMCalgo.mc <- function(s) {
    # initial values
    init_phi2 <- rnorm(5, 0, 0.25)
    init_theta <- runif(5, -3, 3)
    init_v <- rigamma(5, 3, 1)
    MCMCalgo(yt, ym1, init_phi2[s], init_theta[s], init_v[s], m = 21000)
}

system.time(draws_mc <- mclapply(1:5, MCMCalgo.mc, mc.cores = 2))

stopCluster(cl)
# Analysis using coda package
draws.mc = lapply(draws_mc, post.sample, sampleidx = sampleidx)
coda.draws.mc = lapply(draws.mc, mcmc)
# mean, sd, and quantiles of the two chains
lapply(coda.draws.mc, summary)
# traceplots, and densities
lapply(coda.draws.mc, plot)
# pairwise correlations
lapply(coda.draws.mc, function(x) pairs(data.frame(x)))

# Convergence diagnostics
combinedchains = mcmc.list(coda.draws.mc[[1]], coda.draws.mc[[2]],
                           coda.draws.mc[[3]], coda.draws.mc[[4]],
                           coda.draws.mc[[5]])
plot(combinedchains)
# acf
autocorr.diag(combinedchains)
autocorr.plot(combinedchains)
# crosscorr
crosscorr.plot(combinedchains)
# Gelman and Rubin potential scale reduction factor
gelman.diag(combinedchains)  # should be close to 1
gelman.plot(combinedchains)
# GewekeâĂŹs convergence diagnostic
geweke.diag(combinedchains)  # Z-scores
geweke.plot(combinedchains)
# Heidelberger and WelchâĂŹs convergence diagnostic
heidel.diag(combinedchains)

# Raftery and LewisâĂŹs diagnostic
raftery.diag(combinedchains)
# AMS 223 Time Series HW1 Q6
# Cheng-Han Yu, Dept of Statistics UC Santa Cruz
# Time series and Baysian inference overview
```

```r
# (a) plot the data
# EEG at channel F3
eegf3 <- read.table("http://users.soe.ucsc.edu/~raquel/tsbook/data/eegF3.dat")
eegf3 <- as.vector(t(eegf3))
# head(eegf3)  # check data
plot(eegf3, type = "l", axes = F, xlab = "time", ylab = " ")
axis(1)
axis(2)
# GDP
gdp <- read.table("http://users.soe.ucsc.edu/~raquel/tsbook/data/gdp.dat",
                  header = TRUE, skip = 2)
# head(gdp)  # check data
par(mfrow = c(3, 3))
for (i in 2: ncol(gdp)) {
  plot(gdp[, 1], gdp[, i], type = "l", xlab = "year", ylab = " ", axes = F,
       main = colnames(gdp)[i], cex.axis = 0.5)
  axis(1)
  axis(2)
}
dev.off()
# Southern Oscillation Index (SOI)
soi <- read.table("http://users.soe.ucsc.edu/~raquel/tsbook/data/soi.dat")
# head(soi)  # check data
ts.plot(soi, gpars = list(xlab = "time", ylab = " ", axes = F))
axis(1)
axis(2)
# (b) ACF, smoothing SOI, differencing GDP
# (1) ACF
acf(eegf3)
par(mfrow = c(3,3))
for (i in 2: ncol(gdp)) {
  acf(gdp[, i], main = colnames(gdp)[i])
}
acf(soi)
# (2) smoothing SOI
# moving avg of order 5 with equal weights
soi.ma5 <- filter(soi, filter = c(.2, .2, .2, .2, .2), side = 2)
# moving avg of order 11 with equal weights
soi.ma11 <- filter(soi, filter = rep(1, 11)/11, side = 2)
par(mfrow = c(2, 1))
ts.plot(soi.ma5,
        gpars = list(xlab = "time", ylab = " ", axes = F, cex.main = 0.95),
        main = "Smoothing Series of SOI: MA order 5 with equal weights")
axis(1)
axis(2)
ts.plot(soi.ma11,
        gpars = list(xlab = "time", ylab = " ", axes = F, cex.main = 0.95),
        main = "Smoothing Series of SOI: MA order 11 with equal weights")
axis(1)
```

```r
axis(2)
# moving avg of order 5 with unequal weights (0.1, 0.15, 0.2, 0.25, 0.3)
soi.ma5un1 <- filter(soi, filter = c(0.1, 0.15, 0.2, 0.25, 0.3), side = 2)
# moving avg of order 5 with unequal weights (0.3, 0.25, 0.2, 0.15, 0.1)
soi.ma5un2 <- filter(soi, filter = c(0.3, 0.25, 0.2, 0.15, 0.1), side = 2)
par(mfrow = c(2, 1))
ts.plot(soi.ma5un1, gpars = list(xlab = "time", ylab = " ", axes = F,
                                cex.main = 0.9),
        main = "Smoothing SOI: MA order 5 with unequal weights
        (0.1, 0.15, 0.2, 0.25, 0.3)")
axis(1)
axis(2)
ts.plot(soi.ma5un2, gpars = list(xlab = "time", ylab = " ", axes = F,
                                cex.main = 0.9),
        main = "Smoothing SOI: MA order 5 with unequal weights
        (0.3, 0.25, 0.2, 0.15, 0.1)")
axis(1)
axis(2)
# (3) differencing GDP ACF
gdp.d1 <- matrix(NA, nrow = nrow(gdp) - 1, ncol = ncol(gdp) - 1)
for (i in 2: ncol(gdp)) {
  gdp.d1[, i - 1] <- diff(gdp[, i])
}
gdp.d1 <- data.frame(cbind(gdp[(2:nrow(gdp)), 1], gdp.d1))
colnames(gdp.d1) = paste("1st dif", colnames(gdp))

par(mfrow = c(3, 3))
for (i in 2: ncol(gdp)) {
  plot(gdp.d1[, 1], gdp.d1[, i], type = "l", xlab = "year", ylab = " ", axes = F,
       main = colnames(gdp)[i])
  axis(1)
  axis(2)
}
par(mfrow = c(3,3))
for (i in 2: ncol(gdp)) {
  acf(gdp.d1[, i], main = paste("ACF of 1st diff of", colnames(gdp)[i]),
      cex.main = 0.85)
}
gdp.d2 <- matrix(NA, nrow = nrow(gdp) - 2, ncol = ncol(gdp) - 1)
for (i in 2: ncol(gdp)) {
  gdp.d2[, i - 1] <- diff(gdp[, i], difference = 2)
}
gdp.d2 <- data.frame(cbind(gdp[(3: nrow(gdp)), 1], gdp.d2))
colnames(gdp.d2) = paste("2nd dif", colnames(gdp))
par(mfrow = c(3, 3))
for (i in 2: ncol(gdp)) {
  plot(gdp.d2[, 1], gdp.d2[, i], type = "l", xlab = "year", ylab = " ", axes = F,
       main = colnames(gdp)[i])
  axis(1)
```

```
  axis(2)
}
par(mfrow = c(3,3))
for (i in 2: ncol(gdp)) {
  acf(gdp.d2[, i], main = paste("ACF of 2nd diff of", colnames(gdp)[i]),
      cex.main = 0.85)
}
```

References

Gandrud, C. (2016). *repmis: Miscellaneous Tools for Reproducible Research*. R package version 0.5.

Nychka, D., Furrer, R., Paige, J., and Sain, S. (2016). *fields: Tools for Spatial Data*. R package version 8.4-1.

Plummer, M., Best, N., Cowles, K., Vines, K., Sarkar, D., Bates, D., Almond, R., and Magnusson, A. (2015). *coda: Output Analysis and Diagnostics for MCMC*. R package version 0.18-1.

Xie, Y. (2016). *knitr: A General-Purpose Package for Dynamic Report Generation in R*. R package version 1.14.