CODA *
Convergence Diagnosis and Output Analysis Software for Gibbs sampling output
Version 0.30

Nicky Best† **Mary Kathryn Cowles**‡

Karen Vines†

†*MRC Biostatistics Unit, Institute of Public Health*
Robinson Way, Cambridge CB2 2SR, UK
Tel: 44-1223-330300, *Fax: 44-1223-330388*
e-mail: bugs@mrc-bsu.cam.ac.uk, *ftp: ftp.mrc-bsu.cam.ac.uk*

‡*University of Nebraska Medical Center, USA*

August 14, 1996

Permission and Disclaimer

Authorship note

CODA was conceived and motivated by Cowles (1994), who developed the original program as part of her PhD thesis concerning practical issues in implementing the Gibbs sampler. It has been modified by the first author of this manual (NGB) to provide software suitable for general release.

 The support of the Economic and Social Research Council (UK) is gratefully acknowledged. The work was funded in part by ESRC (UK) Award Number H519 25 5023. We are also grateful to Brad Carlin for many helpful comments and ideas concerning the CODA software and manual, and to Steve Brooks for suggesting the graphical implementations of the Geweke and Gelman & Rubin convergence diagnostics

Contents

1 Introduction

1.1 What is CODA?

CODA is a menu-driven set of S-Plus[1] functions which serves as an output processor for the BUGS[2] (Bayesian inference Using Gibbs Sampling) software. For a Bayesian analysis specified by the user, BUGS carries out Gibbs sampling and writes the requested series of simulated values ("chains") to an ASCII text file. CODA may then be used to perform convergence diagnostics and statistical and graphical output analysis of the contents of these files. Text output from CODA is displayed on the screen and recorded in a file called CODA.LOG, which may be printed after exiting from CODA. Hard copies of the graphical output may be obtained directly by using the "print" option in the graphics device window, or by storing the image as a postscript file for future use. CODA may also be used in conjunction with Gibbs sampling output from users' own programs. Section §7 describes the file format required for such output to be read into CODA.

1.2 A cautionary note on convergence diagnostics

Before proceeding further with CODA, all users should be aware that none of the convergence diagnostics implemented in this package (or indeed, anywhere else) are foolproof. Cowles and Carlin (1995) tested a number of these methods, and found examples when each failed to detect lack of convergence. We recommend using a combination of diagnostics plus visual inspection of the trace plots and summary statistics generated by CODA. In this way, the user should be able to assess the convergence of the BUGS output with a reasonable degree of confidence, whilst recognising that it is not possible to say with certainty that the sample is representative of the underlying stationary distribution.

1.3 Routines implemented in CODA version 0.30

CODA version 0.30 implements the following graphical analyses, summary statistics and convergence diagnostic tests:

Graphical analyses

- Plots of the sample trace for each variable in each chain
- Plots of the kernel density estimate for each variable
- Plots of the autocorrelation function for each variable in each chain
- Plots of the cross-correlations between variables
- Plots of Geweke (1992)'s diagnostic
- Plots of Gelman and Rubin (1992)'s diagnostic

[1]S-Plus ©1988, 1993 Statistical Sciences, Inc., Seattle, Washington, USA
[2]BUGS ©1994 MRC Biostatistics Unit, Cambridge, UK

Summary statistics

- Empirical means, standard deviations and quantiles
- Standard error of the mean using:
  - naive methods (assuming independent samples)
  - time-series methods to account for within-chain correlations
  - batch means method to account for within-chain correlations

Convergence diagnostics

- Geweke (1992)
- Gelman and Rubin (1992)
- Raftery and Lewis (1992*b*)
- Heidelberger and Welch (1983)
- Autocorrelations for each variable in each chain
- Cross-correlations between variables

1.4 Hardware/Software requirements

`CODA` runs under S-Plus version 3.0 and above, and is suitable for PC and UNIX environments.

1.5 Referring to `CODA`

This manual may be referred to as Best, N.G., Cowles, M.K. and Vines, S.K. (1995). *CODA Manual version 0.30.* MRC Biostatistics Unit, Cambridge, UK.

2 Getting started

2.1 Obtaining `CODA`

`CODA` may be obtained by anonymous ftp from `ftp.mrc-bsu.cam.ac.uk`, and is in the directory `pub/methodology/bugs`. The `BUGS` software and documentation are also available in this ftp directory. Alternatively, `CODA` may be obtained on a disk from the authors (NGB at MRC Biostatistics Unit). The program is supplied as an ASCII text file called `coda03.SPARC` or `coda03.pc` for UNIX and DOS versions respectively.

2.2 Installation

2.2.1 Within *UNIX*

Set up a `/CODA` directory and move the `coda03.SPARC` file here. Then create a `/CODA/.Data` directory. Invoke S-Plus from the `/CODA` directory and install the `CODA` functions using the S-Plus command:

```
source("coda03.SPARC")
```

If you then wish to run `CODA` from a different directory (e.g. from the directory in which you have stored the output from a particular `BUGS` analysis), create a `.Data` sub-directory for this directory. Then invoke S-Plus and use the `attach()` function to load `CODA`. For example, if you have installed `CODA` in a directory with path `/usr/fred.bloggs/CODA`, then type:

```
attach("/usr/fred.bloggs/CODA/.Data")
```

from within S-Plus. You may also create a `.First` function containing this command, e.g.

```
.First <- function() {
                attach("/usr/fred.bloggs/CODA/.Data")
          }
```

This will cause `CODA` to be loaded automatically every time S-Plus is invoked from that particular directory.

2.2.2 Within *DOS*

Set up a \\`CODA` directory and move the `coda03.pc` file here. Then create a \\`CODA`\\`_DATA` directory. Invoke S-Plus from the \\`CODA` directory and install the `CODA` functions using the S-Plus command:

```
source("coda03.pc")
```

If you then wish to run `CODA` from a different directory (e.g. from the directory in which you have stored the output from a particular `BUGS` analysis), create a _DATA sub-directory for this directory. Then invoke S-Plus and use the `attach()` function to load `CODA`. For example:

```
attach("C:/CODA/_Data")
```

from within S-Plus. (Note the use of the forward slash "/" when specifying path names within S-Plus, rather than the usual DOS backslash "\\"). You may also create a `.First` function containing this command, e.g.

```
.First <- function() {
                attach("C:/CODA/_DATA")
          }
```

This will cause `CODA` to be loaded automatically every time S-Plus is invoked from that particular directory.

2.3 Testing the installation

First check that all the CODA functions have been loaded by listing the contents of the CODA/.Data (or CODA/_DATA if in *DOS*) directory. This should contain the following 56 files:

```
CODA              draw.trace          inddat                 parmcorr
autocorr          gandr               invlogit               parmcorr.plot
autocorr.plot     gandr.shrink.plot   kernel.width           pause
calcbatch2        gandr.trace.plot    log.var                plot.defaults
call.plots        geweke              logit                  plot.to.file
change.defaults   geweke.cd           logit.var              pretty.print
chisqdf           geweke.nse          main                   ps.file.options
col.means         geweke.plot         my.format              randl2
col.vars          geweke.power        my.string.bbox         readdat
cov               gibbsit2            my.string.break.line   setup.plots
diag.defaults     gpar                open.graphics          stats.defaults
diags             gpar.log            outanal                test.bandwidth
display.defaults  gpar.logit          outplots               tidy.up
draw.dens         heidel              outstats               working.data
```

Then move to the directory where you have installed the line example supplied with the BUGS software, and carry out the BUGS runs described below in order to generate a set of test data for use with CODA.

2.3.1 Generating BUGS output for testing CODA

BUGS run 1

First edit the line.in file supplied with BUGS as follows:

```
list(tau = 5, alpha = -5, beta = 5, seed = 9876543210)
```

Note the inclusion of a random number seed in this file.

Carry out the following BUGS analysis using the line.bug and line.dat files supplied with BUGS, and the new line.in file which you have just created. (If you have saved the latter under a different name, such as line1.in, make sure that you also edit the name of the file which appears after the inits in ... statement in the line.bug file):

```
Bugs>compile("line.bug")
model line;
const
   N = 5;  # number of observations
var
   x[N],Y[N],mu[N],alpha,beta,tau,sigma,x.bar;
data in  "line.dat";
inits in "line.in";
{
    for (i in 1:N) {
          mu[i] <- alpha + beta*(x[i] - x.bar);
          Y[i]   ~ dnorm(mu[i],tau);
    }
    x.bar   <- mean(x[]);
    alpha    ~ dnorm(0.0,1.0E-4);
    beta     ~ dnorm(0.0,1.0E-4);
    tau      ~ dgamma(1.0E-3,1.0E-3);
    sigma   <- 1.0/sqrt(tau);

 }
Bugs>monitor(alpha)
Bugs>monitor(beta)
Bugs>monitor(sigma)
Bugs>update(200)
Bugs>q()
```

Now rename the `bugs.out` and `bugs.ind` files to `line1.out` and `line1.ind` respectively to ensure they don't get over-written after the next BUGS run.

BUGS run 2

Re-edit the file `line.in` to change the initial values and random number seed as follows:

```
list(tau = 0.01, alpha = 0.01, beta = 0.01, seed = 1234567890)
```

Now re-run the line example in BUGS for 200 iterations using the new initial values file.

Having completed the second BUGS run, rename the `bugs.out` and `bugs.ind` files to `line2.out` and `line2.ind` respectively.

You have now generated 2 parallel runs of the `line` problem in preparation for multiple chain analysis in CODA.

2.3.2 Using CODA to analyse the line output

Create a `.Data` (or `_DATA` if in *DOS*) sub-directory of the directory where the `line1.out`,... file *etc.* are located. Invoke S-Plus and attach the CODA functions as described in §2.2. To run CODA, type

```
CODA()
```

at the S-Plus prompt. You will receive a welcome message on the screen and a menu of options:

```
 ---------------------------------------------------------------
|                                                               |
|                     Welcome to CODA!                          |
|  Convergence Diagnostics and Output Analysis for BUGS output  |
|_______________________________________________________________|
|                                                               |
|  Authors : Nicky Best, Kate Cowles & Karen Vines.             |
|                                                               |
|  CODA : Copyright (c) 1995 MRC Biostatistics Unit.            |
|  All rights reserved                                          |
|  Version 0.30                                                 |
|                                                               |
|_______________________________________________________________|

Do you wish to:

1: Begin a new CODA session using BUGS output files
2: Begin a new CODA session using data saved from a previous CODA session
3: Quit
Selection:
```

Select option 1 by typing `1` after the word `Selection:`. You will be prompted to enter the names of the `BUGS .ind` and `.out` files that you wish to analyse. To specify the files created by the 2 chain `line` analysis, respond as follows (user-responses are shown in bold):

Enter BUGS output filenames, separated by return key (Leave blank to exit):
1: **line1**
2: **line2**
3:

Note that the filename extensions are not required.

Next you will be prompted to enter a title for the analysis. This is optional and may be left blank:

Enter problem title, followed by return key:
1: **Line problem from the BUGS manual**

The program then reports on its progress in reading the specified files.

```
Reading Data file...
Abstracting beta ... 200 valid values
Abstracting alpha ... 200 valid values
Abstracting sigma ... 200 valid values
Reading Data file...
Abstracting beta ... 200 valid values
Abstracting alpha ... 200 valid values
Abstracting sigma ... 200 valid values
```

Next, you will be prompted to specify which (if any) variables take values restricted to either the range (0, 1) or to the positive real line. `CODA` requires this information in order to correctly compute Gelman and Rubin (1992)'s convergence diagnostic for non-normal variables (see §4.2), and to produce kernel density estimates within the appropriate range (see §3.1).

```
Are any variables restricted to values between 0 and 1 (y/n) ?
1:
```

For the `line` example, you should respond **n** to this question. The next prompt to appear is as follows:

```
Are any variables restricted to all positive values (y/n) ?
1:
```

For the `line` example, you should respond **y** to this question, which causes the following display to appear:

```
Available variables:

+---------------+-------------+
|VARIABLE NUMBER|VARIABLE NAME|
+---------------+-------------+
|1              |alpha        |
+---------------+-------------+
|2              |beta         |
+---------------+-------------+
|3              |sigma        |
+---------------+-------------+

Enter relevant variable number(s), separated by commas
(Ranges such as 3:7 may be specified)
(Enter 0 for none):
1:
```

The `sigma` parameter represents the standard deviation of the residual error in the `line` model, and so is restricted to positive values. Therefore, you should enter the number **3** at the above prompt.

After a slight pause whilst `CODA` creates a 'working' copy of the data, the main menu is then displayed.

```
CODA Main Menu
**************

1: Output Analysis
2: Diagnostics
3: List/Change Defaults
4: Quit
Selection:
```

The following sections of this manual describe the various menu options in detail, and give examples of the output obtained when each option is applied to the `line` example.

3 Output Analysis

Selecting option 1 of the `CODA Main Menu` calls up the `CODA Output Analysis Menu`:

```
CODA Output Analysis Menu
*************************

1: Plots
2: Statistics
3: List/Change Defaults
4: Return to Main Menu
Selection:
```

3.1 Plots

Option 1 of the `CODA Output Analysis Menu` produces a graphical summary of the iterates for each monitored variable and chain in the `BUGS` output. By default, 2 plots are produced for each variable. The first shows the traces from each chain as separate time series on the same graph. The second shows a kernel density estimate calculated by combining the samples of each variable from all chains. Figure 1 shows this output for the `line` example.

Note that the S-Plus `density` function used by `CODA` to produce these plots yields estimates whose support extends the range of the sampled values by 0.75 times the bandwidth (see §5.5.4 for further details on bandwidth specification). This can result in negative estimates for variables such as variance parameters, whose sampled values must be positive but may be close to zero. `CODA` avoids this problem by using the reflection method described in Silverman (1986) pp.30–31 to calculate kernel density estimates for variables specified to be positive or restricted to the range (0,1).

The the following menu will appear on screen the first time graphical output is requested during a `CODA` session:

```
Select graphics device required:

1: openlook (UNIX)
2: motif (UNIX)
3: X11 (UNIX)
4: win.graph (DOS)
Selection:
```

The user should respond by typing the number of the required graphics device after the prompt. UNIX users may choose any of the first 3 devices, whilst DOS users should select the fourth option. (Please contact the authors if other options are needed). Note that `CODA` occasionally crashes when trying to open a graphics window. This problem only happens when the `BUGS` input file is very large, and is due to the way S-Plus uses up memory during a session. One solution

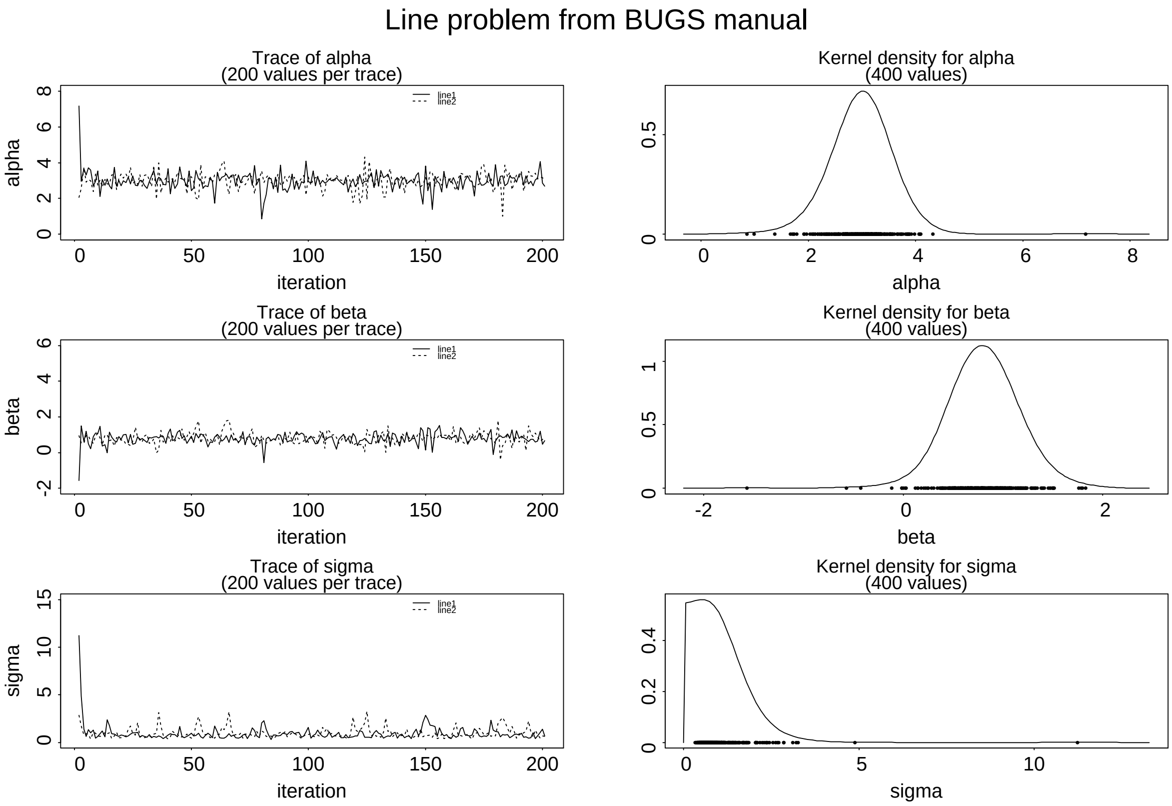

Figure 1: Graphical summary of the output from the `line` example produced by selecting `Plots` (Option 1) from the `CODA Output Analysis Menu`

may be to first read the `BUGS` files into `CODA` and then immediately quit and save these as in S-Plus format. Re-start `CODA` and select option 2 ('`Begin a new CODA session using data saved from a previous CODA session`') to read the data directly in S-Plus format. This should leave sufficient memory available to then open a graphics window and proceed with the `CODA` session.

In `CODA`, kernel density estimation is carried out using the S-Plus `density` function. A Gaussian kernel is used with default window width given by 0.25 times the range of the sampled values for each variable. This leads to a smooth estimate, but may hide local features of the density. Other bandwidth functions may be used if desired (see §5.5.4).

The user may also change certain other plotting options in order to customize the `CODA` output produced. Possible alternative options include:

- plotting only traces or only kernel density estimates
- adding a smooth line through the trace plots to graphically assess convergence
- producing a separate trace and/or kernel density plot for each chain

  ....

Such changes may be invoked by selecting option 3 (`List/Change Defaults`) of the `CODA Output Analysis Menu`. The user is referred to §5 for further details.

3.1.1 Printing graphical output from CODA

Once CODA has produced a plot in the graphics window, the following prompt will appear in the text window:

```
Do you want to save current plots as a postscript file (y/n) ?
1:
```

If you answer **y** to this question, you will be prompted to enter a name for the file. This causes the contents of the graphics window to be saved as a postscript file. This file may be printed at a later date or included in any text documents (such as Latex) which can import encapsulated postscript objects. §5.5.5 provides details of the page size and orientation formats available for these postscript files. Alternatively, you may want a hard copy of the plot, but not require the postscript file for later use. In this case, respond **n** to the above question and simply click on the print button in the graphics window to send the image directly to the printer. (Note that UNIX users who wish to save graphical output from CODA as postscript files will find that the `openlook` and `motif` devices produce postscript output with a larger font size than does the `X11` device).

3.1.2 Multiple pages of plots

If more plots are requested than will fit in a single graphics window, the following prompt will then appear in the text window:

```
Plots span more than 1 page
Type 0 to continue scrolling:
```

This pause enables the user to activate the print button (see above) if required, before the next set of plots appear in the window. Typing **0** after the prompt will cause CODA to overwrite the current graphics image and display the next series of plots.

Once all plots have been displayed, the following prompt appears in the text window:

```
Finished displaying requested plots
Do you want to display the first page of plots again (y/n)?
1:
```

If you wish to re-display the first page of plots again (for example, if you didn't print the image the first time round and want to do so now), respond **y** to this question. Otherwise, respond **n** and the `CODA Output Analysis Menu` will re-appear on screen.

3.1.3 Displaying multiple graphics windows on screen

You may wish to keep a copy of a particular plot displayed on the screen, perhaps for later comparison with another CODA plot. Whenever a new plot is requested, CODA automatically over-writes the image currently displayed in the graphics window. However, your graphics device *may* have a COPY option under the GRAPH menu in the top left of the graphics window. If so, you can click on this option to produce a second copy of the current graphics window which will not be over-written by the new image about to be produced.

3.2 Statistics

Option 2 of the `CODA Output Analysis Menu` produces summary statistics of the iterates for each monitored variable and chain in the `BUGS` output. By default, 2 tables are produced for each chain. The first table gives the empirical (sample) mean and standard deviation for each variable, plus the standard error of the mean. The empirical standard deviation estimates the square root of the variance of the posterior distribution, whilst the standard error provides a measure of the precision of the sample mean as a point estimate for the true posterior mean. This precision depends on the number of iterates and the degree of autocorrelation within the sample. Three separate estimates of the standard error are provided: a naive estimate (the empirical standard deviation divided by the square root of the number of iterates) which assumes the sampled values to be independent; a time-series estimate which gives the asymptotic standard error (the square root of the spectral density estimate divided by the sample size); and a batch estimate which involves dividing the sample into consecutive batches of size 25 and calculating the square root of the variance of the batch means divided by the number of batches. The lag-1 autocorrelation between the batch means is also reported — this should be close to zero if approximate independence between the batches has been achieved. Otherwise, the batch size should be increased (see §5.6.2 for how to do this).

The second table reports various empirical quantiles for each variable. By default, these are the 2.5%, 25%, 50%, 75% and 97.5% quantiles (although other quantiles may be selected — see §5.6.3).

Selecting `Statistics` (option 2:) from the `CODA Output Analysis Menu` produces the following output for the `line` example:

```
SUMMARY STATISTICS
==================

Iterations used = 1:200
Thinning interval = 1
Sample size per chain = 200

Batch size for calculating Batch SE = 25

1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:

Chain: line1
============

-+----------+------------------------------------------------+-
 | VARIABLE |   Mean      SD        Naive SE  Time-series SE   |
 | ======== |   ====      ==        ========  ==============   |
 |          |                                                  |
 | alpha    |    2.98000   0.53100   0.03760   0.02890         |
 | beta     |    0.78700   0.34100   0.02410   0.01740         |
 | sigma    |    0.95400   0.88900   0.06290   0.04300         |
-+----------+------------------------------------------------+-
```

```
-+----------+------------------------------------+-
 | VARIABLE |   Batch SE  Lag-1 batch autocorr   |
 | ======== |   ========  ====================   |
 |          |                                    |
 | alpha    |    0.04470  -0.26900               |
 | beta     |    0.02340   0.00411               |
 | sigma    |    0.09390  -0.23200               |
-+----------+------------------------------------+-
```

```
Chain: line2
============
```

```
-+----------+----------------------------------------------------------+-
 | VARIABLE |   Mean     SD       Naive SE  Time-series SE  Batch SE   |
 | ======== |   ====     ==       ========  ==============  ========   |
 |          |                                                          |
 | alpha    |    2.9900   0.4640   0.0328    0.0277          0.0253    |
 | beta     |    0.8120   0.3330   0.0236    0.0266          0.0371    |
 | sigma    |    0.9820   0.5570   0.0394    0.0548          0.0555    |
-+----------+----------------------------------------------------------+-
-+----------+--------------------------+-
 | VARIABLE |   Lag-1 batch autocorr   |
 | ======== |   ====================   |
 |          |                          |
 | alpha    |    0.0223                |
 | beta     |   -0.3850                |
 | sigma    |   -0.6060                |
-+----------+--------------------------+-
```

```
2. Quantiles for each variable:

Chain: line1
============
```

```
-+----------+-----------------------------------------------+-
 | VARIABLE |   2.5%     25%      50%      75%      97.5%    |
 | ======== |   ====     ===      ===      ===      =====    |
 |          |                                               |
 | alpha    |   2.090    2.730    2.970    3.230    3.840    |
 | beta     |   0.141    0.626    0.792    0.986    1.380    |
 | sigma    |   0.425    0.610    0.778    1.030    2.360    |
-+----------+-----------------------------------------------+-
```

```
Chain: line2
============

-+----------+---------------------------------------------------+-
 | VARIABLE |   2.5%     25%      50%      75%      97.5%       |
 | ======== |   ====     ===      ===      ===      =====       |
 |          |                                                   |
 | alpha    |   1.970    2.760    3.060    3.260    3.890       |
 | beta     |   0.150    0.596    0.801    0.995    1.510       |
 | sigma    |   0.428    0.631    0.819    1.090    2.680       |
-+----------+---------------------------------------------------+-
```

4 Convergence Diagnostics

Selecting option 2 of the `CODA Main Menu` calls up the `CODA Diagnostics Menu`:

```
CODA Diagnostics Menu
*********************

1: Geweke
2: Gelman & Rubin
3: Raftery & Lewis
4: Heidelberger and Welch
5: Autocorrelations
6: Cross-Correlations
7: List/Change Defaults
8: Return to Main Menu
Selection:
```

This is not an exhaustive list of the methods developed for diagnosing convergence of Gibbs samplers and other Markov chain Monte Carlo sequences. Rather, the methods available in `CODA` represent a selection of the most popular and easily implemented approaches. We do not recommend any particular method as being superior, but suggest that users try a combination of diagnostics when checking for convergence. A brief summary of the theory and interpretation of each diagnostic is given below, and readers are referred to recent reviews by Cowles and Carlin (1995) and Brooks and Roberts (1995), plus the individual references to each method for further details.

4.1 Geweke

Geweke (1992) proposes a convergence diagnostic based on standard time-series methods. The test is appropriate for use with single chains when convergence of the *mean* of some function of the sampled variables is of interest. For each (function of the) variable, the chain is divided into 2 "windows" containing the first $x\%$ (`CODA` default is 10%) and the last $y\%$ (`CODA` default is 50%) of the iterates. If the whole chain is stationary, the means of the values early and late in the sequence should be similar. Geweke's approach involves calculation of the sample mean and asymptotic variance in each window, the latter being determined by spectral density estimation.

His convergence diagnostic Z is the difference between these 2 means divided by the asymptotic standard error of their difference. As the chain length $\rightarrow \infty$, the sampling distribution of $Z \rightarrow N(0,1)$ if the chain has converged. Hence values of Z which fall in the extreme tails of a standard normal distribution suggest that the chain was not fully converged early on (i.e. during the 1st window).

Selecting `Geweke` (option 1:) from the `CODA Diagnostics Menu` produces the following output for the `line` example:

```
GEWEKE CONVERGENCE DIAGNOSTIC (Z-score):
========================================

Iterations used = 1:200
Thinning interval = 1
Sample size per chain = 200

Fraction in 1st window = 0.1
Fraction in 2nd window = 0.5


-+----------+---------------------+-
 | VARIABLE |   line1    line2    |
 | ======== |   =====    =====    |
 |          |                     |
 | alpha    |    4.540   -0.488   |
 | beta     |   -2.370   -1.100   |
 | sigma    |    5.270   -0.680   |
-+----------+---------------------+-
```

The results for chain `line2` provide no evidence *against* convergence for each variable (although the fact that the Z-scores could be reasonably thought to arise from an N(0, 1) distribution does not *prove* convergence). However, the values of Z for each monitored variable in chain `line1` are rather extreme, suggesting that the first 10% of the samples do not arise from the same distribution as do the last 50%. By discarding the first 20 (10%) iterations of chain `line1` and re-computing Geweke's diagnostic, we could next test whether there is any evidence that iterations 21–200 have not converged. If the resulting Z-scores were still extreme, a further 10% of iterations could be discarded and so on.

4.1.1 Plotting Geweke's diagnostic

You will then be given the option of producing plots to illustrate these diagnostics:

```
Geweke Plots Menu
*****************

1: Plot Z-scores
2: Return to Diagnostics Menu
Selection:
```

Selecting option 1 will generate plots like those shown in Figure 2. These plots are produced by splitting the entire chain for each variable into a number of segments. For a chain of length N, the first segment contains all N samples in the chain; the second contains the last $N - n$ samples (i.e. iterations $n + 1$–N); the third contains samples from iterations $2n + 1$–N and so on. The final segment is chosen such that it contains at least the last 50 samples. The default bin size is n=10 for chains of 500 iterations or less. For longer chains, the bin size is determined by splitting the chain equally into 50 bins. However these sizes may be changed by the user (see §5.7.2). Geweke's diagnostic is computed for each segment, and the resulting Z-scores are plotted against the number of the first iteration in the segment. Horizontal dotted lines at $Z = \pm 1.96$ are added to the plot to indicate the 95% confidence interval for an N(0, 1) distribution. A large number of Z-scores falling outside this interval suggests possible convergence failure. These plots may take some time to produce due to the large number of computations invovled: the process may be speeded up by reducing the total number of bins, or increasing the bin size.

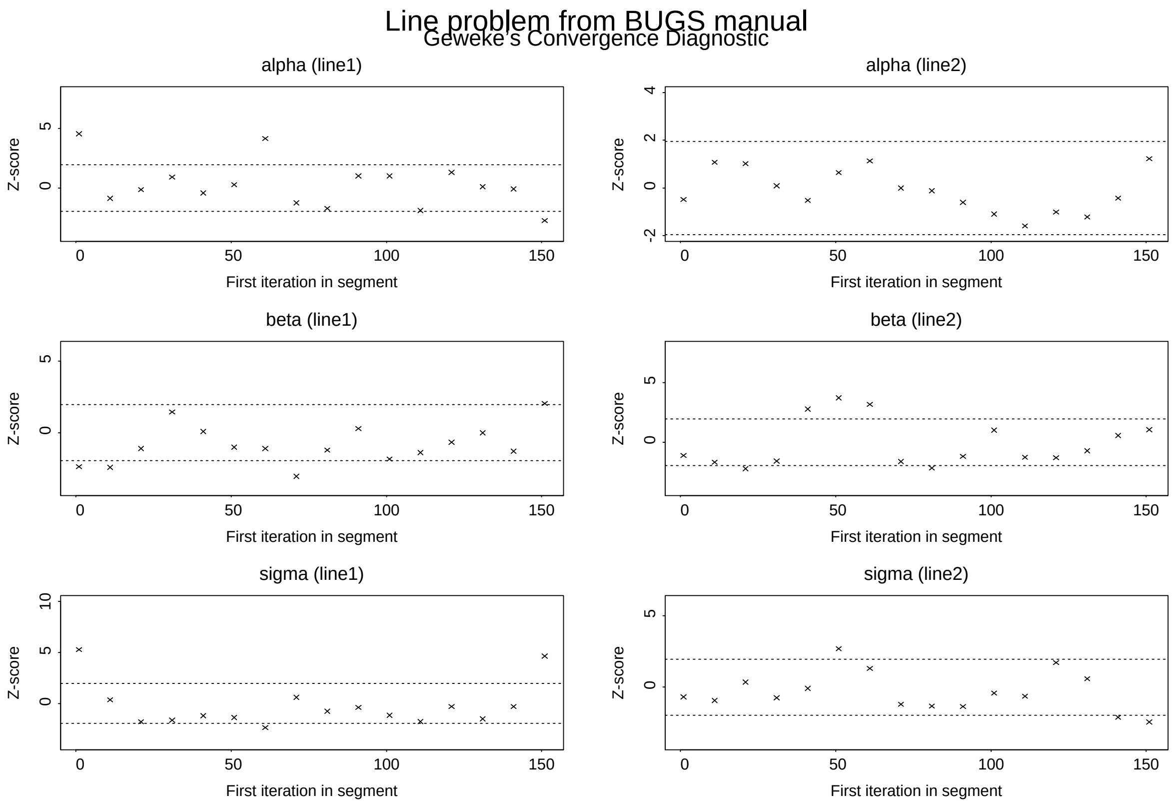

Figure 2: Plot of Geweke's diagnostic produced by selecting `Plot Z-scores` (Option 1) from the `Geweke Plots Menu`

4.2 Gelman & Rubin

The S-Plus code for implementing Gelman and Rubin (1992)'s convergence diagnostic in CODA is based on the `itsim` *function contributed to the Statlib archive by Andrew Gelman (e-mail* `statlib@lib.stat.cmu.edu`*)*

Gelman and Rubin (1992) propose a convergence test based on 2 or more parallel chains, each started from different initial values which are over-dispersed with respect to the true posterior distribution (see Gelman and Rubin (1992) for details concerning construction of over-dispersed starting distributions). Their method is based on a comparison of the within and between chain variances for each variable (essentially a classical analysis of variance). This comparison is used to estimate the factor by which the scale parameter of the marginal posterior distribution of each variable might be reduced if the chain were run to infinity. Best results are obtained for parameters whose marginal posterior densities are approximately normal. Hence `CODA` transforms any variables specified to be positive or restricted to the range (0,1) to the logarithmic or logit scales respectively before calculating this diagnostic.

The Gelman & Rubin diagnostics reported by `CODA` are the 50% and 97.5% quantiles of the sampling distribution for this scale reduction or shrink factor. Note that these quantiles are estimated from the second half of each chain only. If both quantiles are approximately 1.0, effective convergence may be diagnosed *i.e.* samples from the second half each chain may be assumed to have arisen from the stationary distribution. In this case, summary statistics and density estimates etc. may be calculated by combining the latter 50% of iterates from all chains.

Selecting `Gelman & Rubin` (option 2:) from the `CODA Diagnostics Menu` produces the following output for the `line` example:

```
GELMAN AND RUBIN 50% AND 97.5% SHRINK FACTORS:
==============================================

Iterations used = 1:200
Thinning interval = 1
Sample size per chain = 200

-+----------+-----------------------------+-
 | VARIABLE |  Point est. 97.5% quantile  |
 | ======== |  ========== ==============  |
 |          |                             |
 | alpha    |  1.02       1.02            |
 | beta     |  1.00       1.00            |
 | sigma    |  1.04       1.12            |
-+----------+-----------------------------+-
```

This suggests that 100 iterations were sufficient to achieve convergence for `alpha`, `beta` and `sigma`, and that samples 101–200 from both chains may be assumed to arise from the marginal posterior distributions for each variable. Occasionally, a value of `NA` may be shown for the point estimate and 97.5% quantile. This is due to numerical problems during calculation of the shrink factors, and usually arises when there is little or no mixing between chains. This indicates slow convergence requiring much longer runs of the Gibbs sampler.

4.2.1 Plotting Gelman & Rubin's diagnostic

You will then be given the option of producing plots to illustrate these diagnostics:

```
Gelman & Rubin Plots Menu
*************************

1: Trace Plots
2: Shrink Factor Plots
3: Return to Diagnostics Menu
Selection:
```

Selecting option 1 will produce plots of the multiple traces for each variable, with the Gelman & Rubin diagnostics printed above each graph.

Selecting option 2 will generate plots like those shown in Figure 3. These plots are produced by splitting the chain for each variable into a number of segments as follows: the first contains samples $1:50$; the second contains samples $1:(50+n)$; the third contains samples $1:(50+2n)$ and so on. (Note that these segments are constructed differently to those for plotting Geweke's Z-scores — see §4.1.1). The default bin size is n=10 for chains of 500 iterations or less. For longer chains, the bin size is determined by splitting the chain equally into 50 bins. These sizes may be changed by the user (see §5.7.2). Gelman & Rubin's diagnostic is computed for each segment, and the median and 97.5% quantile of the sampling distribution for the resulting shrink factor are plotted against the maximum iteration number for the segment. The plots in Figure 3 show that both the median and 97.5% quantile for `beta` stabilize around a value of 1.0 (indicated by the horizontal dashed line) for chain segments containing the first 150 iterations or more. Since the diagnostic is calculated from the second half of each chain only, this suggests that convergence was first achieved after approximately 75 iterations. The estimated shrink factors for `alpha` and `sigma` also appear to have stabilized around 1.0 for chain segments greater than 150 iterations. However, both show small jumps at about iteration 190 which implies that a longer run may be appropriate. (Note that the vertical scale differs for each of these plots). These plots may take some time to produce due to the large number of computations invovled: the process may be speeded up by reducing the total number of bins, or increasing the bin size.

The early iterations in each chain often show poor mixing. This may occasionally lead to numerical errors when calculating Gelman and Rubin's diagnostic for the initial segments. In such circumstances, `CODA` will display the following warning message:

```
******* Warning: *******
Could not compute Gelman & Rubin's diagnostic for * chain segments
```

where * gives the number of segments generating errors. However, the shrink factors for the later segments remain valid and will be shown on the plot. If *all* segments generate errors, no plots will be produced and `CODA` displays the following error message:

```
******* Error: *******
Cannot compute Gelman & Rubin's diagnostic for any chain segments
This indicates convergence failure ==> Run chains for more iterations
```

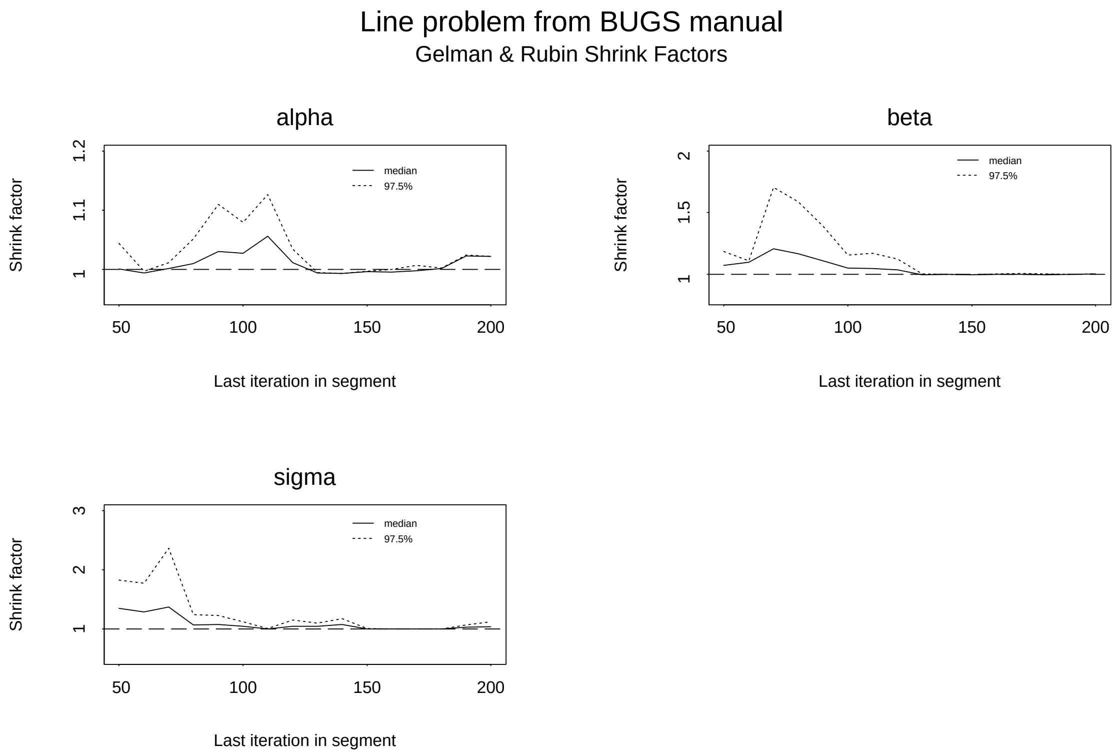

Figure 3: Plot of Gelman & Rubin's diagnostic produced by selecting `Shrink Factor Plots` (Option 2) from the`Gelman & Rubin Plots Menu`

4.3 Raftery & Lewis

The S-Plus code for implementing Raftery and Lewis (1992b)'s convergence diagnostic in CODA is based on the `gibbsit` *function contributed to the Statlib archive by Steven Lewis (e-mail* `statlib@lib.stat.cmu.edu`*)*

Raftery and Lewis (1992*b*)'s method applies to single chains. It is intended both to detect convergence to the stationary distribution and to provide bounds for the accuracy of the estimated quantiles of functions of variables of interest. The user must specify the quantile to be estimated (the CODA default is the $2.5th$ percentile), the desired degree of accuracy for the estimate of this quantile (the CODA default is $\pm$ 0.005) and the required probability of attaining this degree of accuracy (the CODA default is 0.95). The CODA output then reports `Nmin` — the minimum number of iterations that would be needed to estimate the specified quantile to the desired precision if the samples in the chain were independent. This is a theoretical value based on the binomial variance and provides a lower bound for the run-length of the Gibbs sampler. (Note however that in the unusual event that consecutive samples in the output chain are *negatively* correlated, fewer than `Nmin` iterations will be needed — for example, see the `line` output below). `Nmin` will increase as the required probability and degree of accuracy increase. Somewhat counter-intuitively, it also will be larger when estimating quantiles close to the median compared to more extreme quantiles. If the chain length of the BUGS output is less than `Nmin` for the quantile, accuracy and probability values currently specified in CODA, the program will not compute the Raftery & Lewis diagnostics, but will issue an error message stating the value of `Nmin`. If sufficient BUGS iterations are available, CODA will report N — the total number of iterations that should be run for each variable; M — the number of initial iterations to discard as the 'burn-in'; and k — the thinning interval to be used. The final column in the CODA output reports $I = \frac{N}{Nmin}$. This measures the increase in number of iterations needed to reach convergence due to dependence between the samples in the chain. Values of I much greater than 1.0 indicate high within-chain correlations and probable convergence failure (Raftery and Lewis (1992*a*) suggest that $I > 5.0$ often indicates problems). In this case, reparameterization of the model is advised.

Selecting `Raftery & Lewis` (option 3:) from the `CODA Diagnostics Menu` produces the following output for the `line` example:

```
RAFTERY AND LEWIS CONVERGENCE DIAGNOSTIC:
=========================================

Iterations used = 1:200
Thinning interval = 1
Sample size per chain = 200

Quantile = 0.025
Accuracy = +/- 0.005
Probability = 0.95

Chain: line1
============

******* Error: *******
Nmin = 3746; Available chain length is 200
Re-run BUGS program for at least 3746 iterations
OR reduce accuracy, probability and/or quantile to be estimated


Chain: line2
============

******* Error: *******
Nmin = 3746; Available chain length is 200
Re-run BUGS program for at least 3746 iterations
OR reduce accuracy, probability and/or quantile to be estimated
```

This implies that in order to estimate the 2.5% quantile of each parameter to within $\pm$ 0.005 with 95% probability, a minimum of 3746 iterations of the Gibbs sampler are needed. However, if we relax the precision and degree of certainty required for the estimate of this quantile to $\pm$ 0.02 and 90% respectively (see §5.7.3 for details of how to change these parameters in CODA), we obtain the following diagnostics:

```
RAFTERY AND LEWIS CONVERGENCE DIAGNOSTIC:
=========================================

Iterations used = 1:200
Thinning interval = 1
Sample size per chain = 200

Quantile = 0.025
Accuracy = +/- 0.02
Probability = 0.9

Chain: line1
============
-+----------+-----------------------------------------------------+-
 |          | Thin    Burn-in  Total   Lower bound  Dependence   |
 | VARIABLE | (k)       (M)    (N)       (Nmin)     factor (I)   |
 | ======== | ====    =======  =====   ===========  ==========   |
 |          |                                                    |
 | alpha    | 1       5        244     165          1.48         |
 | beta     | 1       2        131     165          0.794        |
 | sigma    | 1       2        160     165          0.97         |
 |          |                                                    |
-+----------+-----------------------------------------------------+-

Chain: line2
============
-+----------+-----------------------------------------------------+-
 |          | Thin    Burn-in  Total   Lower bound  Dependence   |
 | VARIABLE | (k)       (M)    (N)       (Nmin)     factor (I)   |
 | ======== | ====    =======  =====   ===========  ==========   |
 |          |                                                    |
 | alpha    | 1       2        160     165          0.97         |
 | beta     | 1       2        160     165          0.97         |
 | sigma    | 1       2        160     165          0.97         |
 |          |                                                    |
-+----------+-----------------------------------------------------+-
```

The small burn-in values reported above suggest that both chains converge almost immediately for each monitored variable in the `line` example. For `alpha` in chain `line1`, the first 5 iterations should be discarded, and an estimated 44 additional `BUGS` iterations (on top of the 200 already obtained) performed in order to estimate the 2.5th percentile of the posterior distribution to the specified accuracy and probability. For the remaining variables and chains, 200 iterations are more than sufficient to estimate this quantile to the required precision, with only the first 2 iterations needing to be discarded. Note that `N` $<$ `Nmin` for each of the latter variables. This is reflected by `I` < 1, and indicates negative correlations between consecutive iterates in the `BUGS` output.

4.4 Heidelberger and Welch

Heidelberger and Welch (1983) devised a method for detecting an initial transient in simulated sequences of discrete events, but which is also appropriate for use as a convergence diagnostic for the output of Gibbs samplers. Their idea is based on Brownian bridge theory and uses the Cramer-von-Mises statistic to test the null hypothesis that the sampled values for each variable form a stationary process. If the null hypothesis is rejected for a given variable, the test is repeated after discarding the first 10% of iterations. If the hypothesis is again rejected, a further 10% of iterations are discarded. This process is repeated until either a portion of the chain (of length $\geq 50\%$ of the total number of iterations) passes the stationarity test, or 50% of the iterations have been discarded and the null hypothesis is still rejected. If the latter occurs, `CODA` reports the Cramer-von-Mises statistic and indicates that the stationarity test was failed. Missing values (`NA`) are shown in all other columns of the output table for that variable. This indicates that a longer `BUGS` run is needed in order to achieve convergence.

If the stationarity test is passed, `CODA` reports the number of iterations to keep (i.e. which are diagnosed to arise from a stationary process), the number of initial iterations to discard and the Cramer-von-Mises statistic. A halfwidth test is then carried out as follows: for each variable, the portion of the chain which passed the stationarity test is used to estimate the asymptotic standard error of the mean via the time-series method described in §3.2. `CODA` reports the sample mean of the retained iterates and the halfwidth of the associated 95% confidence interval for this mean (*i.e.* $1.96\times$ asymptotic standard error). If the halfwidth is less than ϵ times the sample mean (where ϵ is a small fraction), the halfwidth test is passed and the retained sample is deemed to estimate the posterior mean with acceptable precision. If the halfwidth test is failed, this implies that a longer `BUGS` run is needed to increase the accuracy of the posterior estimates for the given variable. The `CODA` default for ϵ is 0.1; see §5.7.4 for details of how change this value.

Selecting `Heidelberger and Welch` (option 4:) from the `CODA Diagnostics Menu` produces the output shown on the next page for the `line` example.

This suggests that convergence was achieved immediately for the second `BUGS` run (`line2`), and after 20 iterations for the first run (`line1`) for each variable. The halfwidth test indicates that iterations 21–200 from `line1` or all 200 iterations from `line2` (or a combined sample of all these iterations) should yield estimates of the posterior means for `alpha` and `beta` which meet the `CODA` default accuracy criterion of $\epsilon = 0.1$. However, whilst the posterior sample for `sigma` from `line1` passes the halfwidth test, the 200 samples from `line2` do not provide a sufficiently precise estimate of `sigma`. Running this chain for longer should help to increase the precision.

```
HEIDELBERGER AND WELCH STATIONARITY AND INTERVAL HALFWIDTH TESTS:
=================================================================
Iterations used = 1:200
Thinning interval = 1
Sample size per chain = 200

Precision of halfwidth test = 0.1

Chain: line1
============

-+----------+-----------------------------------------------------+-
 |          |  Stationarity   # of iters.  # of iters.  C-vonM    |
 | VARIABLE |      test         to keep    to discard    stat.    |
 | ======== |  ============   ===========  ===========  ======    |
 |          |                                                     |
 | alpha    |  passed         180          20           0.177     |
 | beta     |  passed         180          20           0.243     |
 | sigma    |  passed         180          20           0.205     |
-+----------+-----------------------------------------------------+-
-+----------+------------------------------------+-
 |          |  Halfwidth                         |
 | VARIABLE |    test      Mean    Halfwidth     |
 | ======== |  =========   ====    =========     |
 |          |                                    |
 | alpha    |  passed      2.950   0.0600        |
 | beta     |  passed      0.798   0.0366        |
 | sigma    |  passed      0.868   0.0854        |
-+----------+------------------------------------+-

Chain: line2
============

-+----------+-----------------------------------------------------+-
 |          |  Stationarity   # of iters.  # of iters.  C-vonM    |
 | VARIABLE |      test         to keep    to discard    stat.    |
 | ======== |  ============   ===========  ===========  ======    |
 |          |                                                     |
 | alpha    |  passed         200          0            0.0698    |
 | beta     |  passed         200          0            0.1200    |
 | sigma    |  passed         200          0            0.0550    |
-+----------+-----------------------------------------------------+-
-+----------+------------------------------------+-
 |          |  Halfwidth                         |
 | VARIABLE |    test      Mean    Halfwidth     |
 | ======== |  =========   ====    =========     |
 |          |                                    |
 | alpha    |  passed      2.990   0.0544        |
 | beta     |  passed      0.812   0.0521        |
 | sigma    |  failed      0.982   0.1070        |
-+----------+------------------------------------+-
```

4.5 Autocorrelations

Option 5 of the `CODA Diagnostics Menu` produces a table of autocorrelations within each chain for each monitored variable at lags of 1, 5, 10, and 50. High autocorrelations within chains indicate slow mixing and, usually, slow convergence. This will be characterized by plots of sample traces which "snake" slowly up and down, as opposed to showing more rapid fluctuations over the sample space. Reparameterization may help to reduce autocorrelations. Alternatively, it may be necessary to increase the thinning interval to say, every $5th$ or $10th$ iteration, before calculating summary statistics and density estimates, in order to achieve a less highly correlated sample.

Selecting `Autocorrelations` (option 5:) from the `CODA Diagnostics Menu` produces the following output for the `line` example:

```
LAGS AND AUTOCORRELATIONS WITHIN EACH CHAIN:
============================================

Iterations used = 1:200
Thinning interval = 1
Sample size per chain = 200

-+---------+------------+-----------------------------------------------+-
 | Chain   |  Variable  |   Lag 1      Lag 5      Lag 10     Lag 50      |
 | =====   |  ========  |   =====      =====      ======     ======      |
 |         |            |                                                |
-+---------+------------+-----------------------------------------------+-
 | line1   |  alpha     |   -0.072600   0.060100   0.000849   0.002270   |
 |         |  beta      |   -0.112000   0.066900   0.013600   0.043800   |
 |         |  sigma     |    0.376000   0.029500  -0.004680   0.005140   |
 |         |            |                                                |
-+---------+------------+-----------------------------------------------+-
 | line2   |  alpha     |   -0.124000  -0.009190   0.052500   0.048000   |
 |         |  beta      |    0.053100  -0.012200  -0.033300   0.057600   |
 |         |  sigma     |    0.440000  -0.087400   0.099200  -0.021500   |
 |         |            |                                                |
-+---------+------------+-----------------------------------------------+-
```

4.5.1 Autocorrelation plots

You will then be given the option of producing a plot of the autocorrelation function:

```
Autocorrelation Plots Menu
**************************

1: Plot Autocorrelations
2: Return to Diagnostics Menu
Selection:
```

Selecting option 1 produces the plots shown in Figure 4 for the `line` example.

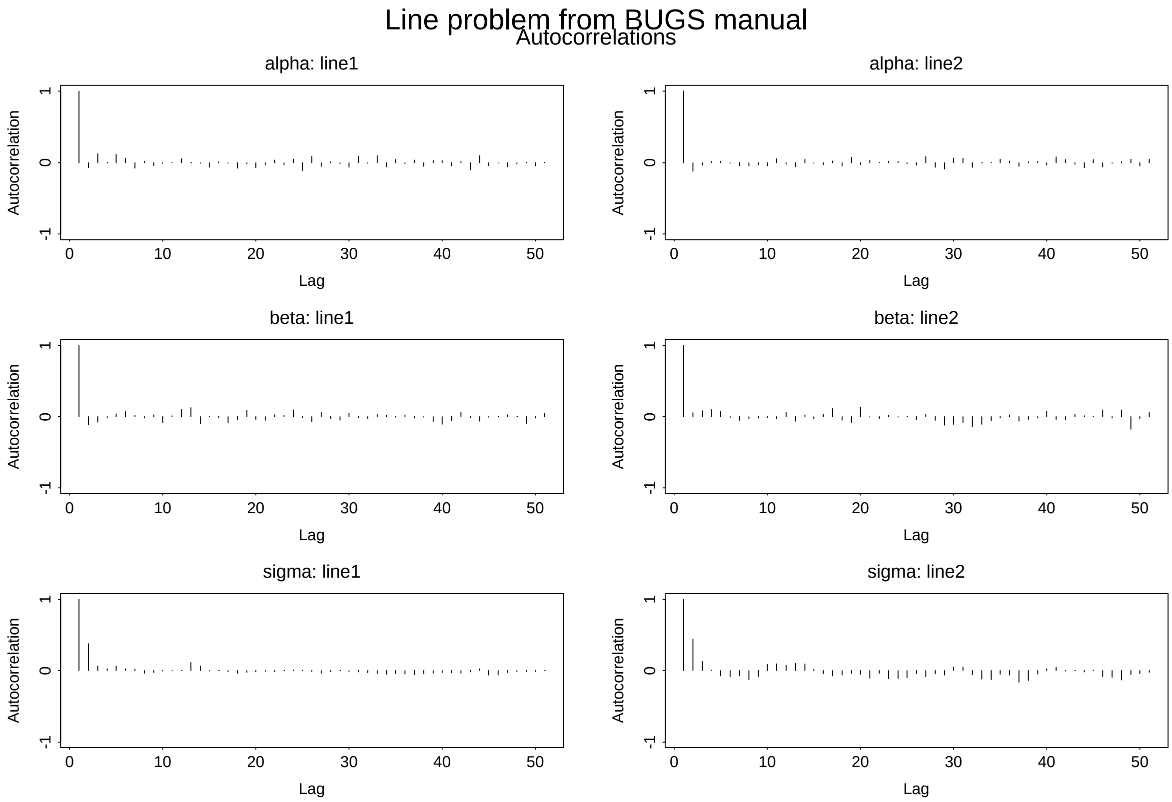

Figure 4: CODA autocorrelation plots for the line example

4.6 Cross-correlations

Option 6 of the CODA Diagnostics Menu produces a table of cross correlations between the monitored variables for each chain. High correlations among parameters are associated with slow convergence and may indicate a need for reparameterization of the model. The following results were obtained for the line example:

```
CROSS-CORRELATION MATRIX:
=========================

Iterations used = 1:200
Thinning interval = 1
Sample size per chain = 200

Chain: line1
============

-+----------+-------------------------------+-
 | VARIABLE |   alpha    beta     sigma     |
 | ======== |                               |
 |          |                               |
 | alpha    |    1.000                      |
 | beta     |   -0.273    1.000             |
 | sigma    |    0.385   -0.383    1.000    |
 |          |                               |
-+----------+-------------------------------+-

Chain: line2
============

-+----------+-------------------------------+-
 | VARIABLE |   alpha    beta     sigma     |
 | ======== |                               |
 |          |                               |
 | alpha    |    1.0000                     |
 | beta     |    0.0080   1.0000            |
 | sigma    |   -0.2650   0.0622   1.0000   |
 |          |                               |
-+----------+-------------------------------+-
```

4.6.1 Cross Correlation plots

You will then be given the option of producing plots to illustrate these cross correlations:

```
Cross Correlation Plots Menu
****************************

1: Plot Cross Correlations
2: Return to Diagnostics Menu
Selection:
```

Selection option 1 produces the plots shown in Figure 5 for the `line` example.

Here, each box corresponds to a cell in the cross correlation matrix. The shading density indicates the magnitude of the correlation, and the angle of the shading shows whether the correlation is negative (lines have negative gradient) or positive (lines have positive gradient).

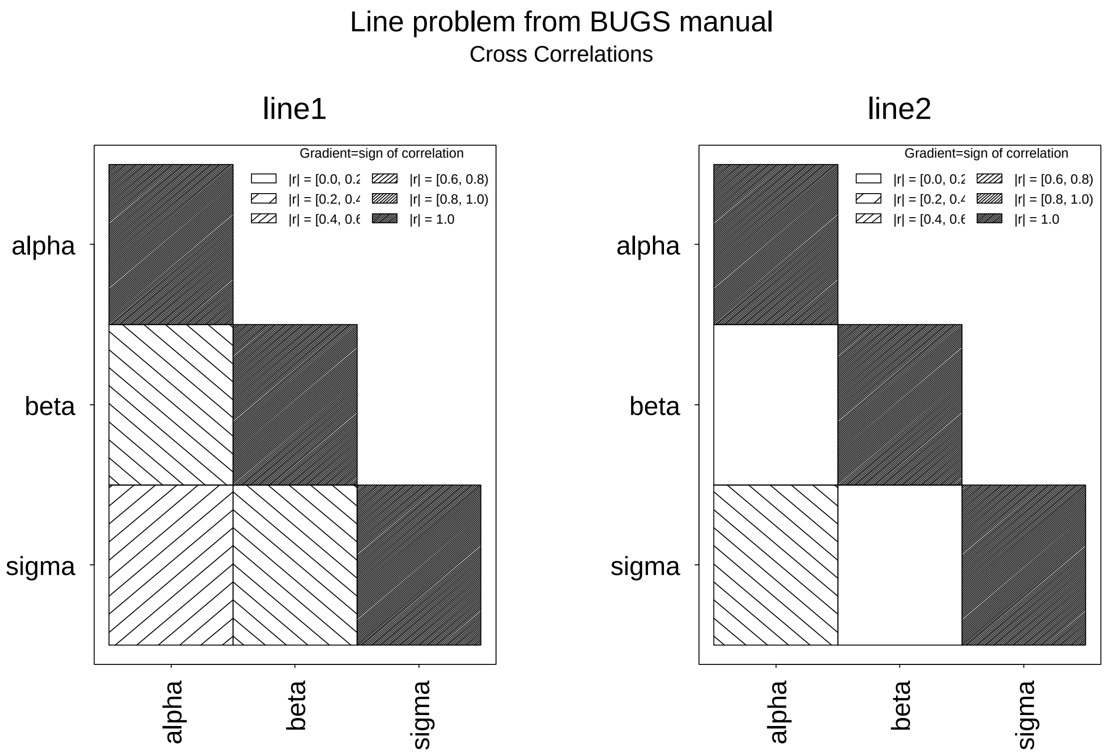

Figure 5: CODA plots of the cross-correlation matrices for the `line` example

5 Changing the CODA defaults

Each CODA menu contains an option labelled `List/Change Defaults`. Selecting this item will display the `CODA Main Defaults Menu`:

```
CODA Main Defaults Menu
***********************

1: List current defaults
2: Select variables for analysis
3: Select chains for analysis
4: Select iterations for analysis
5: Select thinning interval
6: Plots - Defaults Menu
7: Summary Statistics - Defaults Menu
8: Diagnostics - Defaults Menu
9: Return to Output Analysis Menu
10: Return to Diagnostics Menu
11: Return to Main Menu
Selection:
```

This menu enables the user to customize the CODA output for different BUGS problems. Selecting the first option will display a list of all the parameters which may be changed by the user, together with their current values. By default, these are as follows:

```
+----------------------------------------------------------------------+
| DATA DEFAULTS                                                        |
+---------------------+------------------------------------------------+
| Variables selected: | alpha, beta, sigma                             |
|                     |                                                |
| Chains selected:    | line1, line2                                   |
|                     |                                                |
| Iterations - start: | 1                                              |
|                end: | 200                                            |
|                     |                                                |
| Thinning interval:  | 1                                              |
+---------------------+------------------------------------------------+
| SUMMARY STATISTIC DEFAULTS                                           |
+---------------------+------------------------------------------------+
| Significant digits: | 3                                              |
|                     |                                                |
| Combine chains:     | No                                             |
|                     |                                                |
| Batch size:         | 25                                             |
|                     |                                                |
| Quantiles:          | 2.5%, 25%, 50%, 75%, 97.5%                     |
+---------------------+------------------------------------------------+
```

```
+---------------------+-------------------------------------------+
| PLOTTING DEFAULTS                                               |
+---------------------+-------------------------------------------+
| Trace:              | Yes                                       |
|                     |                                           |
| Smooth lines:       | No                                        |
|                     |                                           |
| Density:            | Yes                                       |
|                     |                                           |
| Bandwidth:          | {(max(y) - min(y))/4}                     |
|                     |                                           |
| Separate plot/chain:| No                                        |
|                     |                                           |
| Separate page/var.: | No                                        |
|                     |                                           |
| Postscript options: | Full page; Landscape                      |
|                     |                                           |
| Exponent size:      | 6                                         |
+---------------------+-------------------------------------------+
| DIAGNOSTICS DEFAULTS                                            |
+---------------------+-------------------------------------------+
| Geweke              |                                           |
| ------              |                                           |
| Window 1 fraction:  | 0.1                                       |
| Window 2 fraction:  | 0.5                                       |
| Bin width:          | 10                                        |
| Max number of bins: | 50                                        |
|                     |                                           |
| Gelman & Rubin      |                                           |
| --------------      |                                           |
| Bin width:          | 10                                        |
| Max number of bins: | 50                                        |
|                     |                                           |
| Raftery & Lewis     |                                           |
| ---------------     |                                           |
| Quantile (q):       | 0.025                                     |
| Precision (+/- r):  | 0.005                                     |
| Probability (s):    | 0.95                                      |
|                     |                                           |
| Cross-correlations  |                                           |
| ------------------  |                                           |
| Combine chains:     | No                                        |
+---------------------+-------------------------------------------+
```

5.1 Selecting variables for analysis

Specific variables may be selected for analysis using option 2. A list of all monitored variables will be displayed, and the user prompted to enter the numbers of the variables of interest:

```
Available variables:

+---------------+-------------+
|VARIABLE NUMBER|VARIABLE NAME|
+---------------+-------------+
|1              |alpha        |
+---------------+-------------+
|2              |beta         |
+---------------+-------------+
|3              |sigma        |
+---------------+-------------+

Enter variable number(s) you wish to analyse, separated by commas
(Ranges such as 3:7 may be specified):
1:
```

For example, if you wish to examine only `beta` and `sigma`, you should enter **2:3** (or equivalently **2, 3**) at the prompt. Once you have finished entering the variable numbers you require, hit the `<RETURN>` key twice and the `CODA Main Defaults Menu` will be re-displayed on screen. Note that an error check ensures that only valid variable numbers can be accepted — if illegal numbers or characters are entered, the user is prompted to re-type the required numbers.

5.2 Selecting chains for analysis

Option 3 has a similar format to option 2, and enables selection of specific chains for analysis when multiple `BUGS` runs are available.

5.3 Selecting iterations for analysis

Selecting option 4 from the `CODA Main Defaults Menu` will print which iterations are available, and prompt the user to enter the first and last iterations to be analysed. An error check ensures that the specified iterations are within range, and that the first iteration is less than the last! You will then be prompted as follows:

```
Do you wish to retain ALL iterations for trace plots:
1:
```

Responding **y** to this question causes `CODA` to use the entire chain whenever trace plots are requested, but to use only the selected iterations for kernel density plots and calculation of summary statistics and convergence diagnostics. This option is particularly useful for producing simultaneous plots showing a trace of the whole sample plus a kernel density estimate based on the 'post-convergence' iterations only.

5.4 Selecting the thinning interval

Selecting option 5 from the `CODA Main Defaults Menu` will prompt the user to enter the thinning interval required. For example, the default thinning interval of 1 selects every iteration, whilst an interval of 5 will use every 5*th* iterate.

5.5 Default settings for plots

Selecting option 6 of the `CODA Main Defaults Menu` will display a further menu listing parameters for customizing `CODA` plots:

```
CODA Plotting Defaults Menu
***************************

1: Plot trace of samples
2: Plot kernel density estimate
3: Add smooth line through trace plot
4: Separate plots per chain
5: Separate page of plots per variable
6: Specify page layout for plots
7: Select bandwidth function for kernel smoothing
8: Options for saving plots to postscript file
9: Exponent for scientific notation
10: Return to Previous Menu
Selection:
```

5.5.1 Selecting which plots to produce

Options 1–3 of the `CODA Plotting Defaults Menu` require a yes (**y**) or no (**n**) response, and allow the user to chose whether to produce plots of the sample traces or kernel density estimates or both, and whether or not to add a smooth line to the trace plots. For example, to produce only trace plots without a smooth line, set option 1 to **y** and options 2 and 3 to **n**. See §3.1 for a description of these plots.

5.5.2 How to plot each chain on a separate graph

By default, `CODA` produces a single trace plot for each variable, with different chains shown by different line styles. A single kernel density plot is also produced for each variable, where the samples from multiple chains have been pooled together. If separate trace or density plots are required for each chain, the user should select option 4 of the `CODA Plotting Defaults Menu`:

```
Do you want separate plots per chain for each variable (y/n) ?
1:
```

and respond **y**.

5.5.3 Controlling the page layout of CODA plots

The layout and number of plots appearing on a single graphics page is governed by attempts to minimize the total number of pages of plots produced. Hence CODA may fit up to 9 plots onto one page. However, the user may over-ride this default in various ways. Selecting option 5 of the `CODA Plotting Defaults Menu` enables the user to request a new page of plots for each variable. Hence, if one chain is available, and the user has requested just density plots, a single plot will be produced per page, with one page per variable. If 3 separate chains were available, and the user had also used option 4 of the `CODA Plotting Defaults Menu` to request separate plots for each chain, then 3 plots would appear on each page of graphics output, and so on. Alternatively, the user can control which variables and chains are plotted together by pre-selecting these using options 2 and 3 of the `CODA Main Defaults Menu` before requesting the plots option.

A third alternative is to select option 6 of the `CODA Plotting Defaults Menu`. This enables the user to chose how many rows and columns of plots should appear in the graphics window, rather than leaving CODA to decide this. The following prompt appears first:

```
Do you want to specify your own page layout for the plots (y/n) ?
1:
```

If you respond **y** to this question, you will then be asked to enter the number of rows (maximum allowed = 7) and number of columns (maximum allowed = 8) of plots required. Responding **n** will simply cause the `CODA Plotting Defaults Menu` to be re-displayed on the screen.

5.5.4 Specifying the bandwidth function for the kernel density estimator

The size of bandwidth chosen for kernel density estimation determines the degree of smoothing produced. It is often useful to examine several density plots of the same samples, all smoothed by different amounts, in order to gain greater insight into the data. Selecting option 7 of the `CODA Plotting Defaults Menu` displays the following list of functions which available in CODA for calculating the bandwidth of the kernel density estimator:

```
Select kernel bandwidth function required:

1: Default: Smooth (0.25 * sample range)
2: Coarse (Silverman 1986 eqn. 3.28 & 3.30)
3: User-defined function
4: Return to Plotting Defaults Menu
Selection:
```

Option 1 is the default, and produces a smooth density estimate but one which may hide local features of the posterior.

The bandwidth function recommended by Silverman (1986) (equations 3.28 and 3.30 on pp. 45 and 47) is

$$bw \quad = \quad 1.06 \times \min(\sqrt{\mathrm{var}(y)}, \frac{\mathrm{iqr}(y))}{1.34}) \times n^{-\frac{1}{5}}$$

where bw is the bandwidth to be calculated, y are the sampled values, iqr(y) is the inter-quartile range of y, and n is the length of y. This bandwidth generally produces a less smooth, but possibly more accurate density estimate than does the CODA default, and may be implemented by selecting option 2 of the bandwidth menu.

Selecting the third option from this menu enables the user to specify his or her own bandwidth function. This must be entered as a function of y, and must be written in the S-Plus language. For example, to enter the function

$$bw = 0.9 \times \sqrt{\text{var}(y)} \times n^{-\frac{1}{5}}$$

type:

0.9 * sqrt(var(y)) * length(y)^(−1/5)

after the prompt. Alternatively, a single numeric value (such as **0.5** or **3.2**) may be entered after the prompt if a fixed-value bandwidth is required. An error check is carried out to ensure that the function entered is readable by S-Plus and will return a single numeric value; if not the user is prompted to re-type it.

5.5.5 Producing a postscript file of the graphical output from CODA

CODA permits 3 options for the size and orientation of graphics objects saved as a postscript files (see §3.1.1 for details of how to produce postscript files in CODA). The default is to produce a full page landscape graphic. Alternatively, option 8 of the `CODA Plotting Defaults Menu` enables the user to select either a half page portrait graphic or a full page portrait graphic. Note, however, that half page graphics are only recommended when no more than 4 plots appear on the same page — otherwise the font size becomes too small to read clearly.

5.5.6 Specifying the exponent size required for producing plot labels in scientific notation

Selecting option 9 of the `CODA Plotting Defaults Menu` prompts the user to enter the required exponent size for producing plot labels in scientific notation. The default is 6, which means that numbers ≤ 0.000001 or ≥ 1000000 will appear in scientific notation (e.g. `1.e-6` or `1.e6` respectively) on all CODA plot labels.

5.6 Default settings for summary statistics

Selecting option 7 of the `CODA Main Defaults Menu` will display a further menu listing parameters used during the calculation of summary statistics:

```
CODA Summary Statistics Defaults Menu
*************************************

1: Combine chains for summary statistics
2: Batch size for calculating batch means
3: Quantiles for summary statistics
4: Number of significant digits for printing
5: Return to Previous Menu
Selection:
```

5.6.1 Combining chains to produce a single summary for each variable

By default, `CODA` calculates separate summary statistics for each chain. However, the user may request that the samples for each variable be combined together for these calculations by selecting option 1 of the `CODA Summary Statistics Defaults Menu`.

5.6.2 Setting the batch size

Selecting option 2 of the `CODA Summary Statistics Defaults Menu` prompts the user to enter the number of iterates required in each batch when calculating the batch-mean standard error in the summary statistics table (see §3.2).

5.6.3 Selecting the posterior quantiles to be estimated

Selecting option 3 of the `CODA Summary Statistics Defaults Menu` produces the following prompt:

```
Enter quantiles required, separated by commas
(Default = 0.025, 0.25, 0.5, 0.75, 0.975):
1:
```

The user should then type the value of each quantile required as a fraction between 0 and 1. Once all values have been entered, press the `<RETURN>` key twice to return to the `CODA Summary Statistics Defaults Menu`.

5.6.4 Controlling the number of significant digits displayed for text output

Selecting option 4 of the `CODA Summary Statistics Defaults Menu` prompts the user to enter the required number of significant digits to be displayed on all output. The default is 3, and applies to all summary statistics and convergence diagnostic tables produced by `CODA`.

5.7 Default settings for convergence diagnostic tests

Selecting option 8 of the `CODA Main Defaults Menu` will display a further menu of parameters relating specifically to the various convergence tests:

```
CODA Diagnostics Defaults Menu
******************************

1: Window sizes for Geweke's diagnostic
2: Bin width for plotting Geweke's diagnostic
3: Bin width for plotting Gelman & Rubin's diagnostic
4: Parameters for Raftery & Lewis' diagnostic
5: Halfwidth precision for Heidelberger & Welch's diagnostic
6: Combine chains to calculate correlation matrix
7: Return to Previous Menu
Selection:
```

5.7.1 Changing the window size for Geweke's diagnostic

Selecting option 1 of the `CODA Diagnostics Defaults Menu` enables the user to change the window widths used to calculate Geweke (1992)'s convergence diagnostic (see §4.1). You will be prompted to enter the fraction of the chain to include in each window. An error check ensures that the sum of these fractions doesn't equal or exceed 1.0 (hence the windows should contain distinct portions of the chain with a gap between them).

5.7.2 Changing the bin width for Geweke's and Gelman & Rubin's diagnostics

§4.1.1 and §4.2.1 describe how plots of Geweke (1992)'s and Gelman and Rubin (1992)'s convergence diagnostics are generated by splitting the entire chain for each variable into segments, and calculating the relevant diagnostic for each segment. Selecting options 2 or 3 of the `CODA Diagnostics Defaults Menu` enables the user to change the bin widths (i.e. the number of additional iterations to include in each segment) or total number of bins used to split up the chain. Selecting option 2 of the `CODA Diagnostics Defaults Menu` gives the following menu:

```
Options for defining bin size to plot Geweke's diagnostic:

1: Default: bin width = 10; maximum number of bins = 50
2: User-specified bin width
3: User-specified maximum number of bins
Selection:
```

Option 1 is the default described in §4.1.1. Selecting option 2 will prompt you to enter the number of iterations required per bin. An error check ensures that this number is in the range (1, maximum chain length-50). (Note that specifying a small bin size with a long chain will result in a large total number of bins; since Geweke's diagnostic must be computed separately for each bin, this slows down the plotting process considerably). Alternatively, you may specify the total number of bins required by selecting option 3 from the above menu. An error check ensures that this number is in the range (2, maximum chain length-50).

A similar menu is available for changing the bin size used to plot Gelman & Rubin's diagnostic, and is obtained by selecting option 3 of the `CODA Diagnostics Defaults Menu`

5.7.3 Specifying the parameters for Raftery & Lewis' diagnostic

Raftery and Lewis (1992*b*)'s diagnostic requires the specification of a particular quantile (q) of the posterior distribution to be estimated within $\pm r$ with probability s (see §4.3). Option 4 of the `CODA Diagnostics Defaults Menu` allows the user to specify the values required for q, r, and s. Each value should be a decimal fraction between 0 and 1.

5.7.4 Specifying the precision for Heidelberger and Welch's halfwidth test

Heidelberger and Welch (1983)'s diagnostic requires the specification of a value for ϵ, the precision of the halfwidth test. If the halfwidth of the 95% confidence interval (*i.e.* 1.96 times asymptotic standard error of the mean) exceeds ϵ times the mean, the test is failed (see §4.4). The default value for ϵ is 0.1, but this may be changed using option 5 of the `CODA Diagnostics Defaults Menu`.

5.7.5 Combining chains for cross-correlation analysis

By default, `CODA` computes the cross-correlation matrix separately for each chain. However, in situations where multiple parallel chains have been run and the convergence of each chain diagnosed, the user may wish to pool the iterates from all chains to form a single sample from the posterior. If option 5 from the `CODA Diagnostics Defaults Menu` is selected, the following prompt appears:

```
Do you want to combine all chains to calculate cross-correlation matrix (y/n) ?
1:
```

Responding **y** to this question will cause `CODA` to compute a single cross-correlation matrix for the combined sample when the `Cross-Correlations` option (option 6) of the `CODA Convergence Diagnostics Menu` is subsequently selected.

6 Exiting from CODA and saving the output

To exit from `CODA`, return to the `CODA Main Menu` and select option 4. The following message will appear:

```
Quitting CODA....

Do you want to save the BUGS output as an S-Plus object file(y/n) ?
1:
```

Responding **y** to this question will cause `CODA` to prompt you for a file name. The `BUGS` output will be stored as an S-Plus list object with this name. Each element of the list corresponds to a single `BUGS` chain, and each list element consists of a matrix whose rows correspond to iterations and whose columns correspond to variables. This S-Plus object may be re-used in a subsequent `CODA` session by selecting option 2 `Begin a new CODA session using data saved from a previous CODA session` after the welcome message and prompt which appear when `CODA` is first invoked. **Warning: Saving all your `BUGS` output files as S-Plus objects will quickly consume large**

amounts of storage space. You are advised only to save your BUGS output as an S-Plus object if you intend to carry out repeated CODA analyses on the same data, and CODA takes a long time to read the data from the usual bugs.out file.

If you respond **n** to the above question, CODA will delete all copies of the BUGS output within S-Plus (the original BUGS .out and .ind ASCII files are not deleted however). All other temporary objects created during the CODA session are also deleted, and control is returned to the usual S-Plus prompt (>). To quit S-Plus completely, type **q()** at this prompt.

7 Using CODA with Gibbs sampling output from other programs

CODA is not restricted only for use with BUGS output, but may be applied to the output from any Gibbs sampler. The user must simply ensure that such output is stored in the same format as the BUGS output files. The following 2 files illustrate the requirements:

Output file: myfile.out

```
1  3.095403
2  5.342834
3  2.985434
4  3.839403
5  4.899420
1  10.324302
2  13.234031
3  12.094324
4   9.093849
5  12.849322
1  -0.034222
2  -0.069030
3  -0.049302
4  -0.093243
5  -0.044034
```

Index file: myfile.ind

```
alpha       1    5
beta[1]     6   10
beta[2]    11   15
```

That is, the samples should be saved in a space delimited rectangular ASCII file with filename extension '.out'. Column 1 represents iteration number and column 2 contains the sampled values for each parameter, stored 'end-to-end'. A second ASCII file with extension '.ind' should contain the index associated with the '.out' file. This consists of 3 columns giving the variable names, and line numbers corresponding to the first and last values for each variable in the '.out' file.

Alternatively, CODA can read Gibbs sampler output which is stored directly as an S-Plus object (to input this data format, select option 2: 'Begin a new CODA session using data saved from a previous CODA session' after invoking CODA — see §2.3.2). The S-Plus object must be a list, each element of which represents a separate run of the Gibbs sampler. Optional names may be assigned to each element in the list; otherwise, `CODA` will automatically assign the names ' `chain_1`', ' `chain_2`',... etc. Each element itself consists of a matrix where column 1 (which MUST be named '`iter`') contains the iteration number, and subsequent columns (which MUST be given variable names) contain the sampled values for each variable of interest. The following example illustrates the required S-Plus format for 2 parallel runs of the Gibbs sampler:

```
> my.S.file
[[1]]:
     iter    alpha   beta[1]   beta[2]
[1,]    1 3.095403 10.324302 -0.034222
[2,]    2 5.342834 13.234031 -0.069030
[3,]    3 2.985434 12.094324 -0.049302
[4,]    4 3.839403  9.093849 -0.093243
[5,]    5 4.899420 12.849322 -0.044034

[[2]]:
     iter    alpha   beta[1]   beta[2]
[1,]    1 2.437995  8.303402 -0.065222
[2,]    2 4.710123 11.033923 -0.067843
[3,]    3 5.762293 12.218843 -0.059523
[4,]    4 5.513237  7.221548 -0.079514
[5,]    5 3.660194  8.263983 -0.039586
```

References

Brooks, S. and Roberts, G. (1995). *A review of diagnostic methods for Markov chain Monte Carlo.* Technical report, Dept of Pure Maths and Mathematical Statistics, University of Cambridge. (Available via anonymous ftp from: `ftp.statslab.cam` in directory `/pub/mcmc` or via NetScape or Mosaic at the URL `http://www.statslab.cam.ac.uk/ mcmc/html`).

Cowles, M. K. (1994). *Practical issues in gibbs sampler implementation with application to bayesian hierarchical modeling of clinical trial data.* PhD thesis, Division of Biostatistics, University of Minnesota.

Cowles, M. K. and Carlin, B. P. (1995). Markov chain Monte Carlo convergence diagnostics: a comparative review. *J Amer Statist Assoc*, **(to appear)**.

Gelman, A. and Rubin, D. B. (1992). Inference from iterative simulation using multiple sequences. *Statistical Science*, **7**, 457–72.

Geweke, J. (1992). Evaluating the accuracy of sampling-based approaches to calcualting posterior moments. In *Bayesian Statistics 4*, (ed. J. M. Bernardo, J. O. Berger, A. P. Dawid, and A. F. M. Smith). Clarendon Press, Oxford, UK.

Heidelberger, P. and Welch, P. (1983). Simulation run length control in the presence of an initial transient. *Operations Research*, **31**, 1109–44.

Raftery, A. L. and Lewis, S. (1992*a*). Comment: One long run with diagnostics: Implementation strategies for Markov chain Monte Carlo. *Statistical Science*, **7**, 493–7.

Raftery, A. L. and Lewis, S. (1992*b*). How many iterations in the Gibbs sampler? In *Bayesian Statistics 4*, (ed. J. M. Bernardo, J. O. Berger, A. P. Dawid, and A. F. M. Smith), pp. 763–74. Oxford University Press.

Silverman, B. W. (1986). *Density Estimation for Statistics and Data Analysis.* Chapman and Hall, Bristol.