

Bayesian Sampling Environment & Language

**BASEL** package

PC Version 1.0

Project manager: Prof. Wolfgang Polasek

Contributors: Lei Ren, Momtchil Pojarliev, Carolyn Holliger

Institute of Statistics and Econometrics  
University of Basel  
Holbeinstrasse 12  
CH-4051 Basel  
Switzerland

BASEL package ©ISO-WWZ University of Basel. All rights reserved.  
This work was funded in part by the Swiss National Fund.

February 8, 2000

# Contents

<b>1</b>	<b>Introduction</b>	<b>4</b>
1.1	Hardware and software . . . . .	4
1.2	Installation . . . . .	4
1.2.1	Distribution . . . . .	4
1.2.2	How to install . . . . .	4
1.3	About this manual . . . . .	5
<b>2</b>	<b>Markov chain Monte Carlo Methods</b>	<b>6</b>
2.1	The Gibbs sampling algorithm . . . . .	6
2.2	The Metropolis algorithm . . . . .	7
2.3	Diagnostics for MCMC algorithms . . . . .	8
<b>3</b>	<b>Univariate Models</b>	<b>9</b>
3.1	Univariate models with fractional prior . . . . .	9
3.1.1	The marginal likelihood of a regression model with fractional prior . . . . .	9
3.1.2	The fractional marginal likelihood for the regression model with outliers . . . . .	11
3.1.3	The marginal likelihood of an ARX regression model with a fractional prior . . . . .	12
3.1.4	The fractional marginal likelihood for the ARX(p) model with outliers . . . . .	14
3.2	Univariate models with informative prior . . . . .	15
3.2.1	The marginal likelihood of a regression model with an informative prior . . . . .	15
3.2.2	The marginal likelihood of an AR regression model with an informative prior . . . . .	17
3.3	The Bayesian AR(p) Model . . . . .	19

3.4	ARCH models . . . . .	22
3.4.1	The full conditional distributions (f.c.d.) . . . . .	23
<b>4</b>	<b>Multivariate models</b>	<b>30</b>
4.1	Multivariate models with fractional prior . . . . .	30
4.1.1	The marginal likelihood of a multivariate regression model with fractional prior	30
4.1.2	Multivariate regression model with outliers . . . . .	32
4.1.3	The marginal likelihood of a VAR regression model with a fractional prior . . .	33
4.1.4	The fractional marginal likelihood for the VARX(p) model with outliers . . . .	35
4.2	Multivariate models with informative prior . . . . .	36
4.2.1	Marginal likelihood of a multivariate regression model with an informative prior	36
4.2.2	The marginal likelihood of a VARX regression model with an informative prior	38
4.3	The VAR( $k$ )-GARCH( $p,q$ )-X( $s$ ) models . . . . .	39
4.3.1	The full conditional distributions (f.c.d.) . . . . .	40
4.3.2	Program description . . . . .	42
4.4	The VAR( $k$ )-GARCH( $p,q$ )-M( $r$ ) models . . . . .	45
4.4.1	The full conditional distributions (f.c.d.) . . . . .	46
4.4.2	Program description . . . . .	49
<b>5</b>	<b>Miscellaneous Models</b>	<b>52</b>
5.1	Unit root test with marginal likelihood . . . . .	52
5.1.1	The AR( $p$ ) model in first differences (non-stationary model) . . . . .	52
5.1.2	The simple ADF model (stationary model) . . . . .	53
5.1.3	The ADF model with mean (stationary model) . . . . .	53
5.1.4	The ADF model with mean and trend (stationary model) . . . . .	53
5.1.5	Marginal likelihoods for the AR(p) model with fractional prior . . . . .	53
5.1.6	Module usage . . . . .	54
<b>6</b>	<b>Appendix</b>	<b>56</b>
6.0.7	References . . . . .	56

# Chapter 1

## Introduction

### 1.1 Hardware and software

This package is written for S-PLUS. Since the compiled Fortran code runs faster than interpreted S-PLUS code, all the model functions are written in C or in Fortran 77. All the compiled codes are well written, tested and linked to S-PLUS.

The current PC version 1.0 of the BASEL package can be run on any PC with S-PLUS 4.5 and Windows 95, 98 or NT. You must have **WinZip** software on your PC.

### 1.2 Installation

#### 1.2.1 Distribution

The BASEL package can be downloaded from the Internet. Go to <http://www.unibas.ch/iso>, select 'Programs' and then 'Basel' in the menu on the left-hand side of the screen. There are two files to download: the package itself and the user's manual, which you are currently reading. You can also ask for a set of the BASEL package disks instead. In this case, you will be charged for the cost of diskettes and delivery.

#### 1.2.2 How to install

First make a directory 'Basel.PC' on your C: disk with the subdirectory 'object' and then extract all files from Basel.PC.zip to this subdirectory. You'll need to download the program 'winzip' if you don't have it.

Start the program S-Plus and open the file 'baselpackage' from subdirectory 'object'.

Run the program 'baselpackage' and all functions and default data will then be available for use.

Read the User's Guide (ps file) and learn step by step how to use the functions. If there are any installation problems, please contact Lei Ren (e-mail:[lei@iso.iso.unibas.ch](mailto:lei@iso.iso.unibas.ch)).

### 1.3 About this manual

In Chapter 2 we briefly introduce Markov chain Monte Carlo (MCMC) methods including the Gibbs sampler and the Metropolis algorithm. We also discuss three convergence diagnostics used to check the convergence of Monte Carlo Markov chains.

In chapters 3 and 4 we illustrate the functions with examples.

Bernardo and Smith (1994) contains an extensive list of textbooks or monographs on Bayesian inference. Zellner (1971) and Poirier (1995) give excellent introductions to Bayesian inference in econometrics in two different manners. Polasek (1996) mainly compiles updated material of Bayesian studies on several time series models, and especially provides a solid background for the present manual. Hamilton (1994) is a recent volume on time series analysis. Cowles and Carlin (1994) provides a survey on convergence diagnostics. There are plenty of books on S-Plus including the S-Plus documentation (1995). A list of some useful Internet addresses is as follows:

<a href="http://lib.stat.cmu.edu">http://lib.stat.cmu.edu</a>	large free library of statistical software
<a href="http://www.mathsoft.com/splus/">http://www.mathsoft.com/splus/</a>	manufacturer of SPlus
<a href="mailto:s-news@wubios.wustl.edu">s-news@wubios.wustl.edu</a>	SPlus discussion list
<a href="http://www.stat.mat.ethz.ch/S-FAQ">http://www.stat.mat.ethz.ch/S-FAQ</a>	frequently asked questions of that list

Most of the manual is based on contributions and efforts made by all members of the group: Lei Ren, Momtchil Pojarliev, Carolyn Holliger, led by Professor Wolfgang Polasek. It also benefits from the assistance of several colleagues at the ISO, WWZ, University of Basel in one way or another. Their valuable help is gratefully acknowledged.

## Chapter 2

# Markov chain Monte Carlo Methods

In this chapter, we briefly describe Markov chain Monte Carlo (MCMC) methods which have recently become very popular in statistical analysis, especially in Bayesian inference. The idea is to simulate a Markov chain whose stationary distribution is the distribution of interest (usually a posterior distribution in Bayesian inference).

There are two main methods for generating a Markov chain. One of these is the Gibbs sampler (Geman and Geman, 1984), which was popularized by Gelfand and Smith (1990). The other method is the Metropolis algorithm (Metropolis et al., 1953). Hastings (1970) provides a generalization of the Metropolis algorithm. For introductions or reviews of these theories and their applications, see e.g., Casella and George (1992), Besag and Green (1993), Tierney (1994) and Chib and Greenberg (1995).

### 2.1 The Gibbs sampling algorithm

Suppose we wish to sample from the joint distribution  $\pi(\theta_1, \dots, \theta_k)$  or from marginal distributions such as  $\pi(\theta_i)$  of a random vector  $\theta = (\theta_1, \dots, \theta_k)$ , where  $\theta_i$  might be a vector or scalar. The Gibbs sampler enables us to draw random samples from this joint distribution in situations where  $\pi(\theta_1, \dots, \theta_k)$  is unknown or not available, but where the conditional distributions

$$p(\theta_i | \{\theta_j, j \neq i\})$$

are available for sampling for all  $i = 1, \dots, k$ .

Gibbs sampling consists of the following iterative updating scheme:

- Step 0     Choose a starting value  $\theta^{(0)} = (\theta_1^{(0)}, \dots, \theta_k^{(0)})$ .  
 For  $i = 1, 2, \dots$   
 Step 1     Draw  $\theta_1^{(i)}$  from  $p(\theta_1 | \theta_2^{(i-1)}, \dots, \theta_k^{(i-1)})$ .  
 Step 2     Draw  $\theta_2^{(i)}$  from  $p(\theta_2 | \theta_1^{(i)}, \theta_3^{(i-1)}, \dots, \theta_k^{(i-1)})$ .  
            $\vdots$   
 Step  $k$     Draw  $\theta_k^{(i)}$  from  $p(\theta_k | \theta_1^{(i)}, \dots, \theta_{k-1}^{(i)})$ .

It can be shown that under mild conditions the Markov chain converges, i.e.,

$$\theta^{(i)} \rightarrow \theta$$

in distribution as  $i \rightarrow \infty$ .

## 2.2 The Metropolis algorithm

The Metropolis algorithm shares many of the characteristics of the Gibbs sampler, but is more generally applicable, as it allows us to sample from difficult distributions.

Again, suppose we are interested in sampling from the joint distribution  $p(\theta)$  of a random vector  $\theta$ . The Metropolis algorithm begins with the choices of the proposal distribution  $p(\theta^* | \theta^{(i)})$  and a starting parameter value  $\theta^{(0)}$ .

The Metropolis algorithm consists of the following iterative update scheme:

- Step 0     Choose a starting value  $\theta^0$  and the proposal distribution  $p(\theta^* | \theta^{(i)})$   
 For  $i = 1, 2, \dots$   
 Step 1     Draw  $\theta^*$  from  $p(\theta^* | \theta^{(i-1)})$ .  
 Step 2     Calculate  $\alpha = \min\{1, \frac{p(\theta^*)}{p(\theta^{(i-1)})}\}$ .  
 Step 3     Accept  $\theta^*$  as  $\theta^{(i)}$  with a probability  $\alpha$ . Otherwise set  $\theta^{(i)} = \theta^{(i-1)}$ .

The candidates of the proposal distribution include a normal distribution or a t-distribution. If the proposal distribution is not symmetric, the acceptance probability  $\alpha$  is replaced by

$$\alpha = \min \left\{ 1, \frac{p(\theta^*)p(\theta^{(i-1)} | \theta^*)}{p(\theta^{(i-1)})p(\theta^* | \theta^{(i-1)})} \right\}.$$

This generalization is called the Hastings algorithm (Hastings, 1970).

In some applications, it is difficult or impossible to sample from some conditional distributions  $p(\theta_i | \{\theta_j, j \neq i\})$  encountered in the Gibbs sampler. However, the Metropolis (Hastings) algorithm can be included to update  $\theta_i$  in the Gibbs sampler (see Tierney, 1994). This is called the Metropolis (or Hastings) within Gibbs algorithm.

## 2.3 Diagnostics for MCMC algorithms

As the theory shows, a Markov chain generated from a MCMC algorithm converges to the distribution of interest as iterations go toward infinity. However, in practice, we have to stop a chain at a certain iteration where the chain is considered close to the posterior distribution.

There are several criteria for checking the convergence of MCMC algorithms (see, for example, Cowles and Carlin (1994), and Brooks and Roberts (1995)). We provide three convergence diagnostics proposed by Geweke (1992), Raftoy and Lewis (1992), and Gelman and Rubin (1992).

Many researchers prefer to look at the Markov chains proper. Examination of the autocorrelation function (acf) of each Markov chain serves the same purpose. An option of the function `CF` computes cross-correlations between different Markov chains as well. High correlations indicate a poor variability in sampled values.

# Chapter 3

## Univariate Models

### 3.1 Univariate models with fractional prior

#### 3.1.1 The marginal likelihood of a regression model with fractional prior

##### Model definition

We consider the linear model

$$y_t = \beta_0 + \beta_1 x_t^1 + \dots + \beta_{Mx} x_t^{Mx} + u_t \quad (3.1)$$

with a  $T \times (1 + Mx)$  regression matrix  $X$  of full rank. Define  $X' = (x_1, \dots, x_T)$ ,  $x_i = (1, x_i^1, \dots, x_i^{Mx})$ ,  $y = (y_1, \dots, y_T)$  and  $\beta = (\beta_0, \dots, \beta_{Mx})$  then the normal linear regressor model is giv

**Data description**

We use monthly returns of the MSCI indices. The S-Plus object *MSCI* is a matrix which contains the monthly returns of the MSCI indices for North America, France, Germany, Switzerland, UK, Spain and the Pacific in USD from February 1990 until September 1999. *MSCI.W* is a vector with the monthly returns of the MSCI world index in USD for the same time horizon. The vector *cpi.re* contains monthly returns of the real consumer expenditure in the USA for the same period.

**Module usage****Function Name:**

`mlfra`

**Usage:**

`mlfra(dataY,Mx,dataX,b)`

**Required Arguments:**

**dataY:** a univariate time series or vector of length  $T$ .

**Mx:** the number of columns of the dataX matrix.

**dataX:** a  $T \times (1 + Mx)$  matrix of time series or a vector.

**b:** the fractional parameter (default  $b = 1/T$ ).

**Example:**

```
>ml.model<-mlfra(MSCI[,1],Mx=1,MSCI.W).
```

According to the capital asset pricing model (CAPM) the returns on an asset should equal the risk-free rate of return plus a risk premium, given by returns on the market portfolio. In order to compute a given asset's *beta*, a measure of the asset's sensitivity to market fluctuations, we regress the asset's returns on the "market return", which is here approximated by the MSCI world index. To estimate this *beta*, we simply fit the following regression with the fractional prior distribution:

$$y_t = \beta_0 + \beta_1 x_t^1 + u_t. \quad (3.7)$$

The dependent variable is in each case a vector of the returns on the individual country index and the independent variable is a vector of the returns on the world index, or market portfolio. The results can then be stored as an object *ml.model*. The function *names* gives the components of the object *ml.model*, which are *beta*, *sigma* and *log.ML*. The first component of *beta* is the intercept, or  $\beta_0$ , and the second is the the slope coefficient,  $\beta_1$ .

The component *log.ML* gives the marginal likelihoods of the regression model. *Sigma* is the residual standard error. In our case we have a slope of 0.738, which is the estimated beta according to the CAPM. If the whole market changes by a factor of 1, the MSCI North America index will change by a factor of 0.738.

Model output of the function *mlfra* is given as:

\$beta: [1] 0.00657064 0.73874502, which is the OLS estimate of (3.5).

\$sigma: [1] 0.0234245, which is the ML estimate of the standard deviation (the square root of (3.6)).

\$log.ML: [1] 263.557.

The fractional likelihood in (3.4) is given by the following fractional parameter:  $b = [1]1/116$ .

### 3.1.2 The fractional marginal likelihood for the regression model with outliers

#### Model definition

In this section we consider the univariate regression model with outliers. Let  $y$  be a  $T \times 1$  vector of the dependent variable. We consider  $T$  outlier models, indexed by  $j = 1, \dots, T$ , i.e.

$$y = X_j \beta_j + D_j \theta + \varepsilon, \quad j = 1, \dots, T, \quad (3.8)$$

where  $D_j$  is a dummy variable, i.e. the  $j$ -th unity vector of length  $T$ ,  $y = (y_1, \dots, y_T)$ ,  $X_j' = (x_{1j}, \dots, x_{Tj})$ ,  $x_{ij} = (1, x_{ij}^1, \dots, x_{ij}^{Mx})$ ,  $y = (y_1, \dots, y_T)$ ,  $\beta_j = (\beta_0^j, \dots, \beta_{Mx}^j)$  and  $\theta$  is the regression coefficient which measures the size of the outlier effect at observation  $j$  of the regression model.

The univariate outlier model can be written in compact form as

$$y = \tilde{X}_j \tilde{\beta}_j + \varepsilon \quad (3.9)$$

with  $\tilde{X}_j = (X : D_j)$  and  $\tilde{\beta}_j = (\beta_j : \theta)$ . If no confusion is possible, we will drop the index  $j$ . The ordinary least squares or OLS estimate of model (3.9) is given by

$$\tilde{\beta}_j = (\tilde{X}_j' \tilde{X}_j)^{-1} \tilde{X}_j' y. \quad (3.10)$$

Assuming a fractional prior distribution, the fractional marginal likelihood is given by

$$f_b^1(y|j) = b^{\frac{Tb-1}{2}} (\pi ESS_j)^{-T(b-1)/2} \Gamma\left(\frac{T-1}{2}\right) / \Gamma\left(\frac{Tb-1}{2}\right). \quad (3.11)$$

#### Proof:

O'Hagan (1995) considers for the regression model the non-informative class of priors  $p(\beta, \sigma^2) \propto \sigma^{2t}$  where  $t$  can be any integer. In this manual we will use  $t = 1$  for all fractional priors. For details see also Polasek and Ren (1997).

#### Module usage

##### Function Name:

`mloutfra`

##### Usage:

`mloutfra(dataY, Mx, dataX, year, month, b)`

**Required Arguments:**

- dataY:** a univariate time series or a vector.
- Mx:** the number of columns of the X matrix.
- dataX:** a matrix of time series or a vector.
- year:** indicates the begin of the time horizon. (Needed for graph)
- month:** indicates the begin of the time horizon.(Needed for graph)
- b:** fractional parameter  $b = 1/T$ .

**Example:**

```
>mloutfra(MSCI[,1],Mx=1,MSCI.W,year=1990, month=1).
```

The command above will plot the dependent time series and the marginal likelihood. Reference: see Polasek and Ren (1997)

### 3.1.3 The marginal likelihood of an ARX regression model with a fractional prior

**Model definition**

We consider the linear autoregressive model

$$y_t = \beta_0 + \beta_1^1 x_{t-1}^1 + \dots + \beta_p^1 x_{t-p}^1 + \dots + \beta_1^{M_x} x_{t-1}^{M_x} + \dots + \beta_p^{M_x} x_{t-p}^{M_x} + u_t \quad (3.12)$$

with a  $T \times (1 + pM_x)$  regression matrix  $\mathbf{X}$  of full rank, and  $k = pM_x + 1$

$$y \sim N[X\beta, \sigma^2 I_T]. \quad (3.13)$$

The fractional marginal likelihood with  $b \in (0, 1)$  is given by

$$f_b^1(y) = b^{\frac{Tb-k}{2}} (\pi ESS)^{-(T-Tb)/2} \Gamma\left(\frac{T-k}{2}\right) / \Gamma\left(\frac{Tb-k}{2}\right). \quad (3.14)$$

If  $b = \frac{k}{T}$ , then  $\frac{Tb-k}{2} = 0$  and

$$f_b^1(y) = (\pi ESS)^{-\frac{T-k}{2}} \Gamma\left(\frac{T-k}{2}\right), \quad (3.15)$$

where

$$ESS = (y - X\hat{\beta})'(y - X\hat{\beta}), \quad \hat{\beta} = (X'X)^{-1}X'y, \quad \hat{\sigma}^2 = \frac{ESS}{T-p}. \quad (3.16)$$

**Module usage****Function Name:**

`marfra`

**Usage:**

```
marfra(dataY,Mx,dataX,pmax,b)
```

**Required Arguments:**

**dataY:** a univariate time series or a vector.  
**Mx:** the number of columns of the X matrix.  
**dataX:** a matrix of time series or a vector.  
**pmax:** the maximum order of the AR process. Default: 5  
**b:** the fractional parameter  $b = k/T$ . Default: 5

**Example:**

```
>ar.models<-marfra(MSCI[,1],Mx=1,MSCI.W,pmax=5).
```

In this example we regress the returns of the MSCI North America index on the lagged returns of the MSCI world index in USD. You can fit  $p$  ARX models and store them as an object `ar.model` with the function above. The function `names` gives the components of the object `ar.models`, which are `beta`, `sigma` and `log.ML`. `Beta` is a matrix with the number of columns equal to the fitted ARX models, in this case five. The first column contains the coefficient from the ARX(1) model and the last the coefficient from the ARX(5) model. The component `log.ML` is a vector of the marginal likelihoods for each model. The highest element of this vector gives us the order of the optimal AR process. In this case, element four is the highest, so we choose the ARX(4) model.

```
$beta:
      [,1]      [,2]      [,3]      [,4]      [,5]
[1,] 0.0121670 0.0123906 0.0140354 0.0146094 0.0156386
[2,] -0.0561858 -0.0573544 -0.0886978 -0.0934321 -0.0909584
[3,] 0.0000000 -0.0351914 -0.0592342 -0.0505501 -0.0628578
[4,] 0.0000000 0.0000000 -0.1082164 -0.1021519 -0.1213351
[5,] 0.0000000 0.0000000 0.0000000 -0.1694741 -0.1837649
[6,] 0.0000000 0.0000000 0.0000000 0.0000000 -0.0577228

$sigma: [1] 0.0378609 0.0379989 0.0376583 0.0367383 0.0367350
(this is the standard deviation given in 3.16)

$log.ML: [1] 208.827 207.422 207.463 209.245 208.304
```

```
The fractional parameter b: [1]1/116
```

### 3.1.4 The fractional marginal likelihood for the ARX(p) model with outliers

**Model definition**

In this section we consider the univariate autoregressive model with outliers. Let  $y$  be a  $T \times 1$  vector of the dependent variable and, in the case of an AR process of order  $p$  the first column of  $X$  is a vector of ones and the other columns are  $p$  lags such that  $k = p + 1$ . The ARX(p) option specifies a

$T \times (k + p)$  matrix of the full rank of independent variables and the AR terms. We consider  $T$  outlier models, indexed by  $j = 1, \dots, T$ , i.e.

$$y = X_j \beta_j + D_j \theta + \varepsilon, \quad j = 1, \dots, T, \quad (3.17)$$

where  $D_j$  is a dummy variable, i.e. the  $j$ -th unity vector of length  $T$ .  $\theta$  is the regression coefficient which measures the size of the outlier effect at observation  $j$  on the regression model.

The univariate outlier model can be written in compact form as

$$y = \tilde{X}_j \tilde{\beta}_j + \varepsilon \quad (3.18)$$

with  $\tilde{X}_j = (X : D_j)$  and  $\tilde{\beta}_j = (\beta_j : \theta)$ . If no confusion is possible, we will drop the index  $j$ . The ordinary least squares (OLS) estimate of model (3.18) is given by

$$\tilde{\beta}_j = (\tilde{X}_j' \tilde{X}_j)^{-1} \tilde{X}_j' y. \quad (3.19)$$

Assuming a fractional prior distribution, the fractional marginal likelihood is given by

$$f_b^1(y|j) = b^{\frac{Tb-k}{2}} (\pi ESS_j)^{-\frac{T(b-1)}{2}} \Gamma\left(\frac{T-k}{2}\right) / \Gamma\left(\frac{Tb-k}{2}\right). \quad (3.20)$$

**Proof:**

O'Hagan (1995) considers for the regression model the non-informative class of priors  $p(\beta, \sigma^2) \propto \sigma^{2t}$  where  $t$  can be any integer. In this paper we will use  $t = 1$  for all fractional priors.

Reference: see Polasek and Ren (1997).

**Module usage**

**Function Name:**

`maroutfra`

**Usage:**

`maroutfra(dataY, Mx, dataX, pmax, year, month)`

**Required Arguments:**

`dataY`: a matrix of time series or a vector (y in 3.16).

`Mx`: the number of columns of the X matrix.

`dataX`: a matrix of time series or a vector.

`pmax`: the maximum order of the AR process. Default: 2

`years`: the begin of the time horizon. Default: 1990

`month`: the begin of the time horizon. Default: 1

**Example:**

`maroutfra(MSCI[, 1], Mx=1, MSCI.W, pmax=2, year=1990, month=1)`. The command above will plot the monthly returns of the MSCI North America index and the marginal likelihoods for AR(1) and AR(2)

## 3.2 Univariate models with informative prior

### 3.2.1 The marginal likelihood of a regression model with an informative prior

#### Model definition

We consider the linear model (3.1) and (3.2) with informative (conjugate) normal-gamma prior distribution

$$(\beta, \sigma^2) \sim N\Gamma(\beta_*, H_*, s_*^2, n_*). \quad (3.21)$$

The marginal likelihood is then given by

$$f(y) = \pi^{-\frac{T}{2}} \frac{|H_{**}|^{1/2}}{|H_*|^{1/2}} \cdot \frac{\Gamma(n_{**}/2)}{\Gamma(n_*/2)} \cdot \frac{(n_* s_*^2)^{\frac{n_*}{2}}}{(n_{**} s_{**}^2)^{\frac{n_{**}}{2}}}, \quad (3.22)$$

where

$$\begin{aligned} n_{**} &= n_* + T, \\ H_{**}^{-1} &= X'X + H_*^{-1}, \\ n_{**} s_{**}^2 &= n_* s_*^2 + ESS + (\beta_* - \hat{\beta})'((X'X)^{-1} + H_*)^{-1}(\beta_* - \hat{\beta}), \\ ESS &= (y - X\hat{\beta})'(y - X\hat{\beta}), \end{aligned}$$

and

$$\hat{\beta} = (X'X)^{-1}X'y.$$

#### Module usage

##### Function Name:

`m1inf`

##### Usage:

`m1inf(dataY,Mx,dataX,b.star,H.star,s2,n.star)`

##### Required Arguments:

**dataY:** a univariate time series or a vector.  
**Mx:** the number of columns of the X matrix.  
**dataX:** a matrix of time series or a vector.

##### Optional Arguments:

**b.star:** a  $(1+p) \times 1$  mean vector of  $\beta$ :  $\beta_*$  in (3.21). Default:  $c(\text{rep}(0, Mx+1))$   
**H.star:** a  $(1+p) \times (1+p)$  matrix, the inverse of the variance matrix  $H_*$  in (3.21):  $H_*^{-1}$ . Default:  $\text{diag}(1, Mx+1)$

`s2`: the prior scale parameter of  $s^2$  in (3.21):  $s_*^2$ . Default:  $\text{var}(y)/2$

`n.star`: prior d.f.:  $n_*$ . Default: 1

**Example:**

```
>ml.inf.model<-mlinf(MSCI[,1],Mx=1,MSCI.W).
```

You can fit a simple regression model and store it as an object *ml.model* with the function above. The function *names* gives the components of the object *ml.inf.model*, which are *beta*, *sigma* and *log.ML*. The first component of *beta* gives the intercept and the second the slope coefficient.

The component *log.ML* gives the marginal likelihoods of the regression model. *Sigma* is the residual standard error. In our case we have a slope of 0.738, which is the estimated beta according to the CAPM. If the whole market changes by a factor of 1, the MSCI North America index will change by a factor of 0.738.

```
$beta: [1] 0.00657064 0.73874502
```

```
$sigma: [1] 0.0491899
```

```
$log.ML: [1] 151.619
```

### 3.2.2 The marginal likelihood of an AR regression model with an informative prior

**Model definition**

We consider the linear autoregressive model

$$y_t = \beta_0 + \beta_1 y_{t-1} + \dots + \beta_p y_{t-p} + u_t \quad (3.23)$$

with a  $T \times k$  regression matrix  $X$  of full rank, where  $k = p + 1$ ,

$$y \sim N[X\beta, \sigma^2 I_T] \quad (3.24)$$

with informative (conjugate) normal-gamma prior distribution

$$(\beta, \sigma^2) \sim N\Gamma(\beta_*, H_*, s_*^2, n_*). \quad (3.25)$$

The marginal likelihood is then given by

$$f(y) = \pi^{-\frac{T}{2}} \frac{|H_{**}|^{1/2}}{|H_*|^{1/2}} \cdot \frac{\Gamma(n_{**}/2)}{\Gamma(n_*/2)} \cdot \frac{(n_* s_*^2)^{\frac{n_*}{2}}}{(n_{**} s_{**}^2)^{\frac{n_{**}}{2}}}, \quad (3.26)$$

where

$$n_{**} = n_* + T, \quad (3.27)$$

$$H_{**}^{-1} = X'X + H_*^{-1}, \quad (3.28)$$

$$n_{**}s_{**}^2 = n_*s_*^2 + ESS + (\beta_* - \hat{\beta})'((X'X)^{-1} + H_*)^{-1}(\beta_* - \hat{\beta}), \quad (3.29)$$

and

$$ESS = (y - X\hat{\beta})'(y - X\hat{\beta}), \quad (3.30)$$

$$\hat{\beta} = (X'X)^{-1}X'y. \quad (3.31)$$

### Module usage

#### Function Name:

`marinf`

#### Usage:

`marinf(dataY,Mx,dataX,pmax,b.star,H.star,s2,n.star)`

#### Required Arguments:

`dataY`: a univariate time series or a vector  
`Mx`: the number of columns of the X matrix.  
`dataX`: a matrix of time series or a vector.

#### Optional Arguments:

`pmax`: the maximum order of the AR process. Default: 5.  
`b.star`: a  $(1+p) \times 1$  prior mean vector of  $\beta$ :  $\beta_*$  in (3.25). Default: `c(rep(0,pmax+1))`  
`H.star`: a  $(1+p) \times (1+p)$  matrix of the inverse of variance matrix of H:  $H_*^{-1}$ . Default: `diag(1,pmax+1)`  
`s2`: a prior scale parameter of  $s^2$ :  $s_*^2$ . Default: 0.5  
`n.star`: a prior d.f.:  $n_*$ . Default: 1

#### Example:

`>mar.inf.model<-marinf(MSCI[,1],Mx=1,MSCI.W)`. In this example we regress the returns of the MSCI North America index on the lagged returns of the MSCI world index in USD. You can fit  $p$  ARX models and store them as an object 'ar.inf.model' with the function above. The function `names` gives the components of the object `ar.models`, which are `beta`, `sigma` and `log.ML`. `Beta` is a matrix with the number of columns equal to the fitted ARX models, in

this case five. The first column contains the coefficient from the ARX(1) model and the last the coefficient from the ARX(5) model. The component  $\log.ML$  is a vector of the marginal likelihoods for each model. The highest element of this vector gives us the order of the optimal AR-process. In this case, the first element is the largest, so we choose the ARX(1) model.

```
$prior$bstar: [1] 0 0 0 0 0 0
```

```
$prior$s2.star: [1] 200
```

```
$prior$HstarI:
```

```
      [,1] [,2] [,3] [,4] [,5] [,6]
[1,]  100   0   0   0   0   0
[2,]   0   1   0   0   0   0
[3,]   0   0   2   0   0   0
[4,]   0   0   0   3   0   0
[5,]   0   0   0   0   4   0
[6,]   0   0   0   0   0   5
```

```
$prior$n.star: [1] 1
```

```
$beta:
```

```
      [,1]      [,2]      [,3]      [,4]      [,5]
[1,] 0.0121670 0.0123906 0.0140354 0.0146094 0.0156386
[2,] -0.0561858 -0.0573544 -0.0886978 -0.0934321 -0.0909584
[3,] 0.0000000 -0.0351914 -0.0592342 -0.0505501 -0.0628578
[4,] 0.0000000 0.0000000 -0.1082164 -0.1021519 -0.1213351
[5,] 0.0000000 0.0000000 0.0000000 -0.1694741 -0.1837649
[6,] 0.0000000 0.0000000 0.0000000 0.0000000 -0.0577228
```

which is the OLS estimate of  $\hat{\beta}$  in (3.31) as a function of  $p$ .

```
$sigma: [1] 0.0378609 0.0379989 0.0376583 0.0367383 0.0367350
which is the residual standard deviation $log.ML: [1] 135.232
134.775 134.727 132.585 129.244,
```

which is the log marginal likelihood (3.26) as a function of  $p$ .

### 3.3 The Bayesian AR(p) Model

Consider the stationary time series  $\{y_1, y_2, \dots, y_T\}$  with

$$E(y_t) = \mu, \quad \text{Var}(y_t) = \sigma^2$$

and the AR(p) model as described in 3.23 which can be written as

$$\mathbf{x}'_t \beta + \varepsilon_t$$

with

$$\mathbf{x}'_t = \underbrace{(1, y_{t-1}, \dots, y_{t-p})}_{p+1 \times 1}, \beta'_t = (\beta_0, \dots, \beta_p).$$

Conditioning on  $p$  starting values  $y_0, y_{-1}, \dots, y_{-p+1}$  we can formulate a linear model

$$\mathbf{y} = \mathbf{X}\beta + \varepsilon,$$

where

$$\mathbf{y} = \begin{pmatrix} y_1 \\ \vdots \\ y_T \end{pmatrix}, \quad \mathbf{X}' = \underbrace{\begin{pmatrix} \mathbf{x}_1 & \dots & \mathbf{x}_T \end{pmatrix}}_{(p+1) \times T}, \quad \varepsilon = \begin{pmatrix} \varepsilon_1 \\ \vdots \\ \varepsilon_T \end{pmatrix}$$

the starting values are contained in  $\{x_1, x_2, \dots, x_p\}$ .

### AR(p) with Tightness Prior Distribution

For the prior distribution we assume as in Littermann (1986) a tightness structure in conjugate form:

$$\begin{aligned} (\beta, \sigma^{-2}) &\sim \mathcal{N}\Gamma[\beta_*, \mathbf{H}_*, s_*^2, n_*] \\ &= \mathcal{N}[\beta_*, \sigma^2 \mathbf{H}_*] \cdot \Gamma[\sigma^{-2} | s_*^2, n_*] \end{aligned}$$

$$\beta_* = \sigma_{p+1}, \quad \mathbf{H}_*^{-1} = \underbrace{\text{diag}(\varepsilon, 1, 2, \dots, p)\sigma_y^2}_{\text{"tightness structure"}}$$

$$\varepsilon = 10^{-6}, \quad s_*^2 = \frac{\sigma_y^2}{10}, \quad n_* = 1 \dots \text{d.f. for } \sigma_\varepsilon^2.$$

The posterior distribution is given by

$$\begin{aligned} (\beta, \sigma^{-2}) &\sim \mathcal{N}\Gamma[\beta_{**}, \mathbf{H}_{**}, s_{**}^2, n_{**}] \\ &= \mathcal{N}[\beta_{**}, \sigma^2 \mathbf{H}_{**}] \cdot \Gamma[s_{**}^2, n_{**}], \end{aligned}$$

$$\begin{aligned} \beta_{**} &= (\mathbf{X}'\mathbf{X} + \mathbf{H}_*^{-1})^{-1}(\mathbf{X}'\mathbf{X}\mathbf{b} + \mathbf{H}_*^{-1}\beta_*) \\ \mathbf{H}_{**}^{-1} &= \mathbf{X}'\mathbf{X} + \mathbf{H}_*^{-1} \\ n_{**}s_{**}^2 &= n_*s_*^2 + ESS + (\beta_* - \mathbf{b})'[(\mathbf{X}'\mathbf{X})^{-1} + \mathbf{H}_*]^{-1}(\beta_* - \mathbf{b}) \\ n_{**} &= T + n_* \\ ESS &= (\mathbf{y} - \mathbf{X}\mathbf{b})'(\mathbf{y} - \mathbf{X}\mathbf{b}) \\ \mathbf{b} &= (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y}. \end{aligned}$$

The marginal distribution for  $\beta$  is a multivariate  $t$  distribution

$$\begin{aligned} \beta | \mathbf{y} &\sim t_{p+1}[\beta_{**}, s_{**}^2 \mathbf{H}_{**}, n_{**}] \\ &\sim \not\prec \beta_{**}, \frac{n_{**}}{n_{**} - 2} s_{**}^2 \mathbf{H}_{**} \not\succeq. \end{aligned}$$

Each element of  $\beta$  is univariate t-distributed

$$\beta_i \sim t[\beta_{**i}, s_{**}^2 \mathbf{H}_{**ii}, n_{**}], \quad i = 1, \dots, p + 1.$$

### The Predictive Distribution

Forecasting the next observation at time  $T$  we use

$$\begin{aligned} y_{T+1} &= \beta_0 + \beta_1 y_T + \dots + \beta_p y_{T-p+1} + \varepsilon_{T+1} \\ &= \mathbf{x}'_{T+1} \beta + \varepsilon_{T+1} \end{aligned}$$

with  $\beta$  a location parameter of the posterior distribution (usually the posterior mean) and

$$\mathbf{x}'_{T+1} = (1, y_T, \dots, y_{T-p+1}).$$

The predictive density for  $Y_{T+1}$  is given by the  $t$  density

$$\begin{aligned} p(y_{T+1}|y) &= t[\mathbf{x}'_{T+1} \beta_{**}, s_{**}^2 M_{**}, n_{**}] \\ M_{**} &= 1 + \mathbf{x}'_{T+1} H_{**} \mathbf{x}_{T+1}. \end{aligned}$$

For large  $T (> 30)$  the  $t$  distribution converges to the normal distribution

$$\begin{aligned} \lim_{T \rightarrow \infty} p(y_{T+1}|y) &= N[\mathbf{x}'_{T+1} \beta_{**}, s_{**}^2 M_{**}] \\ &\sim \not\prec \mathbf{x}'_{T+1} \beta_{**}, \frac{n_{**}}{n_{**} - 2} s_{**}^2 M_{**} \not\succeq. \end{aligned}$$

### Function Name:

`premarin`

### Usage:

```
premarin(dataY = cpi.re, pmax = 1, b.star = c(rep(0, pmax + 1)), H1 = c(0.0001, 1:pmax)
* var(dataY), H.star.I = diag(H1), s2.star = var(dataY)/10, n.star = 1, libpath =
"c : \\Basel.PC\\object")
```

### Required Arguments:

`dataY`: a univariate time series or a vector.

### Optional Arguments:

`p`: the order of the AR process.  $k \geq 0$  is an integer. Default: 1.

`bstar`: the prior information about the regression coefficients. Default: `b.star = c(rep(0, pmax + 1))`.

`H1`: prior information for the precision matrix ( $H_*^{-1}$ ). Default: `H1 = c(0.0001, 1:pmax) * var(dataY)`.

`s2.star`: prior information for  $s_*^2$ . Default: `var(dataY)/10`.

`n.star`: prior d.f. for  $n$ . Default: 1.

**Example:**

`>premarin<-(cpi.re)`. The data used for the estimation are monthly returns of the real consumer expenditure in the USA from February 1990 until September 1999. The prior information for the slope coefficients are as above.

```
$prior$b.x: [1] 0 0
```

```
$prior$s2.x: [1] 2.23692e-007
```

```
$prior$H.x.I:
```

```
          [,1]      [,2]
[1,] 2.23692e-010 0.00000e+000
[2,] 0.00000e+000 2.23692e-006
```

```
$prior$n.x: 1
```

```
$beta: 0.00156492 0.33516205
```

```
$var: 1.9852e-006
```

```
$H.xx:
```

```
          [,1]      [,2]
[1,] 2.23692e-010 4.59094e-015
[2,] 4.59094e-015 2.21738e-006
```

```
$n.xx: 116
```

```
$s2.xx: 1.95505e-006
```

```
$b.xx: 0.0015718 0.3322354
```

```
$nstep: 1
```

```
$Predic.Mean: 4.66069e-006
```

```
$Predic.Var: 1.95505e-006
```

```
$log.ML: -440.418
```

### 3.4 ARCH models

The AR( $k$ )-GARCH( $p,q$ )-M( $s$ ) model can be written as:

$$y_t = \beta_0 + \sum_{i=1}^k y_{t-i}\beta_i + \sum_{i=1}^s h_{t-i}\psi_i + \varepsilon_t, \quad t = 1, \dots, T, \quad (3.32)$$

$$\varepsilon_t \sim N[0, h_t],$$

$$h_t = \alpha_0 + \sum_{i=1}^p \alpha_i h_{t-i} + \sum_{i=1}^q \phi_i \varepsilon_{t-i}^2. \quad (3.33)$$

Compactly, we can write the equation (3.32) as follows:

$$y_t \sim N[\mathbf{x}'_t \boldsymbol{\beta}, h_t],$$

where

$$\begin{aligned} \mathbf{x}'_t &= (1, y_{t-1}, \dots, y_{t-k}, h_{t-1}, \dots, h_{t-s}) \\ \boldsymbol{\beta} &= (\beta_0, \beta_1, \dots, \beta_k, \psi_1, \dots, \psi_s)' \\ h_t &= z'_t \boldsymbol{\alpha} \end{aligned} \quad (3.34)$$

is a  $1 \times (1 + k + s)$  vector of regression coefficients and

$$\boldsymbol{\alpha} = (\alpha_0, \alpha_1, \dots, \alpha_p, \phi_1, \dots, \phi_q)' \quad (3.35)$$

$$z'_t = (1, h_{t-1}, \dots, \varepsilon_{t-q}^2) \quad (3.36)$$

is a  $1 \times (1 + p + q)$  vector of unknown parameters.

The default or automatic prior distribution is specified as

$$\boldsymbol{\beta} \sim N[\mathbf{b}_*, \mathbf{H}_*], \quad (3.37)$$

$$\boldsymbol{\alpha} \sim N_0^\infty[\boldsymbol{\alpha}_*, \mathbf{D}_*], \quad (3.38)$$

where the prior mean is

$$\mathbf{b}_* = \mathbf{0}, \quad (3.39)$$

and the prior precision

$$\mathbf{H}_* = \text{diag}(\varepsilon, 1, \dots, k + s), \quad (3.40)$$

with

$$\varepsilon = 10^{-6}, \quad (3.41)$$

$$\boldsymbol{\alpha}_* = 0.11\mathbf{1}_{1+p+q}, \quad \mathbf{D}_* = 0.01\mathbf{I}_{1+p+q}, \quad (3.42)$$

where  $N_0^\infty$  denotes a truncated normal distribution.

### 3.4.1 The full conditional distributions (f.c.d.)

a) The f.c.d. for  $\beta$  is a normal distribution

$$f(\beta|\alpha, \phi, \mathbf{y}) = N[\mathbf{b}_{**}, \mathbf{H}_{**}], \quad (3.43)$$

$$\mathbf{H}_{**}^{-1} = \mathbf{H}_*^{-1} + \mathbf{X}'\mathbf{D}_h^{-1}\mathbf{X}, \quad (3.44)$$

$$\mathbf{D}_h = \text{diag}(h_1, \dots, h_T), \quad (3.45)$$

$$\mathbf{b}_{**} = \mathbf{H}_{**}(\mathbf{H}_*\mathbf{b}_* + \mathbf{X}'\mathbf{D}_h^{-1}\mathbf{y}), \quad (3.46)$$

where  $\mathbf{H}_*$  and  $\mathbf{b}_*$  are the parameters of the prior distribution.

b) The f.c.d. for  $\alpha$  and  $\phi$  are obtained by the proposal distributions

We use for the f.c.d. of  $\alpha$  and  $\phi$  a Metropolis-within-Gibbs step with a normal distribution which is obtained by an iteration proposal given by

$$\alpha_i \sim N[\hat{\alpha}_i, \hat{\sigma}_i^2], \quad i = 0, 1, \dots, p,$$

and

$$\phi_i \sim N[\hat{\phi}_i, \hat{\sigma}_i], \quad i = 1, \dots, q.$$

The f.c.d. for  $\alpha$  is given by

$$\pi(\alpha|\mathbf{y}, \beta) = \prod_{t=1}^T N[y_t|\mathbf{x}_t'\beta, h_t]$$

where the normal distribution is proportional to

$$N[y_t|\mathbf{x}_t'\beta, h_t] \propto |h_t|^{-1/2} \exp\{-\frac{1}{2}(y_t - \mathbf{x}_t'\beta)'h_t^{-1}(y_t - \mathbf{x}_t'\beta)\}.$$

The mean  $\hat{\alpha}_i = \hat{\phi}_i = 0.01$ , and the variance  $\hat{\sigma}_i^2 = 0.001$  are obtained iteratively from Metropolis runs (e.g. 3 times one thousand runs). Then we calculate the mean and variance and repeat the iterative proposal until a satisfactory accept/rejection ratio is obtained. The *argarchm* function needs a vector or time series as its input data file, which is S-PLUS object. Finally, if one runs the program, the posterior logarithmic marginal likelihood of the model will be reported in the output file.

#### Function Name:

`argarchm`

#### Usage:

```
argarchm(data = MSCI.W, k = 1, s = 1, p = 1, q = 1,
iter = 100, bstar = c(rep(0, 1 + k + s)), Hx = diag(c(1e-005, 1:(k + s))),
alphax = c(rep(0.01, 1 + p + q)), Dx = diag(0.001, 1 + p + q),
nstep = 1, nnrec = 50, show.para.hist = F, show.para = T,
show.para.Markov = F, show.epsilon.plot = F, show.ht.plot = T, show.fore = T, init
= T)
```

#### Required Arguments:

`data`: a univariate time series or a vector.

#### Optional Arguments:

- k:** the order of the AR process.  $k \geq 0$  is an integer. Default: 1.
- s:** the order of the  $h_t$  process.  $s \geq 0$  is an integer. Default: 1.
- p:** the order of the  $h_t$ s in the GARCH process.  $p \geq 0$  is an integer. Default: 1.
- q:** the order of the  $\epsilon_t^2$ s in the GARCH process.  $q \geq 0$  is an integer. Default: 1.
- iter:** the number of iterations.  $iter \geq 1$  is an integer. Default: 100.
- nnrec:** the number of the last iterations to be recorded.  $nnrec \geq 1$  is an integer. Default: 50.
- Hx:** a  $(1+k+s) \times (1+k+s)$  variance matrix of the prior distribution for  $\beta^*$ . Default: `diag(c(1e-005, 1:(k + s)))`.
- Dx:** a  $s \times s$  variance matrix of the proposal distribution for  $D_{\alpha^*}^{-1}$ . Default: `diag(0.001, 1 + p + q)`.
- alphax:** a  $(1+p+q) \times 1$  vector of prior means for  $\alpha_i$ . Default: `c(rep(0.01, 1+p+q))`.
- show.para:** is a logical variable. If one chooses  $T$ , the program will print out the mean and standard error of the simulated posterior of the model. Default: **F**.
- show.para.hist:** is a logical variable. If one chooses  $T$ , the program will show the histogram (graphic output) of the simulated posterior distribution priors of the model parameters for all iterations. Default: **F**.
- show.para.Markov:** is a logical variable. If one chooses  $T$ , the program will plot a graphic, which plots the Markov chains (graphic output) of the simulated posterior of the model for all iterations. Default: **F**.
- show.epsilon.plot:** is a logical variable. If one chooses  $T$ , the program will plot a graphic, which plots the residual of the model for all iterations. Default: **F**.
- show.epsilon:** is a logical variable. If one chooses  $T$ , the program will print out the mean and standard error of the model's residual for all iterations. Default: **F**.
- show.ht.plot:** is a logical variable. If one chooses  $T$ , the program will plot a graphic, which plots the conditional variance of the model for all iterations. Default: **F**.
- show.ht:** is a logical variable. If one chooses  $T$ , the program will print out the mean and standard error of the model's conditional variance of iterations. Default: **F**.
- nstep:** is an integer variable, which is the number of the forecasting period, if  $nstep = 1$ , the program will do one step forecasting calculation. Default: 0.
- show.fore:** is a logical variable. If one chooses  $T$ , the program will show the histogram and the mean for all iterations of the every forecasting period. Default: **F**.
- init:** if **TRUE**, load the Fortran module. Default: **T**.
- libpath:** path to the Fortran module. Default: `libpath`.

**Example:**

`my.model<-argarchm(data=MSCI.W)`. You fit a AR(1)-GARCH(1,1)-M(1) model for the monthly returns of the MSCI.W index from February 1990 until September 1999 and store it as an object 'my.model'. By default, you get the estimated coefficients and the std. error, one step forecast of the series and the marginal likelihood. The command plots the means of the conditional variance and the histogram of the forecasted time series by default.

```

Model:                AR(1)-GARCH(1,1)-M(1) Model
Data:                 MSCI.W
Data's mean and std.: 0.007019    0.040205

```

```
$Model: [1] "AR(1)-GARCH(1,1)-M(1) Model"
```

```
$data.length: [1] 116
```

```
$prior: $prior$b.x : [1] 0 0 0 ,
```

( $b_*$  in (3.39))

```
$prior$H.x:
```

```

      [,1] [,2] [,3]
[1,] 1e-005  0    0
[2,] 0e+000  1    0
[3,] 0e+000  0    2

```

( $H_*$  in (3.40)).

```
$prior$alpha.x: [1] 0.01 0.01 0.01
```

( $\alpha_*$  in (3.42)).

```
$prior$D.x:
```

```

      [,1] [,2] [,3]
[1,] 0.001 0.000 0.000
[2,] 0.000 0.001 0.000
[3,] 0.0000.0000.001

```

( $D_*$  in (3.42)).

```
$coefficients:
```

```

      Mean          Std
[1,] 0.011100572  4.34627e-003
[2,] -0.125464727  1.27358e-001
[3,]-0.108561293  3.48803e-001
[4,] 0.000963236  1.90871e-007
[5,]0.320863669 4. 46307e-002
[6,] 0.164033983  1.65275e-002

```

(posterior mean  $b_{**}$  with standard deviations.)

```
$forecast:
      fore.mean fore.std
[1,] -0.0110996 0.0101414,
```

which is a one step ahead forecast.

```
$log.marginal.likelihood: [1] -109.313
```

Note:

We can fit a GARCH model with the function *garch*, an ARCH model with the function *arch* and a GARCH - in - mean model with the function *garchm*, e.t.c.

### Example:

`my.model<-garch(data=MSCI.W,q=2)`. With the command above you fit GARCH(1,2) for the same time series and store it as an object '*my.model*'. We can compare the marginal likelihood with the model above.

```
Model:          GARCH(1,2) Model
Data:           MSCI.W
Data's mean and std.:  0.007019   0.040205
```

```
$Model: [1] "GARCH(1,2) Model"
```

```
$data.length: [1] 116
```

```
$prior: $prior$b.x: [1] 0
```

```
$prior$H.x:
      [,1] [,2] [,3]
[1,] 1e-005  0    0
[2,] 0e+000  1    0
[3,] 0e+000  0    0
```

```
$prior$alpha.x: [1] 0.01 0.01 0.01 0.01
```

```
$prior$D.x:
      [,1] [,2] [,3] [,4]
[1,] 0.001 0.000 0.000 0.000
[2,] 0.000 0.001 0.000 0.000
[3,] 0.000 0.000 0.001 0.000
[4,] 0.000 0.000 0.000 0.001
```

```

$coefficients:
      Mean      Std
[1,] 0.00956259 4.61517e-003
[2,] 0.00126690 2.65250e-007
[3,] 0.19396706 9.00266e-003
[4,] 0.19015266 2.82230e-003
[5,] 0.09079461 9.18119e-004

$forecast:
      fore.mean fore.std
[1,] -0.0098747 0.0101721

$log.marginal.likelihood: [1] -110.916

```

**Example:**

`my.model<-arch(data=MSCI.W,q=2)`. With the command above you fit an ARCH(1) for the same time series and store it as an object `'my.model'`. We can compare the marginal likelihood with the model above.

```

Model:          ARCH(1,0) Model
Data:           MSCI.W
Data's mean and std.: 0.007019 0.040205

$model: [1] "ARCH(1,0) Model"

$data.length: [1] 116

$prior: $prior$b.x: [1] 0

$prior$H.x:
      [,1] [,2] [,3]
[1,] 1e-005 0 0
[2,] 0e+000 1 0
[3,] 0e+000 0 0

$prior$alpha.x: [1] 0.01 0.01

$prior$D.x:
      [,1] [,2]
[1,] 0.001 0.000 [2,] 0.000 0.001

```

```

$coefficients:
      Mean      Std
[1,] 0.00835234 4.20489e-003
[2,] 0.00099282 3.17223e-007
[3,]0.16462704 4.40833e-003

$forecast:
      fore.mean fore.std
[1,] -0.0105661 0.0106197

$log.marginal.likelihood: [1] -112.706

```

**Example:**

`my.model<-garchm(data=MSCI.W)`. With the command above you fit a GARCH(1,1)-in-mean model for the same time series and store it as an object 'my.model'. We can compare the marginal likelihood with the model above.

```

Model:          GARCH(1,1)-M(1) Model
Data:           MSCI.W
Data's mean and std.: 0.007019  0.040205

$Model: [1] "GARCH(1,1)-M(1) Model"

$data.length: [1] 116

$prior: $prior$b.x: [1] 0 0

$prior$H.x:
      [,1] [,2]
[1,] 1e-005  0
[2,] 0e+000  1

$prior$alpha.x: [1] 0.01 0.01 0.01

$prior$D.x:
      [,1] [,2] [,3]
[1,] 0.001 0.000 0.000
[2,] 0.000 0.001 0.000
[3,] 0.0000.0000.001

$coefficients:
      Mean      Std

```

```
[1,] 0.0123303      2.93551e-003
[2,] -0.0512458     3.12984e-001
[3,] 0.0010681      1.67671e-007
[4,] 0.3484503      1.67427e-002
[5,] 0.1101007      6.56985e-003
```

```
$forecast:
```

```
      fore.mean fore.std
[1,] -0.0111063 0.0103564
```

```
$log.marginal.likelihood: [1] -110.667
```

# Chapter 4

## Multivariate models

### 4.1 Multivariate models with fractional prior

#### 4.1.1 The marginal likelihood of a multivariate regression model with fractional prior

##### Model definition

The model is

$$Y_{T \times M_y} \sim N_{T \times M_y}[XB, \Sigma \otimes I_T],$$

where  $Y = (y_1, \dots, y_{M_y})$ ,  $X = (1, x_1, \dots, x_{M_x})$ ,  $x_i = (x_i^1, \dots, x_i^T)'$ ,  $B = (b_0, \dots, b_{M_y})$  and  $b_i = (b_i^0, \dots, b_i^{M_x})$ .

The fractional marginal likelihood for the normal-Wishart regression model is given by

$$f_b^1(Y) = b^{\frac{Tb}{2}} |\pi \hat{U}' \hat{U}|^{-\frac{T-Tb}{2}} \Gamma_{M_y} \left( \frac{T}{2} \right) / \Gamma_{M_y} \left( \frac{Tb}{2} \right), \quad (4.1)$$

and for the simplest choice of  $b = \frac{2}{T}$  we find

$$f_b^1(Y) = \left( \frac{2}{T} \right) |\pi \hat{U}' \hat{U}|^{-\frac{T-2}{2}} \Gamma_{M_y} \left( \frac{T}{2} \right) \quad (4.2)$$

with the residuals

$$\hat{U} = Y - X\hat{B}$$

and

$$\Gamma_M \left( \frac{T-1}{2} \right) = \prod_{i=1}^M \Gamma \left( \frac{T-i}{2} \right). \quad (4.3)$$

The ML estimate of the residual covariance matrix is

$$\hat{\Sigma} = \hat{U}' \hat{U} / T.$$

**Module usage****Function Name:**

`mmlfra`

**Usage:**

`mmlfra(dataY,Mx,dataX,b)`

**Required Arguments:**

**dataY:** a matrix of time series.

**Mx:** the number of columns of the X matrix.

**dataX:** a matrix of time series or a vector.

Note: the fractional parameter `b` is given by default in the program as  $\frac{1}{T}$ , where `T` is the number of observations, and cannot be changed.

**Example:**

```
>mml.model<-mmlfra(MSCI[,1:2],Mx=1,MSCI.W).
```

We regress the returns of the MSCI North America index and of the MSCI Pacific index on the returns of the MSCI world index to estimate the *beta*. A high *beta* (say greater than 1) means that the local market is highly sensitive to changes in the world market.

The function `names` gives the components of the object `mml.model`, which are *marginal.likelihood*, *beta* and *Covar*.

*beta* has the dimension `[,]`, where `[1,1]` is the intercept coefficient for the first equation, that is for MSCI North America, and `[2,2]` is the slope coefficient of the second equation. Comparing the betas for MSCI North America and MSCI Pacific, 0.73 and 1.40, respectively, we see that the Pacific market is much more sensitive to changes in the MSCI world index. Having estimated the *beta*, we can construct a portfolio with any given *beta* because the portfolio beta is just the sum of the individual *beta* weighted by the individual holdings.

The component *marginal.likelihood* is the marginal likelihood of the regression model. *Covar* is the estimated covariance matrix.

Model output is given by

```
$marginal.likelihood:
```

```
  [,1]
```

```
[1,] 513.23
```

```
$beta: , , 1
```

```
      [,1]      [,2]
```

```
[1,] 0.00657064 -0.0111908
```

```
[2,] 0.73874502  1.4004484
```

```
$covar:
```

, , 1  
 [,1] [,2]  
 [1,] 0.000543935 -0.000699212  
 [2,] -0.000699212 0.001551994

### 4.1.2 Multivariate regression model with outliers

In this section we consider the following multivariate regression model with outliers for the  $T \times My$  matrix  $Y$

$$Y = XB + D_j\theta + U, \quad j = 1, \dots, T. \quad (4.4)$$

Again  $D_j$  is a dummy variable defined as the  $j$ -th unity vector of dimension  $T$ , and  $\theta$  is a  $(1 \times My)$  row vector of outliers (a location shift for the  $My$  different regressions), where  $Y = (y_1, \dots, y_{My})$ ,  $X = (1, x_1, \dots, x_{Mx})$ ,  $x_i = (x_i^1, \dots, x_i^T)'$ ,  $B = (b_0, \dots, b_{My})$  a  $(Mx \times My)$  coefficient matrix and  $b_i = (b_i^0, \dots, b_i^{Mx})$ .

The errors are multivariate normally distributed:

$$U \sim N_{T \times My}[0, \Sigma \otimes I_T].$$

The multivariate model can be written in compact form as

$$Y = \tilde{X}\tilde{B} + U, \quad \text{or} \quad Y \sim N[\tilde{X}\tilde{B}, \Sigma \otimes I_T] \quad (4.5)$$

with  $\tilde{X} = (X : D_j)$  a  $T \times (2 + Mx)$  regressor matrix, and  $\tilde{B}' = (B' : \theta')$  the  $(2 + Mx) \times My$  matrix of regression coefficients. For the homoskedastic multivariate regression model, (4.4), the fractional marginal likelihood for  $b \in (0, 1)$  is given by

$$f_b^1(Y) = b^{\frac{Tb}{2}} |\pi \hat{U}' \hat{U}|^{-\frac{T(1-b)}{2}} \Gamma_{My} \left( \frac{T - Mx - 1}{2} \right) / \Gamma_{My} \left( \frac{Tb - Mx - 1}{2} \right) \quad (4.6)$$

with  $\Gamma_M$  as in (4.3) and the estimated residual matrix

$$\hat{U} = Y - \tilde{X}\hat{B}, \quad \text{and} \quad \hat{B} = (\tilde{X}'\tilde{X})^{-1}\tilde{X}'Y. \quad (4.7)$$

Note that for the fraction  $b = \frac{Mx+3}{T}$  we get the simpler formula

$$f_b^1(Y) = \left( \frac{Mx+3}{T} \right)^{\frac{Mx+3}{T}} |\pi \hat{U}' \hat{U}|^{-\frac{T-Mx-3}{2}} \Gamma_{My} \left( \frac{T - Mx - 1}{2} \right). \quad (4.8)$$

The ML estimate of the residual matrix is

$$\hat{\Sigma} = \hat{U}'\hat{U}/T. \quad (4.9)$$

**Proof:** Use the results of Polasek and Ren (1997) for the homoskedastic regression model.

**Module usage****Function Name:**

mmloutfra

**Usage:**

mmloutfra(dataY,Mx,dataX,b)

**Required Arguments:****dataY:** a matrix of time series time series or a vector.**Mx:** the number of columns of the X matrix, if vector  $Mx = 1$ .**dataX:** a regressor matrix of time series or a vector.**Example:**

```
>mmloutfra(MSCI[,c(1,9)],Mx=1,MSCI.W).
```

The command above will plot the dependent time series and the marginal likelihood.

Reference: see Polasek and Ren (1997).

**4.1.3 The marginal likelihood of a VAR regression model with a fractional prior****Model definition**

The model is

$$Y_{T \times My} \sim N_{T \times My}[XB, \Sigma \otimes I_T],$$

where  $Y = (y_1, \dots, y_{My})$ ,  $y_i = (y_i^1, \dots, y_i^T)'$ ,  $X = (1, x_1, \dots, x_{Mx})$ ,  $x_i = (x_i^1, \dots, x_i^p)'$ . The fractional marginal likelihood for the normal-Wishart regression model is given by

$$f_b^1(Y) = b^{\frac{Tb}{2}} |\pi \hat{U}' \hat{U}|^{-\frac{T-Tb}{2}} \Gamma_{My} \left( \frac{T-k+1}{2} \right) / \Gamma_{My} \left( \frac{Tb-k+1}{2} \right), \quad (4.10)$$

and for  $b = \frac{k+1}{T}$  (and a non-informative prior) we have

$$f_b^1(Y) = \left( \frac{k+1}{T} \right)^{\frac{k+1}{2}} |\pi \hat{U}' \hat{U}|^{-\frac{T-k-1}{2}} \Gamma_{My} \left( \frac{T-k+1}{2} \right) \quad (4.11)$$

with the residuals

$$\hat{U} = Y - X\hat{B}$$

and

$$\Gamma_{My} \left( \frac{T-k}{2} \right) = \prod_{i=1}^{My} \Gamma \left( \frac{T-k-i+1}{2} \right)$$

for  $p = 1, \dots, p_{max}$  and  $k = p + 1$ . The ML estimate of the residual covariance matrix is

$$\hat{\Sigma} = \hat{U}' \hat{U} / (T - p).$$

**Module usage****Function Name:**

`mvarfra`

**Usage:**

`mvarfra(dataY,Mx,dataX,pmax,b)`

**Required Arguments:**

**dataY:** a matrix of time series with dimension  $My$ .

**Mx:** the number of columns of the X matrix.

**dataX:** a matrix of time series or a vector.

**pmax:** the maximum order of the AR process. Default:3

Note: the fractional parameter  $b$  is given by default in the program and cannot be changed.

**Example:**

```
>var.models<-mvarfra(MSCI[,1:3],Mx=1,MSCI.W,pmax=3).
```

We will regress the monthly returns of the MSCI North America, MSCI France and MSCI Germany indices on the returns of the MSCI world index with the VAR-X model.

You can fit  $p$  VAR-X-models and store them as an object *var.models* with the function above.

The function `names` gives you the components of the object *ar.models*, which are *marginal.likelihood*, *beta* and *covar*.

The component *marginal.likelihood* is the vector of the marginal likelihoods for each model. The largest element of this vector is an estimate of the order of the AR-process. In this case the first element is the largest, so we choose the VAR(1)-X-model.

*beta* is a list with each element matrix. The number of columns is equal to the number of the columns of the matrix *dataY*, the first column contains the coefficients from the first equation, that is, for the first column of the matrix *dataY*. The first row contains the constants, the second the coefficients of the AR(1) process, etc.

The component *covar* is a list, where the first element is the estimated covariance matrix for the AR(1) model, the second for the AR(2), etc.

Output of the model is given by

```
$marginal.likelihood:
```

```
 [,1]
```

```
[1,] 612.017 [2,] 609.810 [3,] 609.405
```

References: see Polasek and Ren (1997).

### 4.1.4 The fractional marginal likelihood for the VARX(p) model with outliers

#### Model definition

In this section we consider the multivariate autoregressive model with outliers. Let  $Y$  be a matrix of the dependent variables and, in the case of an AR process of order  $p$ , the first column of  $X$  is a vector of ones and the other columns are  $p$  lags such that  $k = p + 1$ . The ARX(p) program specifies a  $T \times (k + p)$  matrix of the full rank of the independent variables and the AR terms. We consider  $T$  outlier models, indexed by  $j = 1, \dots, T$ , i.e.

$$Y = X_j \beta_j + D_j \theta + \varepsilon, \quad j = 1, \dots, T, \quad (4.12)$$

where  $D_j$  is a dummy variable, i.e. the  $j$ -th unity vector of length  $T$ .  $\theta$  is the regression coefficient which measures the size of the outlier effect at observation  $j$  on the regression model.

The multivariate outlier model can be written in compact form as

$$Y = \tilde{X}_j \tilde{\beta}_j + \varepsilon \quad (4.13)$$

with  $\tilde{X}_j = (X : D_j)$  and  $\tilde{\beta}_j = (\beta_j : \theta)$ . If no confusion is possible, we will drop the index  $j$ . The ordinary least squares or OLS estimate of model (4.13) is given by

$$\tilde{\beta}_j = (\tilde{X}_j' \tilde{X}_j)^{-1} \tilde{X}_j' y. \quad (4.14)$$

Assuming a fractional prior distribution, the fractional marginal likelihood is given by

$$f_b^1(Y|j) = b^{\frac{Tb-k}{2}} (\pi ESS_j)^{-\frac{T(b-1)}{2}} \Gamma\left(\frac{T-k}{2}\right) / \Gamma\left(\frac{Tb-k}{2}\right). \quad (4.15)$$

#### Module usage

##### Function Name:

`mvaroutfra`

##### Usage:

`maroutfra(dataY, Mx, dataX, pmax)`

##### Required Arguments:

`dataY`: a matrix of time series or a vector.

`Mx`: the number of columns of the X matrix.

`dataX`: a matrix of time series or a vector.

`pmax`: the maximum order of the AR process. Default:2

##### Example:

`>maroutfra(MSCI[,c(1,9)], Mx=1, MSCI.W, pmax=2)`. The command above will plot the data MSCI North America and MSCI Pacific and the marginal likelihood for VAR(1) and VAR(2).

## 4.2 Multivariate models with informative prior

### 4.2.1 Marginal likelihood of a multivariate regression model with an informative prior

#### Model definition

Consider the regression model for the  $T \times M$  matrix  $Y$

$$Y = XB + U, \quad (4.16)$$

or

$$Y \sim N[XB, \Sigma \otimes C_*],$$

where  $C_*$  is a known covariance matrix (usually  $C_* = I_T$ ). Assuming a normal-Wishart prior

$$f(B, \Sigma^{-1}) = NW[B_*, H_*, \Sigma_*, n_*],$$

the marginal likelihood is

$$f(Y) = (2\pi)^{-\frac{MT}{2}} \frac{c_{n_{**}}}{c_{n_*}} \frac{|\Sigma_*|^{\frac{n_*}{2}} |H_{**}|^{\frac{M}{2}}}{|\Sigma_{**}|^{\frac{n_{**}}{2}} |H_*|^{\frac{M}{2}}} \quad (4.17)$$

with

$$\begin{aligned} \Sigma_{**} &= \Sigma_* + \hat{U}'\hat{U} + \Delta, \quad \hat{U} = Y - X\hat{B}, \\ H_{**}^{-1} &= X'X + H_*^{-1}, \\ \Delta &= (\hat{B} - B_*)'[(X'X)^{-1} + H_*]^{-1}(\hat{B} - B_*), \\ \hat{B} &= (X'X)^{-1}X'Y, \\ n_{**} &= n_* + T, \end{aligned}$$

and  $c_{n_*}$  is given by

$$c_{n_*} = 2^{\frac{Mn_*}{2}} \pi^{\frac{M(M-1)}{4}} \prod_{j=1}^M \Gamma\left[\frac{n_* + 1 - j}{2}\right]. \quad (4.18)$$

#### Module usage

##### Function Name:

`mmlinf`

##### Usage:

`mmlinf(My=3, dataY=MSCI[, 1:2], Mx=1, dataX=MSCI.W)`

##### Required Arguments:

- Yx:** the number of columns of the Y matrix.
- dataY:** a matrix of time series or a vector.
- Mx:** the number of columns of the X matrix.
- dataX:** a matrix of time series or a vector.

**Example:**

```
>mml.inf.model<-mmlinf(My=2,MSCI[,c(1:9)],Mx=1,MSCI.W).
```

We regress the returns of the MSCI North America index and of the MSCI Pacific index on the returns of the MSCI world index to estimate the *beta*. A high *beta* (say greater than 1) means that the local market is highly sensitive to changes in the world market.

The function `names` gives the components of the object `mml.model`, which are *marginal.likelihood*, *beta* and *covar*.

Beta has the dimension `[,,]`, where `[1,1,]` is the intercept coefficient for the first equation, that is for MSCI North America, and `[2,2,]` is the slope coefficient of the second equation. Comparing the *beta* for MSCI North America and MSCI Pacific, 0.73 and 1.40, respectively, we see that the Pacific market is much more sensitive to changes in the MSCI world index. Having estimated the *beta*, we can construct a portfolio with any given *beta* because the portfolio *beta* is just the sum of the individual betas weighted by the individual holdings.

```
$marginal.likelihood:
```

```
      [,1]
[1,] -296.634
```

The component 'marginal.likelihood' is the marginal likelihood of the regression model.

```
$beta:
```

```
      , , 1
      [,1]      [,2]
[1,] 0.00657064 -0.0111908
[2,] 0.73874502  1.4004484
```

```
$covar:
```

```
      , , 1
      [,1]      [,2]
[1,] 4.64867e-006 -5.97573e-006 [2,] -5.97573e-006  1.32639e-005
```

'Covar' is the covariance matrix estimated as  $\frac{\hat{U}'\hat{U}}{n}$ .

## 4.2.2 The marginal likelihood of a VARX regression model with an informative prior

### Model definition

Consider the regression model for the  $T \times M$  matrix  $Y$

$$Y = XB + U, \quad (4.19)$$

or

$$Y \sim N[XB, \Sigma \otimes C_*],$$

where  $C_*$  is a known covariance matrix (usually  $C_* = I_T$ ). Assuming a normal-Wishart prior

$$f(B, \Sigma^{-1}) = NW[B_*, H_*, \Sigma_*, n_*],$$

the marginal likelihood is

$$f(Y) = (2\pi)^{-\frac{MT}{2}} \frac{c_{n_{**}}}{c_{n_*}} \frac{|\Sigma_*|^{\frac{n_*}{2}}}{|\Sigma_{**}|^{\frac{n_{**}}{2}}} \frac{|H_{**}|^{\frac{M}{2}}}{|H_*|^{\frac{M}{2}}} \quad (4.20)$$

with

$$\begin{aligned} \Sigma_{**} &= \Sigma_* + \hat{U}'\hat{U} + \Delta, & \hat{U} &= Y - X\hat{B}, \\ H_{**}^{-1} &= X'X + H_*^{-1}, \\ \Delta &= (\hat{B} - B_*)'[(X'X)^{-1} + H_*]^{-1}(\hat{B} - B_*), \\ \hat{B} &= (X'X)^{-1}X'Y, \\ n_{**} &= n_* + T, \end{aligned}$$

and  $c_{n_*}$  is given by

$$c_{n_*} = 2^{\frac{Mn_*}{2}} \pi^{\frac{M(M-1)}{4}} \prod_{j=1}^M \Gamma\left[\frac{n_* + 1 - j}{2}\right]. \quad (4.21)$$

### Module usage

#### Function Name:

`mvarinf`

#### Usage:

`mvarinf(My, dataY, Mx, dataX, pmax)`

#### Required Arguments:

- My**: the number of columns of the Y matrix.
- dataY**: a matrix of time series or a vector.
- Mx**: the number of columns of the X matrix.
- dataX**: a matrix of time series or a vector.
- pmax**: the maximum order of the AR process. Default:3

**Example:**

```
>var.inf.models<-mvarinf(My=3,MSCI[,1:3],Mx=1,MSCI.W,pmax=3).
```

We will regress the monthly returns of the MSCI North America, MSCI France and MSCI Germany indices on the returns of the MSCI world index with the VAR-X model.

You can fit VAR(p)-X-models and store them as an object *var.models* with the function above. The function `names` gives the components of the object *ar.models*, which are *marginal.likelihood*, *beta* and *covar*.

The component *marginal.likelihood* is the vector of the marginal likelihoods for each model. The largest element of this vector gives us the order of the optimal AR-process. In this case the first element is the largest, so we choose the VAR(1)-X-model.

Beta is a list with each element matrix. The number of columns is equal to the number of the columns of the matrix *dataY*, the first column contains the coefficients from the first equation, that is for the first column of the matrix *dataY*. The first row contains the constants, the second the coefficients of the AR(1) process, etc.

The component *Covar* gives the estimated covariance matrix for each AR-process.

```
$marginal.likelihood: [1,] -422.994 [2,] -443.166 [3,] -443.002
```

### 4.3 The VAR(k)-GARCH(p,q)-X(s) models

The VAR(k)-GARCH(p,q)-X(s) model can be written in the following way (see Polasek and Ren (1997)):

$$y_t^l = \beta_0^l + \sum_{m=1}^M \sum_{i=1}^k \beta_i^{lm} y_{t-i}^m + \sum_{m=1}^{M^x} \sum_{i=1}^r \psi_i^{lm} x_{t-i}^m + u_{l,t} \quad (4.22)$$

$$u_{l,t} \sim N[0, h_t^l], \quad l = 1, \dots, M,$$

$$h_t^l = \alpha_0^l + \sum_{m=1}^M \left( \sum_{i=1}^p \alpha_i^{lm} h_{t-i}^m + \sum_{i=1}^q \phi_i^{lm} u_{m,t-i}^2 \right), \quad (4.23)$$

where the parameters are subject to the stationarity constraint

$$\sum_{m=1}^M \left( \sum_{i=1}^p \alpha_i^{lm} + \sum_{i=1}^q \phi_i^{lm} \right) < 1, \quad (4.24)$$

with all coefficients being positive:  $\alpha_0^{lm} > 0$ ,  $\alpha_i^{lm} \geq 0$ ,  $\phi_i^{lm} \geq 0$  and  $m, l = 1, \dots, M$ .

Equation (4.22) can be written as

$$\mathbf{y}_t = \beta_0 + \sum_{i=1}^k B_i \mathbf{y}_{t-i} + \sum_{i=1}^r \Psi_i \mathbf{x}_{t-i} + \mathbf{u}_t = \boldsymbol{\mu}_t + \mathbf{u}_t, \quad (4.25)$$

where  $\mathbf{y}_t = (y_{t1}, \dots, y_{tM})'$  is an  $M \times 1$  vector of observed time series at time  $t$ ,  $\beta_i$  ( $i = 1, \dots, k$ ) and  $\Psi_i$  ( $i = 1, \dots, s$ ) are fixed  $M \times M$  and  $M \times M^x$  coefficient matrices, respectively,  $\beta_0 = (\beta_{10}, \dots, \beta_{M0})$  is a fixed  $M \times 1$  vector of intercept terms, and  $\mathbf{u}_t = (u_{t1}, \dots, u_{tM})^T$  is an  $M \times 1$  vector of error terms. The Bayesian VAR(k)-GARCH(p,q)-X(s) model is then given by

$$\mathbf{Y} \sim N_{T \times M}[\mathbf{B}\tilde{\mathbf{Y}} + \Psi\tilde{\mathbf{X}}, \text{diag}(\mathbf{H}_1, \dots, \mathbf{H}_T)],$$

$$vech\mathbf{H}_t = \boldsymbol{\alpha}_0 + \sum_{i=1}^q \boldsymbol{\alpha}_i vech(\mathbf{u}_{t-i} \mathbf{u}'_{t-i}) + \sum_{j=1}^p \boldsymbol{\Phi}_j vech\mathbf{H}_{t-j},$$

where

$$\begin{aligned} \mathbf{B} &= [\boldsymbol{\beta}_0, \boldsymbol{\beta}_1, \dots, \boldsymbol{\beta}_k]_{(M \times (Mk+1))}, & \boldsymbol{\Psi} &= [\boldsymbol{\Psi}_1, \dots, \boldsymbol{\Psi}_s]_{(M \times M^{x_s})}, \\ \tilde{\mathbf{Y}} &= (\tilde{\mathbf{Y}}_0, \dots, \tilde{\mathbf{Y}}_{T-1})_{(1+Mk) \times T}, & \tilde{\mathbf{X}} &= (\tilde{\mathbf{X}}_0, \dots, \tilde{\mathbf{X}}_{T-1})_{(M^{x_s} \times T)} \end{aligned}$$

and the prior distribution are chosen from the families of normal distributions, hence

$$\mathbf{B} \sim N_{M \times (1+Mk)}[\mathbf{B}_*, \boldsymbol{\Sigma}_{B_*} \otimes \mathbf{I}_M],$$

$$\boldsymbol{\Psi} \sim N_{M \times M^{x_s}}[\boldsymbol{\Psi}_*, \boldsymbol{\Sigma}_{\Psi_*} \otimes \mathbf{I}_M],$$

where all of the hyper-parameters (which are denoted with a star) are known a priori.

### 4.3.1 The full conditional distributions (f.c.d.)

a) The f.c.d. for the regression coefficients  $\mathbf{B}$

The full conditional density for  $\mathbf{B}$  is a multivariate normal distribution

$$p(\mathbf{B} | \mathbf{Y}, \theta^c) = N_{M \times (1+Mk)}[\mathbf{B}_{**}, \mathbf{D}_{B_{**}}]$$

with

$$\begin{aligned} \mathbf{D}_{B_{**}}^{-1} &= \mathbf{I}_M \otimes \boldsymbol{\Sigma}_{B_*}^{-1} + \langle \mathbf{x}'_t \mathbf{H}_t^{-1} \mathbf{x}_t \rangle, \\ \mathbf{B}_{**} &= \mathbf{D}_{B_{**}} [\text{vec}(\boldsymbol{\Sigma}_{B_*}^{-1} \mathbf{B}_* + \langle \mathbf{x}'_t \mathbf{H}_t^{-1} \tilde{\mathbf{y}}_t \rangle)], \end{aligned}$$

where  $\tilde{\mathbf{Y}} = \mathbf{Y} - \boldsymbol{\Psi} \tilde{\mathbf{X}}$  and  $\theta^c = (\boldsymbol{\Psi}, \mathbf{A}, \boldsymbol{\Phi})$  denotes a vector of all parameters save the arguments of the full conditional distribution.

b) The f.c.d. for the regression coefficients  $\boldsymbol{\Psi}$

The f.c.d. is given by

$$p(\boldsymbol{\Psi} | \mathbf{Y}, \theta^c) = N_{M \times M^{x_s}}[\boldsymbol{\Psi}_{**}, \mathbf{D}_{\Psi_{**}}]$$

with

$$\begin{aligned} \mathbf{D}_{\Psi_{**}}^{-1} &= \mathbf{I}_M \otimes \boldsymbol{\Sigma}_{\Psi_*}^{-1} + \langle \mathbf{x}'_t \mathbf{H}_t^{-1} \mathbf{x}_t \rangle, \\ \boldsymbol{\Psi}_{**} &= \mathbf{D}_{\Psi_{**}} [\text{vec}(\boldsymbol{\Sigma}_{\Psi_*}^{-1} \boldsymbol{\Psi}_* + \langle \mathbf{x}'_t \mathbf{H}_t^{-1} \tilde{\mathbf{y}}_t \rangle)], \end{aligned}$$

and  $\tilde{\mathbf{Y}} = \mathbf{Y} - \mathbf{B}\mathbf{X}$ .

c) The f.c.d. for the GARCH coefficients

We use for the f.c.d. of  $\boldsymbol{\alpha}$  and  $\boldsymbol{\psi}$  a Metropolis-within-Gibbs step with a normal distribution which is obtained by an iteration proposal given by

$$vec\boldsymbol{\alpha}_i \sim N[vec\hat{\boldsymbol{\alpha}}_i, \hat{\boldsymbol{\Sigma}}_{\boldsymbol{\alpha}_i}],$$

$$vec\boldsymbol{\Phi}_i \sim N[vec\hat{\boldsymbol{\Phi}}_i, \hat{\boldsymbol{\Sigma}}_{\boldsymbol{\Phi}_i}],$$

and the f.c.d. is given by

$$p(\mathbf{A}, \Phi | \mathbf{Y}, \theta^c) = \prod_{t=1}^T N[\mathbf{y}_t | \boldsymbol{\mu}_t, \mathbf{H}_t]$$

with  $\boldsymbol{\mu}_t$  given in (4.28) and the normal distribution being proportional to

$$N[\mathbf{y}_t | \boldsymbol{\mu}_t, \mathbf{H}_t] \propto |\mathbf{H}_t|^{-1/2} \exp\left\{-\frac{1}{2}(\mathbf{y}_t - \boldsymbol{\mu}_t)' \mathbf{H}_t^{-1} (\mathbf{y}_t - \boldsymbol{\mu}_t)\right\}.$$

**Note:** If  $\mathbf{H} = \text{diag}(\mathbf{H}_1, \dots, \mathbf{H}_T)$  is a  $TM \times TM$ ,  $\mathbf{W}$  a  $r \times T$ , and  $\mathbf{V}$  a  $T \times k$  matrix, then

$$\begin{aligned} \langle \mathbf{w}_i \mathbf{H}_t \mathbf{v}_{tj} \rangle &= (\mathbf{W} \otimes \mathbf{I}_M) \text{diag}(\mathbf{H}_1, \dots, \mathbf{H}_T) (\mathbf{V} \otimes \mathbf{I}_M) \\ &= \begin{pmatrix} \sum_t \mathbf{w}_{1t} \mathbf{H}_t \mathbf{v}_{t1}, \dots, \sum_t \mathbf{w}_{1t} \mathbf{H}_t \mathbf{v}_{tk} \\ \dots \\ \sum_t \mathbf{w}_{rt} \mathbf{H}_t \mathbf{v}_{t1}, \dots, \sum_t \mathbf{w}_{rt} \mathbf{H}_t \mathbf{v}_{tk} \end{pmatrix} \\ &= \langle \mathbf{w}_i \mathbf{H}_t \mathbf{v}_i \rangle_{rM \times kM}, \end{aligned}$$

where  $\otimes$  denotes the Kronecker product.

The starting values and the variance matrix of the proposal distribution for the Metropolis algorithm are usually obtained from the posterior density function. We start with arbitrary values for  $\mathbf{A}_{old}$  (e.g.,  $\text{vec} \mathbf{A}_{old} = 0.1 \mathbf{1}_{1+M_p+M_q}$ ) and its variance (e.g., identity matrix  $\mathbf{D} = 0.01 \mathbf{I}_{1+M_p+M_q}$ ). We update the starting values and the proposed variances for each run from the output of the MCMC.

Then, we calculate the mean and variance and repeat the iterative proposal until a satisfactory accept/reject ratio is obtained.

For each run, we have 1000 iterations and 100 burn-in simulation points. We collect samples from selected iterations of the converged chains (iteration 2901 to 3000) to ensure that we have independent and identically distributed samples from the joint posterior density.

#### d) Forecasting

You can use the forecasted variance matrix for October 1999 for portfolio selection in the three regions. The function *weights* will compute weights that produce the minimum-variance portfolio according to the forecasted variance matrix. This function simply calculates  $w_t = \frac{\hat{H}_{t+1}^{-1} \mathbf{1}}{\mathbf{1}' \hat{H}_{t+1}^{-1} \mathbf{1}}$ . You have to define a matrix Q as the forecasted variance matrix. The command *weights(Q)* will then give the calculated weights.

### 4.3.2 Program description

An S-plus function for calculating the VAR-GARCH-X model has been written. It is called *varxgarch*. Users can find the program on our home page.

The *varxgarch* function needs a matrix or multivariate time series as its input data file, which must be a S-plus object. You must also input the arguments as follows:

- 1)  $M$ : The dimension of the time series, which can be chosen from 1 to 10.
- 2)  $k$ : The order of the VAR process,  $k \geq 0$  is an integer.
- 3)  $s$ : The order of the X component,  $s \geq 0$  is an integer.

- 4)  $p$ : The order of the GARCH process,  $p \geq 0$  is an integer.
- 5)  $q$ : The order of the ARCH process,  $q \geq 0$  is an integer.
- 6)  $iter$ : The number of iterations,  $iter \geq 1$  is an integer.
- 7)  $Dalpha$ : A  $(1 + Mp + Mq) \times (1 + Mp + Mq)$  matrix of  $\Sigma_{\alpha}$  and  $\Sigma_{\Phi}$ .
- 8)  $alpha$ : A  $(1 + Mp + Mq) \times M$  matrix of  $\hat{\alpha}$  and  $\hat{\Phi}$ .
- 9)  $Dbstar$ : A  $(1 + Mk) \times (1 + Mk)$  matrix of  $\Sigma_{B_*}$  and  $\Sigma_{\Psi_*}$ .
- 10)  $Bstar$ : A  $(1 + Mk + Ms) \times M$  matrix of  $B_*$  and  $\Psi_*$ .
- 11)  $show.para$ : is a logical value. If one chooses  $T$ , the program will print out the mean and standard error of the simulated parameters of the model for all iterations.
- 12)  $show.para.hist$ : is a logical variable. If one chooses  $T$ , the program will show the histogram (graphic output) of the simulated parameters of the model for all iterations.
- 13)  $show.para.Markov$ : is a logical variable. If one chooses  $T$ , the program will plot a graph, which plots the Markov chains (graphic output) of the simulated parameters of the model for all iterations.
- 14)  $show.epsilon.plot$ : is a logical variable. If one chooses  $T$ , the program will plot a graph, which plots the mean of the residuals of the model for all iterations.
- 15)  $show.epsilon$ : is a logical variable. If one chooses  $T$ , the program will print out the mean and standard error of the model's residuals of iterations for all iterations.
- 16)  $show.ht.plot$ : is a logical variable. If one chooses  $T$ , the program will plot a graph, which plots the mean of the conditional variance of the model for all iterations.
- 17)  $show.ht$ : is a logical variable. If one chooses  $T$ , the program will print out the mean and standard error of the model's conditional variance for all iterations.
- 18)  $nstep$ : is an integer value, which is the number of the forecasting period: if  $nstep = 1$ , the program will do one step forecasting calculation.
- 19)  $show.fore$ : is a logical variable. If one chooses  $T$ , the program will show the histogram and mean of the every simulated forecasting period for all iterations.
- 20)  $libpath$ : is the path to the Fortran module
- 21) Finally, if one runs the program, the logarithmic marginal likelihood of the model will be automatically given.

Example:

`my.model<-vargarch(data=MS.re3)`. You fit a VARX(1)-GARCH(1,1) model for the monthly returns of the MSCI North America index, MSCI Europe index and MSCI Pacific index from February 1990 until September 1999 and store it as an object `my.model` with the command above. By default, you get the estimated coefficients and the std. error, one step forecast of the series and of the variance matrix and the marginal likelihood. The output is split into two parts. Part one gives the Gibbs samples, part two gives the means of the sample. By default only part two will be given. The command `names(my.model)` will give

```
> names(my.model)
[1] "output.part1" "output.part2"
```

Typing

```
my.model$output.part2
```

you will get the following output:

```
my.model$output.part2 $Model: [1] "VAR(1)-GARCH(1,1)-X(1) Model"
```

```
$coefficients:
```

	Mean	Std
[1,]	-0.04639208	0.24365633
[2,]	0.00799629	0.18623870
[3,]	-0.01478842	0.18727769
[4,]	0.00945650	0.15311956
[5,]	0.03520719	0.14390165
[6,]	-0.00356363	0.17176715
[7,]	0.01188296	0.15302321
[8,]	0.00509588	0.00231339
[9,]	0.04816601	0.02881839
[10,]	0.03164130	0.01654791
[11,]	0.12309285	0.06982707
[12,]	0.19664578	0.08944997
[13,]	0.09745164	0.05235588
[14,]	0.07827449	0.03407900
[15,]	-0.02138672	0.16771385
[16,]	0.04572156	0.17711741
[17,]	-0.00120196	0.16956592
[18,]	0.01629033	0.15052923
[19,]	-0.04418842	0.20832086
[20,]	-0.06804941	0.16494083
[21,]	-0.03898779	0.17979390
[22,]	0.00342045	0.00105053
[23,]	0.08637602	0.03262938
[24,]	0.04380256	0.03191680
[25,]	0.13363104	0.03534384
[26,]	0.07779944	0.05842045
[27,]	0.23184077	0.08913161
[28,]	0.10598686	0.04718707
[29,]	0.00410501	0.16055658
[30,]	-0.01829825	0.19061967
[31,]	0.03581969	0.19284898
[32,]	-0.02890539	0.19627849
[33,]	-0.00669833	0.14842075
[34,]	0.03420267	0.19563794
[35,]	-0.00339707	0.17238935
[36,]	0.00402546	0.00182443

```

[37,]0.11430713    0.06613368
[38,]0.07985261    0.05063823
[39,]0.13547708    0.04337831
[40,]0.15827104    0.08086459
[41,]0.29391002    0.07446812
[42,]0.06408933    0.03883201

$log.marginal.likelihood: [1] -324.214

$forecast.series:
      fore.mean fore.std
[1,] -0.02687  0.03237 [2,] -0.00897  0.04137 [3,]0.05134
0.06622

$forecast.H: $forecast.H[[1]]:
           [,1]      [,2]      [,3]
[1,] 0.006039633    0.000930624 0.000927021 [2,]0.000930624
0.004390372 0.000963293 [3,]0.000927021    0.000963293
0.006593739

```

A forecast can be made with the following command:

```
my.model<-Q<-my.model$output.part2$forecast.H
```

The output is then given as:

```

> weights(Q,1,mylist)
      north america  europe  pacific
[1,]      0.297523  0.436564  0.265913

```

## 4.4 The VAR( $k$ )-GARCH( $p,q$ )-M( $r$ ) models

The VAR( $k$ )-GARCH( $p,q$ )-M( $r$ ) model proposed by Pelloni and Polasek (1998) can be written in following way:

$$y_t^l = \beta_0^l + \sum_{m=1}^M \sum_{i=1}^k \beta_i^{lm} y_{t-i}^m + \sum_{m=1}^M \sum_{i=1}^r \psi_i^{lm} h_{t-i}^m + u_{l,t} \quad (4.26)$$

$$u_{l,t} \sim N[0, h_t^l], \quad l = 1, \dots, M,$$

or

$$y_t^l \sim N[\mathbf{x}_t \boldsymbol{\beta}, h_t^l],$$

$$h_t^l = \alpha_0^l + \sum_{m=1}^M \left( \sum_{i=1}^p \alpha_i^{lm} h_{t-i}^m + \sum_{i=1}^q \phi_i^{lm} u_{m,t-i}^2 \right), \quad (4.27)$$

where the parameters are

$$\sum_{m=1}^M \left( \sum_{i=1}^p \alpha_i^{lm} + \sum_{i=1}^q \phi_i^{lm} \right) < 1,$$

with all coefficients being positive:  $\alpha_0^{lm} > 0$ ,  $\alpha_i^{lm} \geq 0$ ,  $\phi_i^{lm} \geq 0$  and  $m, l = 1, \dots, M$ .

Equation (4.26) can be written as

$$\mathbf{y}_t = \boldsymbol{\beta}_0 + \sum_{i=1}^k \boldsymbol{\beta}_i \mathbf{y}_{t-i} + \sum_{i=1}^r \boldsymbol{\Psi}_i \text{vech} \mathbf{H}_{t-i} + \mathbf{u}_t = \boldsymbol{\mu}_t + \mathbf{u}_t, \quad (4.28)$$

where  $\mathbf{y}_t = (y_{t1}, \dots, y_{tM})'$  is an  $M \times 1$  vector of observed time series at time  $t$ ,  $\boldsymbol{\beta}_i$  ( $i = 1, \dots, k$ ) and  $\boldsymbol{\Psi}_i$  ( $i = 1, \dots, r$ ) are fixed  $M \times M$  coefficient matrices,  $\boldsymbol{\beta}_0 = (\beta_{10}, \dots, \beta_{M0})$  is a fixed  $M \times 1$  vector of intercept terms, and  $\mathbf{u}_t = (u_{t1}, \dots, u_{tM})^T$  is an  $M \times 1$  vector of error terms.

The above model is rewritten as a multivariate regression system

$$\mathbf{Y} = \mathbf{B}\mathbf{X} + \boldsymbol{\Psi}\tilde{\mathbf{H}} + \mathbf{U}, \quad (4.29)$$

where

$$\begin{aligned} \mathbf{B} &= [\boldsymbol{\beta}_0, \boldsymbol{\beta}_1, \dots, \boldsymbol{\beta}_k]_{(M \times (\tilde{M}k+1))}, & \boldsymbol{\Psi} &= [\boldsymbol{\Psi}_1, \dots, \boldsymbol{\Psi}_r]_{(M \times \tilde{M}r)}, \\ \mathbf{X} &= (\mathbf{X}_0, \dots, \mathbf{X}_{T-1})_{T \times (1+\tilde{M}k)}, & \tilde{\mathbf{H}} &= (\tilde{\mathbf{H}}_0, \dots, \tilde{\mathbf{H}}_{T-1})_{(\tilde{M}r \times T)}, \\ \mathbf{X}_t &= \begin{pmatrix} 1 \\ \mathbf{y}_t \\ \cdot \\ \cdot \\ \cdot \\ \mathbf{y}_{t-k+1} \end{pmatrix}, & \tilde{\mathbf{h}}_t &= \begin{pmatrix} \text{vech} \mathbf{H}_t \\ \cdot \\ \cdot \\ \cdot \\ \text{vech} \mathbf{H}_{t-r+1} \end{pmatrix}, \end{aligned}$$

and  $\tilde{M} = M(M+1)/2$ .

We now show that the model under study has convenient conditional structure to apply the Gibbs sampler.

The Bayesian VAR( $k$ )-GARCH( $p, q$ )-M( $r$ ) model is then given by

$$\begin{aligned} \mathbf{Y} &\sim N_{T \times M}[\mathbf{B}\mathbf{X} + \boldsymbol{\Psi}\tilde{\mathbf{H}}, \text{diag}(\mathbf{H}_1, \dots, \mathbf{H}_T)], \\ \text{vech} \mathbf{H}_t &= \boldsymbol{\alpha}_0 + \sum_{i=1}^q \boldsymbol{\alpha}_i \text{vech}(\mathbf{u}_{t-i} \mathbf{u}'_{t-i}) + \sum_{j=1}^p \boldsymbol{\Phi}_j \text{vech} \mathbf{H}_{t-j}, \end{aligned}$$

and the prior distributions are chosen from the families of normal distributions, hence

$$\begin{aligned} \mathbf{B} &\sim N_{M \times (1+\tilde{M}k)}[\mathbf{B}_*, \boldsymbol{\Sigma}_{B_*} \otimes \mathbf{I}_M], \\ \boldsymbol{\Psi} &\sim N_{M \times \tilde{M}r}[\boldsymbol{\Psi}_*, \boldsymbol{\Sigma}_{\Psi_*} \otimes \mathbf{I}_M], \end{aligned}$$

where all of the hyper-parameters (which are denoted with a star) are known a priori. Thus, we can derive the following full conditional distributions (f.c.d.) for the Gibbs simulation process.

The joint distribution for the data  $\mathbf{Y}$  and the parameters  $\theta = (\mathbf{B}, \Psi, \mathbf{A}, \Phi)$  is with  $\mathbf{A} = (\alpha_0, \alpha_1, \dots, \alpha_q)$  and  $\Phi = (\phi_0, \phi_1, \dots, \phi_p)$

$$\begin{aligned} p(\theta, \mathbf{Y}) &= N[\mathbf{Y}|\mathbf{B}\mathbf{X} + \Psi\tilde{\mathbf{H}}, \text{diag}(\mathbf{H}_1, \dots, \mathbf{H}_T)] \\ &\cdot N[\mathbf{B}|\mathbf{B}_*, \Sigma_{B_*} \otimes \mathbf{I}_M] \cdot N[\Psi|\Psi_*, \Sigma_{\Psi_*} \otimes \mathbf{I}_M] \\ &\cdot \prod_{i=0}^p N[\alpha_i|\alpha_i^*, \Sigma_{\alpha_i}] \cdot \prod_{i=1}^q N[\Phi_i|\Phi_i^*, \Sigma_{\Phi_i}]. \end{aligned}$$

#### 4.4.1 The full conditional distributions (f.c.d.)

a) The f.c.d. for the regression coefficients  $\mathbf{B}$

The full conditional density for  $\mathbf{B}$  is a multivariate normal distribution

$$p(\mathbf{B}|\mathbf{Y}, \theta^c) = N_{M \times (1+\tilde{M}k)}[\mathbf{B}_{**}, \mathbf{D}_{B_{**}}]$$

with

$$\begin{aligned} \mathbf{D}_{B_{**}}^{-1} &= \mathbf{I}_M \otimes \Sigma_{B_*}^{-1} + \langle \mathbf{x}'_t \mathbf{H}_t^{-1} \mathbf{x}_t \rangle, \\ \mathbf{B}_{**} &= \mathbf{D}_{B_{**}} [\text{vec}(\Sigma_{B_*}^{-1} \mathbf{B}_* + \langle \mathbf{x}'_t \mathbf{H}_t^{-1} \tilde{\mathbf{y}}_t \rangle)], \end{aligned}$$

where  $\tilde{\mathbf{Y}} = \mathbf{Y} - \Psi\tilde{\mathbf{H}}$  and  $\theta^c = (\Psi, \mathbf{A}, \Phi)$  denotes a vector of all parameters save the arguments of the full conditional distribution.

b) The f.c.d. for the regression coefficients  $\Psi$

The f.c.d. is given by

$$p(\Psi|\mathbf{Y}, \theta^c) = N_{M \times \tilde{M}r}[\Psi_{**}, \mathbf{D}_{\Psi_{**}}]$$

with

$$\begin{aligned} \mathbf{D}_{\Psi_{**}}^{-1} &= \mathbf{I}_M \otimes \Sigma_{\Psi_*}^{-1} + \langle \mathbf{x}'_t \mathbf{H}_t^{-1} \mathbf{x}_t \rangle, \\ \Psi_{**} &= \mathbf{D}_{\Psi_{**}} [\text{vec}(\Sigma_{\Psi_*}^{-1} \Psi_* + \langle \mathbf{x}'_t \mathbf{H}_t^{-1} \tilde{\mathbf{y}}_t \rangle)], \end{aligned}$$

and  $\tilde{\mathbf{Y}} = \mathbf{Y} - \mathbf{B}\mathbf{X}$ .

c) The f.c.d. for the GARCH coefficients

We use for the f.c.d. of  $\alpha$  and  $\psi$  a Metropolis-within-Gibbs step with a normal distribution which is obtained by an iteration proposal given by

$$\text{vec}\alpha_i \sim N[\text{vec}\hat{\alpha}_i, \hat{\Sigma}_{\alpha_i}],$$

$$\text{vec}\Phi_i \sim N[\text{vec}\hat{\Phi}_i, \hat{\Sigma}_{\Phi_i}],$$

and the f.c.d. is given by

$$p(\mathbf{A}, \Phi|\mathbf{Y}, \theta^c) = \pi(\mathbf{A}, \Phi) \prod_{t=1}^T N[\mathbf{y}_t|\boldsymbol{\mu}_t, \mathbf{H}_t]$$

with  $\boldsymbol{\mu}_t$  given in (4.28) and the normal distribution being proportional to

$$N[\mathbf{y}_t | \boldsymbol{\mu}_t, \mathbf{H}_t] \propto |\mathbf{H}_t|^{-1/2} \exp\left\{-\frac{1}{2}(\mathbf{y}_t - \boldsymbol{\mu}_t)' \mathbf{H}_t^{-1} (\mathbf{y}_t - \boldsymbol{\mu}_t)\right\}.$$

**Note:** If  $\mathbf{H} = \text{diag}(\mathbf{H}_1, \dots, \mathbf{H}_T)$  is a  $TM \times TM$ ,  $\mathbf{W}$  a  $r \times T$ , and  $\mathbf{V}$  a  $T \times k$  matrix, then

$$\begin{aligned} \langle \mathbf{w}_{it} \mathbf{H}_t \mathbf{v}_{tj} \rangle &= (\mathbf{W} \otimes \mathbf{I}_M) \text{diag}(\mathbf{H}_1, \dots, \mathbf{H}_T) (\mathbf{V} \otimes \mathbf{I}_M) \\ &= \begin{pmatrix} \sum_t \mathbf{w}_{1t} \mathbf{H}_t \mathbf{v}_{t1}, \dots, \sum_t \mathbf{w}_{1t} \mathbf{H}_t \mathbf{v}_{tk} \\ \dots \\ \sum_t \mathbf{w}_{rt} \mathbf{H}_t \mathbf{v}_{t1}, \dots, \sum_t \mathbf{w}_{rt} \mathbf{H}_t \mathbf{v}_{tk} \end{pmatrix} \\ &= \langle \mathbf{w}_t \mathbf{H}_t \mathbf{v}_t \rangle_{rM \times kM}, \end{aligned}$$

where  $\otimes$  denotes the Kronecker product. The multivariate ARCH model can be written as

$$\begin{aligned} \mathbf{Y} &\sim N[\mathbf{0}, \text{diag}(\mathbf{H}_1, \dots, \mathbf{H}_T)], \\ \mathbf{H}_t &= \alpha_0 + \sum_{j=1}^p \alpha_j \mathbf{y}_{t-j} \mathbf{y}_{t-j}', \end{aligned}$$

where  $\mathbf{Y} = (\mathbf{y}_1, \dots, \mathbf{y}_T)'$ ,  $\mathbf{y}_i = (y_{1i}, \dots, y_{Mi})$ ,  $\boldsymbol{\alpha}_i = (\alpha_1^i, \dots, \alpha_M^i)$ ,  $i = 0, 1, \dots, p$ .

To implement the MCMC, we use Metropolis draws for each component in an ARCH( $p$ ) model, where the parameters are given by the  $M \times (p+1)$  parameter matrix  $\mathbf{A} = (\alpha_0, \boldsymbol{\alpha}_1, \dots, \boldsymbol{\alpha}_p)$ , and the posterior distribution of  $\mathbf{A}$  is given by:

$$\begin{aligned} \pi(\mathbf{A} | \mathbf{Y}) &\propto \pi(\mathbf{A}) \prod_{t=1}^T N[\mathbf{y}_t | \mathbf{0}, \mathbf{H}_t] \\ &= \pi(\mathbf{A}) \prod_{t=1}^T |\mathbf{H}_t|^{-1/2} \exp\left(-\frac{1}{2} \mathbf{y}_t' \mathbf{H}_t^{-1} \mathbf{y}_t\right). \end{aligned}$$

We specify a prior multivariate normal distribution for  $\mathbf{A}$  where  $\mathbf{A}_*$  and  $\mathbf{H}_*$  are a  $M \times (p+1)$  matrix and a  $M(p+1) \times M(p+1)$  diagonal variance matrix, respectively. Then the  $\pi(\mathbf{A})$  is given by

$$\pi(\mathbf{A}) = N[\mathbf{A}_*, \mathbf{H}_*],$$

or

$$\text{vec} \mathbf{A} = N[\text{vec} \mathbf{A}_*, \mathbf{H}_* = \mathbf{D}_* \otimes \mathbf{I}_M],$$

where we suggest as a default specification  $\text{vec} \mathbf{A}_* = \varepsilon \mathbf{1}_{M(p+1)}$ ,  $\mathbf{D}_* = \varepsilon^2 \mathbf{I}_{p+1}$  and  $\varepsilon = 0.1$ .

Let  $\mathbf{A}_{old}$  denote the current value of  $\mathbf{A}$ : we draw  $\mathbf{A}_{new}$  from the multivariate normal distribution which has mean  $\mathbf{A}_{old}$  and variance matrix  $\mathbf{D} \otimes \mathbf{I}_M$ , or say  $\text{vec} \mathbf{A}_{new} \sim N[\text{vec} \mathbf{A}_{old}, \mathbf{D} \otimes \mathbf{I}_M]$ . Then we calculate the acceptance probability ratio of the complete conditional density  $\alpha(\mathbf{A}_{new}, \mathbf{A}_{old}) = \pi(\mathbf{A}_{old} | \mathbf{Y}) / \pi(\mathbf{A}_{new} | \mathbf{Y})$ . If  $\alpha \geq 1$  we move to  $\mathbf{A}_{new}$ ; if  $\alpha < 1$  we move to  $\mathbf{A}_{old}$  with probability  $\alpha$ . The stationary distribution of this Markov chain is the normalized density associated with complete conditional distribution for  $\mathbf{A}$  given everything else (Hastings, 1970, Tierney, 1994). After updating all  $\mathbf{A}$ , complete one iteration of the modified Gibbs sampler.

The starting values and the variance matrix of the proposal distribution for the Metropolis algorithm are usually obtained from the prior density function. We start with arbitrary values for  $\mathbf{A}_{old}$  (e.g.,

$vec\mathbf{A}_{old}=0.11\mathbf{I}_{M(p+1)}$ ) and its variance (e.g., identity matrix  $\mathbf{D} = 0.01\mathbf{I}_{p+1}$ ). We update the starting values and the proposed variances for each run from the output of the MCMC.

Then we calculate the mean and variance and repeat the iterative proposal until a satisfactory accept/reject ratio is obtained.

For each run, we have 1000 iterations and 100 burn-in simulation points. We collect samples from selected iterations of the converged chains (iterations 2901 to 3000) to ensure that we have independent and identically distributed samples ( $M = 100$ ) from the joint posterior density. The multivariate VAR-GARCH-M model with orders  $k, s, p$  and  $q$  and  $M$  dimensions can be written in a random coefficient form (see Polasek and Pelloni (1997)):

$$\begin{aligned} \mathbf{y}_t^l &= \beta_0^l + \sum_{m=1}^M \sum_{i=1}^k \beta_i^{lm} \mathbf{y}_{t-i}^m + \sum_{m=1}^M \sum_{i=1}^r \psi_i^{lm} h_{t-i}^m + u_{l,t} \\ u_{l,t} &\sim N[0, h_t^l], \quad l = 1, \dots, M, \end{aligned} \quad (4.30)$$

or

$$\begin{aligned} \mathbf{y}_t^l &\sim N[\mathbf{x}_t \boldsymbol{\beta}, h_t^l], \\ h_t^l &= \alpha_0^l + \sum_{m=1}^M \left( \sum_{i=1}^p \alpha_i^{lm} h_{t-i}^m + \sum_{i=1}^q \phi_i^{lm} u_{m,t-i}^2 \right), \end{aligned}$$

where the parameters are

$$\sum_{m=1}^M \left( \sum_{i=1}^p \alpha_i^{lm} + \sum_{i=1}^q \phi_i^{lm} \right) < 1,$$

with all coefficients being positive:  $\alpha_0^l > 0$ ,  $\alpha_i^{lm} \geq 0$ ,  $\phi_i^{lm} \geq 0$  and  $m, l = 1, \dots, M$ .

The equation (4.30) can be written as

$$\mathbf{y}_t = \boldsymbol{\beta}_0 + \sum_{i=1}^k \boldsymbol{\beta}_i \mathbf{y}_{t-i} + \sum_{i=1}^r \boldsymbol{\Psi}_i vech \mathbf{H}_{t-i} + \mathbf{u}_t = \boldsymbol{\mu}_t + \mathbf{u}_t,$$

where  $\mathbf{y}_t = (y_{t1}, \dots, y_{tM})'$  is an  $M \times 1$  vector of an observed time series at time  $t$ ,  $\boldsymbol{\beta}_i$  ( $i = 1, \dots, k$ ) and  $\boldsymbol{\Psi}_i$  ( $i = 1, \dots, r$ ) are fixed  $M \times M$  coefficient matrices,  $\boldsymbol{\beta}_0 = (\beta_{10}, \dots, \beta_{M0})$  is a fixed  $M \times 1$  vector of intercept terms, and  $\mathbf{u}_t = (u_{t1}, \dots, u_{tM})^T$  is an  $M \times 1$  vector of error terms.

The prior distributions for  $\mathbf{B} = [\boldsymbol{\beta}_0, \boldsymbol{\beta}_1, \dots, \boldsymbol{\beta}_k]$  and  $\boldsymbol{\Psi} = [\boldsymbol{\Psi}_1, \dots, \boldsymbol{\Psi}_r]$  are given by

$$\mathbf{B} \sim N[\mathbf{B}_*, \boldsymbol{\Sigma}_{B_*} \otimes \mathbf{I}_M],$$

$$\boldsymbol{\Psi} \sim N[\boldsymbol{\Psi}_*, \boldsymbol{\Sigma}_{\Psi_*} \otimes \mathbf{I}_M].$$

where  $\mathbf{B}_*$ ,  $\boldsymbol{\Sigma}_{B_*}$ ,  $\boldsymbol{\Psi}_*$  and  $\boldsymbol{\Sigma}_{\Psi_*}$  are known priors.

#### 4.4.2 Program description

A S-plus function for calculating the VAR-GARCH-M model has been written. It is called *fvargarchm*. The *fvargarchm* function needs a matrix or multivariate time series as its input data file, which must be a S-plus object. You must also input the arguments as follows:

- 1)  $M$ : The dimension of the time series, which can be chosen from 1 to 10.
- 2)  $k$ : The order of the VAR process,  $k \geq 0$  is an integer.

- 3)  $s$ : The order of the ARCH-M component,  $s \geq 0$  is an integer.
- 4)  $p$ : The order of the GARCH process,  $p \geq 0$  is an integer.
- 5)  $q$ : The order of the ARCH process,  $q \geq 0$  is an integer.
- 6)  $iter$ : The number of iterations,  $iter \geq 1$  is an integer.
- 7)  $Dalpha$ : A  $(1 + Mp + Mq) \times (1 + Mp + Mq)$  matrix of  $\Sigma_\alpha$  and  $\Sigma_\Phi$ .
- 8)  $alpha$ : A  $(1 + Mp + Mq) \times M$  matrix of  $\hat{\alpha}$  and  $\hat{\Phi}$ .
- 9)  $Dbstar$ : A  $(1 + Mk) \times (1 + Mk)$  matrix of  $\Sigma_{\mathbf{B}_*}$  and  $\Sigma_{\Psi_*}$ .
- 10)  $Bstar$ : A  $(1 + Mk + Ms) \times M$  matrix of  $\mathbf{B}_*$  and  $\Psi_*$ .

If the 10 variables from above have been given, you can run the program.

The program will calculate and print out the results as follows:

- 11)  $show.para$ : is a logical value. If one chooses  $TRUE(T)$ , the program will print out the mean and standard error of the simulated parameters of the model for all iterations.
- 12)  $show.para.hist$ : is a logical variable. If one chooses  $TRUE(T)$ , the program will show the histogram (graphic output) of the simulated parameters of the model for all iterations.
- 13)  $show.para.Markov$ : is a logical variable. If one chooses  $TRUE(T)$ , the program will plot a graph, which plots the Markov chains (graphic output) of the simulated parameters of the model for all iterations.
- 14)  $show.epsilon.plot$ : is a logical variable. If one chooses  $TRUE(T)$ , the program will plot a graph, which plots the mean of the residuals of the model for all iterations.
- 15)  $show.epsilon$ : is a logical variable. If one chooses  $TRUE(T)$ , the program will print out the mean and standard error of the model's residuals of iterations for all iterations.
- 16)  $show.ht.plot$ : is a logical variable. If one chooses  $TRUE(T)$ , the program will plot a graph, which plots the mean of the conditional variance of the model for all iterations.
- 17)  $show.ht$ : is a logical variable. If one chooses  $TRUE(T)$ , the program will print out the mean and standard error of the model's conditional variance for all iterations.
- 18)  $nstep$ : is an integer value, which is the number of the forecasting period: if  $nstep = 1$ , the program will do one step forecasting calculation.
- 19)  $show.fore$ : is a logical variable. If one chooses  $TRUE(T)$ , the program will show the histogram and mean of the every simulated forecasting period for all iterations.
- 20)  $libpath$ : is the path to the Fortran module
- 21) The log marginal likelihood of the model is calculated by.

Example:

`my.model<-fvargarchm(data=MS.re3)`. You fit a VAR(1)-GARCH(1,1)-M(1) model for the monthly returns of the MSCI North America index, MSCI Europe index and MSCI Pacific index from February 1990 until September 1999 and store it as an object 'my.model' with the command above. By default, you get the estimated coefficients and the std. error, one step forecast of the series and of the variance matrix and the marginal likelihood. The output is split into two parts. Part one gives the Gibbs samples, part two gives the means of the sample. By default only part two will be given. The command `names(my.model)` will give

```
names(my.model)
[1] "output.part1" "output.part2"
```

Typing

```
my.model$output.part2
```

you will get the following output:

```
$Model: [1] "VAR(1)-GARCH(1,1)-M(1) Model"
```

```
$coefficients:
```

	Mean	Std			
[1,]	0.013485513	0.006284278			
[2,]	0.003064259	0.086630172			
[3,]	0.036685832	0.174901129			
[4,]	-0.059270490	0.136803514			
[5,]	-0.031963095	0.165552835			
[6,]	0.027212965	0.143505640			
[7,]	0.049633252	0.213099840			
[8,]	0.000833391	0.001603078			
[9,]	0.168323853	0.079932432			
[10,]	0.081767315	0.049285046	[11,]	0.101733481	0.031424569
[12,]	0.179370046	0.060187511	[13,]	0.049417530	0.020816559
[14,]	0.033533471	0.015547867	[15,]	-0.045572357	0.144357593
[16,]	-0.039856606	0.203045100	[17,]	0.009349925	0.007449610
[18,]	-0.030015250	0.119003220	[19,]	0.000192899	0.182524809
[20,]	-0.205320361	0.153228739	[21,]	-0.012117073	0.206305578
[22,]	0.000929699	0.000909161	[23,]	0.102489362	0.047644806
[24,]	0.028945929	0.017836806	[25,]	0.099812546	0.032796172
[26,]	0.023219638	0.011958072	[27,]	0.110264420	0.020305975
[28,]	0.033549289	0.015632484	[29,]	-0.020536128	0.128428800
[30,]	-0.002829660	0.207918956	[31,]	0.024146202	0.141895570
[32,]	-0.001722513	0.178746298	[33,]	-0.002262065	0.009306896
[34,]	0.055064726	0.100259920	[35,]	0.033950258	0.172481342
[36,]	0.002488143	0.003794953	[37,]	0.130811236	0.045656642
[38,]	0.161962901	0.054273436	[39,]	0.042916893	0.020247630
[40,]	0.043190106	0.030889274	[41,]	0.044742502	0.016874215
[42,]	0.068175628	0.026764010			

```
$log.marginal.likelihood: [1] -323.989
```

```
$forecast.mean:
```

	fore.mean	fore.std			
[1,]	-0.02700	0.03176	[2,]	-0.00959	0.03725
			[3,]		0.03945
	0.06569				

```
$forecast.H: $forecast.H[[1]]:
              [,1]      [,2]      [,3]
[1,] 0.001623878 0.000258133 0.000279569 [2,]0.000258133
0.001891076 0.000302078 [3,]0.000279569 0.000302078 0.005219453
```

Comparing the log marginal likelihoods we can see the the VAR(1)-GARCH(1,1)-M(1) provides a better fit for the monthly returns of the MSCI North America, MSCI Europe and MSCI Pacific indices than the VARX(1)-GARCH(1,1) model.

You can also fit a VAR-GARCH model with the function 'fvargarch'.

# Chapter 5

## Miscellaneous Models

### 5.1 Unit root test with marginal likelihood

Suppose that a  $p$ th-order autoregressive model  $AR(p)$  satisfies the following:

$$y_t = c + \phi_1 y_{t-1} + \dots + \phi_p y_{t-p} + \varepsilon_t, \quad (5.1)$$

where  $\varepsilon_t$  is white noise with mean zero and variance  $\sigma^2$ .

For the  $AR(p)$  process assume that the stationarity condition ( $E(y_t) = \mu$  for all  $t$ , and  $E(y_t - \mu)(y_{t-j} - \mu) = \gamma_j$  for all  $t$  and any  $j$ ) is satisfied. If the time series  $\{y_t\}$  is  $I(1)$  ( $I(d)$  integrated of order  $d$ ),  $y_t \sim I(1)$  comes from calculus: if  $dy/dt = x$ , then  $y$  is the integral of  $x$ . If  $y_t \sim I(0)$ , then  $dy/dt = 0$ ), i.e.  $y_t$  is non-stationary or has a unit root.

In this section we outline how the classical augmented Dickey and Fuller (1979) regression models for unit roots can be used to calculate the marginal likelihoods for a Bayes test.

#### 5.1.1 The $AR(p)$ model in first differences (non-stationary model)

If the time series  $y_t$  is  $I(1)$ , then the first differences should be a non-stationary (i.e. that it has a unit root)  $AR(p)$  process. Define the first differences  $z_t = y_t - y_{t-1}$  and estimate a non-stationary  $AR(p)$  process ( $\Delta$ - $AR$ ) for  $p = 1, \dots, p_{max}$ :

$$z_t = \alpha_1 z_{t-1} + \dots + \alpha_p z_{t-p} + u_t, \quad t = 1, \dots, T. \quad (5.2)$$

$z_0, \dots, z_{p+1}$  are the first observations of the time series on which the analysis is conditioned.

#### 5.1.2 The simple ADF model (stationary model)

If the  $y_t$  series is stationary (no unit root) we could estimate an  $AR(p)$  model for the  $y_t$ . Such a model cannot be compared with (5.2), therefore we need a model reformulation which can be compared

to (5.2). This is the so-called augmented Dickey-Fuller (ADF) regression model for first differences (abbreviated as DF-AR 1):

$$z_t = \alpha_0 y_{t-1} + \alpha_1 z_{t-1} + \dots + \alpha_p z_{t-p} + u_t, \quad (5.3)$$

which is a reformulation of a stationary AR( $p$ ) model for the  $y_t$ .

### 5.1.3 The ADF model with mean (stationary model)

We extend the simple ADF model (5.3) by the intercept  $\mu$ .

The model is the so-called Dickey-Fuller (DF) regression model with mean (DF-AR 2):

$$z_t = \mu + \alpha_0 y_{t-1} + \alpha_1 z_{t-1} + \dots + \alpha_p z_{t-p} + u_t. \quad (5.4)$$

### 5.1.4 The ADF model with mean and trend (stationary model)

We extend the ADF model with an intercept and a trend component. The model is the so-called Dickey-Fuller (DF) regression model with mean and trend (DF-AR 3):

$$z_t = \mu + \beta t + \alpha_0 y_{t-1} + \alpha_1 z_{t-1} + \dots + \alpha_p z_{t-p} + u_t. \quad (5.5)$$

### 5.1.5 Marginal likelihoods for the AR( $p$ ) model with fractional prior

Since the above models are very similar, we prove only model (5.2) (see, for details, Polasek and Ren (1997)). We consider the linear autoregressive model

$$y_t = \beta_0 + \beta_1 y_{t-1} + \dots + \beta_p y_{t-p} + u_t \quad (5.6)$$

with a  $T \times k$  regression matrix  $\mathbf{X}$  of full rank, where  $k = p + 1$ , conditional on  $p$  starting values.

$$\mathbf{y} \sim N[\mathbf{X}\beta, \sigma^2 \mathbf{I}_T] \quad (5.7)$$

with a non-informative prior distribution for the parameter  $\theta = (\beta, \sigma^2)$ ,

$$f(\beta, \sigma^2) \propto \frac{1}{\sigma^2}, \quad (5.8)$$

then the marginal likelihood is given by

$$f(\mathbf{y}) = |\mathbf{X}'\mathbf{X}|^{-\frac{1}{2}} \cdot (\pi \mathbf{ESS})^{-\frac{T-k}{2}} \Gamma\left(\frac{T-k}{2}\right) / 2. \quad (5.9)$$

**Lemma a** The fractional marginal likelihood with  $b \in (0, 1)$  and  $t = 1$  is given by

$$f_b^1(\mathbf{y}) = b^{\frac{Tb-k}{2}} (\pi ESS)^{-\frac{T-Tb}{2}} \Gamma\left(\frac{T-k}{2}\right) / \Gamma\left(\frac{Tb-k}{2}\right). \quad (5.10)$$

If  $b = \frac{k}{T}$ , then  $\frac{Tb-k}{2} = 0$  and

$$f_b^1(\mathbf{y}) = (\pi ESS)^{-\frac{T-k}{2}} \Gamma\left(\frac{T-k}{2}\right). \quad (5.11)$$

### 5.1.6 Module usage

The unit root test (stationary test) function is called *unit.test* and requires a vector or time series as its input data file. Finally, if one runs the program, the logarithmic marginal likelihood of the model from  $p = 1, \dots, pmax$  will be automatically given.

**Function Name:**

`unit.test`

**Usage:**

`unit.test(data=MS[1,], pmax=3)`

**Required Arguments:**

**data:** a univariate time series or a vector.

**Optional Arguments:**

**pmax:** the order of the AR process.  $pmax \geq 1$  is an integer. Default: 3.

**name:** name of the data.

**Example:**

`my.unitr <- unit.test(MS[1,], 3)`. `my.unitr <- unit.test(MS.re[1,], 3)`. Note: MSre is simply a time series of returns on the initial index series, MS. This program operates by estimating the marginal likelihoods for a non-stationary AR(p) process, given in the first column as Delta-AR, and three different stationary AR(p) processes: simple ADF, ADF with mean and ADF with mean and trend. We select as most likely the model with the highest marginal likelihood. In the case of the first example below, we see that the greatest marginal likelihood is calculated for the Delta-AR(1) process and therefore can suspect that we are looking at a nonstationary process. This is not surprising, given that the data-set in question is a stock index. The second example shows that the greatest marginal likelihood is calculated for the DF-AR 2 model, which is the model with mean, with  $p = 3$ . This suggests a stationary model and, given that we are looking at returns on the stock index, this is again not surprising.

```
> my.model <- unit.test(MS[,1], p=3)
```

log. marginal likelihood

p	Delta-AR	DF-AR 1	DF-AR 2	DF-AR 3
1	-577.445	-579.7783	-581.2975	-584.7247
2	-578.5228	-581.0594	-582.5581	-585.9839
3	-579.6876	-582.2377	-583.6971	-587.0785

```
> my.model <- unit.test(MSCI[,1], p=3)
```

log. marginal likelihood

p	Delta-AR	DF-AR 1	DF-AR 2	DF-AR 3
1	185.9693	205.3876	207.8186	202.7129
2	190.3984	204.3139	207.1082	201.9815
3	191.9068	205.3203	208.9449	204.8261

# Chapter 6

## Appendix

### 6.0.7 References

Albert, J.; Chib, S. (1993): Bayesian inference of autoregressive time series with mean and variance subject to Markov jumps, *Journal of Business and Economic Statistics* 11, 1-15

Bollersev, T.; Chou, R.Y.; Kroner, K.F. (1992): ARCH modeling in finance, *Journal of Econometrics* 52, 5-59.

Chib, F. (1993): Bayesian regression with autoregressive errors: A Gibbs sampling approach, *Journal of Econometrics* 58, 275-294

Engle, R.F. (1982): Autoregressive conditional heteroskedasticity with estimates of the variance of U.K. inflation, *Econometrica* 50, 987-1008

Gelfand, A.E.; Smith, A.F.M. (1990): Sampling based approaches to calculating marginal densities, *Journal of the American Statistical Association* 85, 398-409

Gelfand, A.E.; Dey, D.K. (1992): Bayesian model choice: Asymptotics and exact calculation, University of Connecticut, to appear in *JRSSB*

Gelfand, A.E.; Dey, D.K.; Chang, H. (1991): Model Determination Using Predictive Distributions with Implementations via Sampling-Based Methods, *Bayesian Statistics 4*, Valencia

Gelman, A.; Rubin, D.B. (1992): Inference from iterative simulation using multiple sequences, *Statistical Science* 7, 63-86

Geweke, J. (1989): Exact predictive densities in linear models with ARCH disturbances, *Journal of Econometrics* 44, 307-325

- Geweke, J. (1989): Bayesian inference in econometric models using Monte Carlo integration, *Econometrica* 57, 1317-1339
- Luetkepohl, H. (1993): Introduction to multiple time series analysis, 2nd ed., Springer Verlag, Berlin
- Marriott, J.; Ravishanker, J.; Gelfand, A.E.; Pai, J. (1992): Bayesian analysis of ARIMA processes: Complete sampling based inference under full likelihood, mimeo, University of Connecticut
- McCulloch, R.E.; Tsay, R.S. (1994): Bayesian analysis of autoregressive time series via Gibbs sampler, *Journal of Time series Analysis* 15, 235-250
- Thomas, A.; Spiegelhalter, D.J.; Gilks, W.R. (1992): Bugs: A program to perform Bayesian Inference using Gibbs sampling, in: *Bayesian Statistics 4*, Oxford University Press, 837-842
- Tsay, R.S. (1987): Conditional heteroskedastic time series models, *Journal of the American Statistical Association* 82, 590-604
- Zellner, A.; Chung-ki Min (1995): Gibbs sampler convergence Criteria (GSC2), *JASA*, Sep. 95