PostGibbs analysis

Evangelina López de Maturana & Oscar González-Recio

Genome-wide prediction

Topics

MCMC methods

Gibbs sampling

Visual inspection of the chains

Convergence diagnostics

Overview

Simple example

How to generate the plots

How to perform the diagnostic tests

MCMC methods

Exact posterior inference may not be tractable in most cases, because the integration may not always be feasible

MCMC provides a general method by drawing values of the parameters ($\boldsymbol{\theta}$) from an approximate distribution and then correcting those draws to better approximate the target posterior distribution

Gibbs sampling

Gibbs sampling is one of the most used McMC algorithm

Main idea: Given a multivariate distribution it is simpler to sample from a conditional distribution than to integrate over a joint distribution. Iterative process

Toy example:

Step 1. Initialize residual variance and mean

Step 2. Sample residual variance from $p\left(\sigma^2 | \mu , y \right)$

Step 3. Sample mean from $p\left(\mu | \sigma^2 , y \right)$

Gibbs Sampling - Toy example

To estimate the mean and variance of a normal distribution

Simple_example_GS1.R

Diagnostics

After running an MCMC analysis, it is important to assess its performance

Three aspects:

- Mixing
- Burn-in
- Run length

Visually inspecting the chain: Mixing

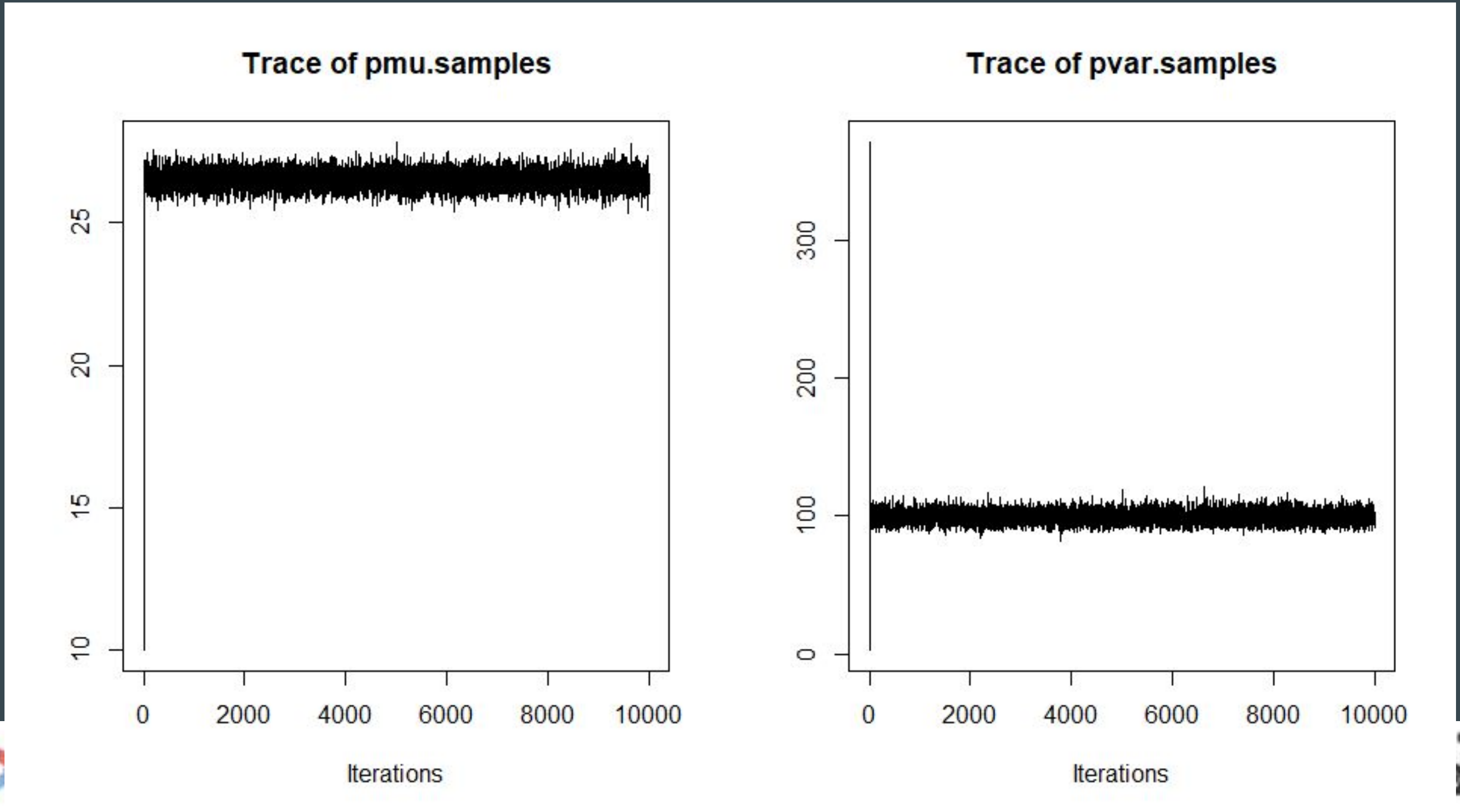
Trace plot: displays the number of iteration vs the value of the parameter

To check if the chain explore the full shape of the target distribution (marginal posterior distribution)

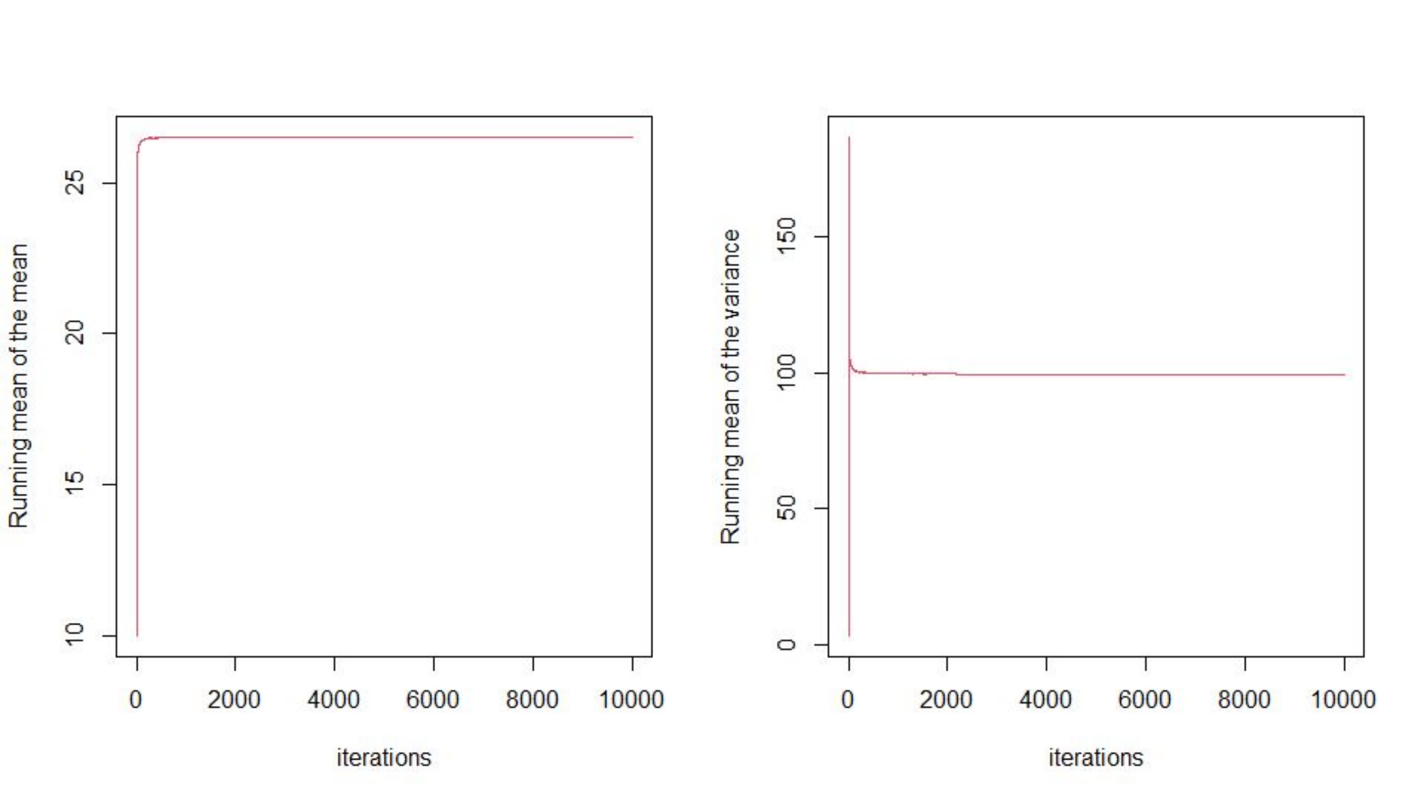
It is an important tool for assessing mixing of a chain

Coda: `traceplot`

Traceplot


Trace of pmu.samples
25
20
15
10
0
2000
4000
6000
8000
10000
Iterations
Trace of pvar.samples
300
200
100
0
0
2000
4000
6000
8000
10000
Iterations

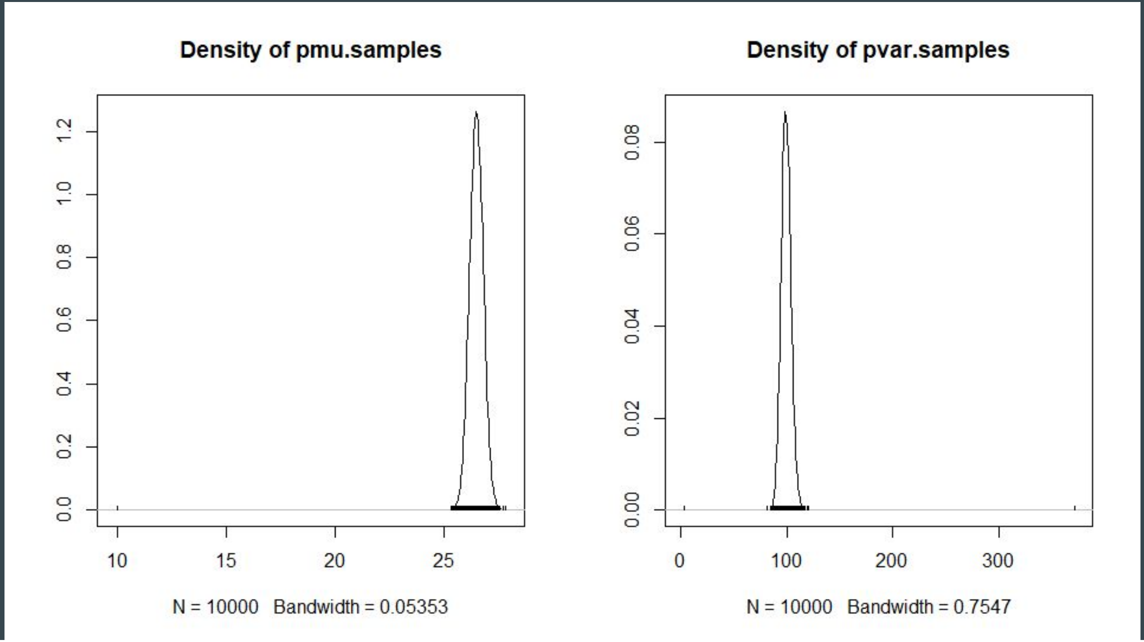
Visually inspecting the chain: running mean



Visually inspecting the chain: Mixing

Density plot of the samples of the conditional posterior distribution of the parameters

Coda: `densplot`



Genome-wide prediction

Autocorrelations

In MCMC, each sample depends on the one before - autocorrelation

We can check the autocorrelations between the samples

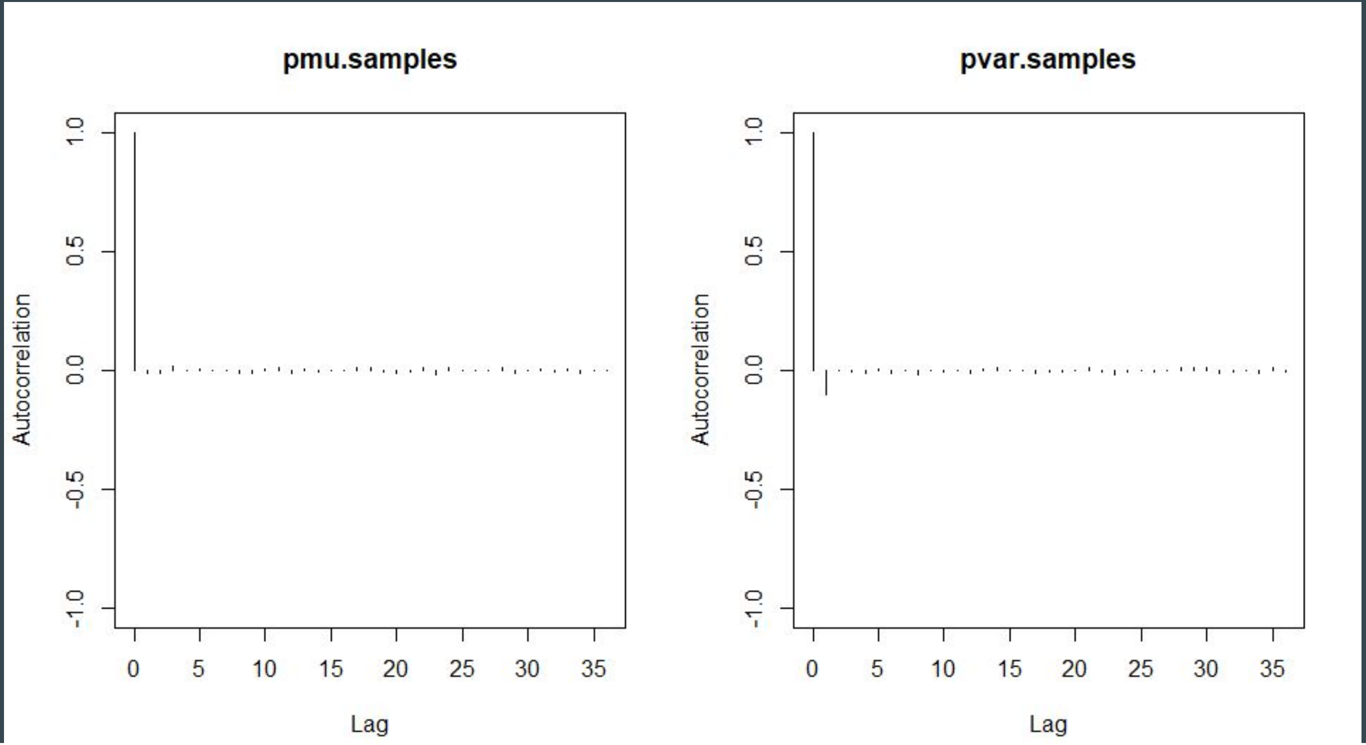
Lag-k autocorrelation: correlation between every sample and the sample k samples after

It decreases as k increases; if it remains high → high level of autocorrelation → poor mixing

Solution: thinning and increase the length of the chain → more efficient storage

Coda: `autocorr`, `autocorr.diag`, `autocorr.plot`

Autocorrelation plot



Burn-in

It is possible to assess it by:

- Visually inspecting the trace plots
- Visually inspecting the running means
- Use of convergence tests such as:
    - Geweke diagnostic
    - Heidelberger-Welch
    - Raftery-Lewis
    - Gelman-Rubin

Checking the convergence

The convergence of an MCMC algorithm in important for the correct estimation of the posterior distribution of the parameters of interest

Problem: the convergence may not be diagnosed as clearly as in optimization methods

It is important to specify:

- The length of the burnin period
- The number of samples that will be used for the posterior analysis
- Specification of the thinning interval

Conditional posterior distribution → Marginal posterior distribution of the parameters

Geweke’s convergence test (Geweke, 1992)

Test for equality of the means of the first and last part of a Markov chain (by default the first 10% and the last 50%)

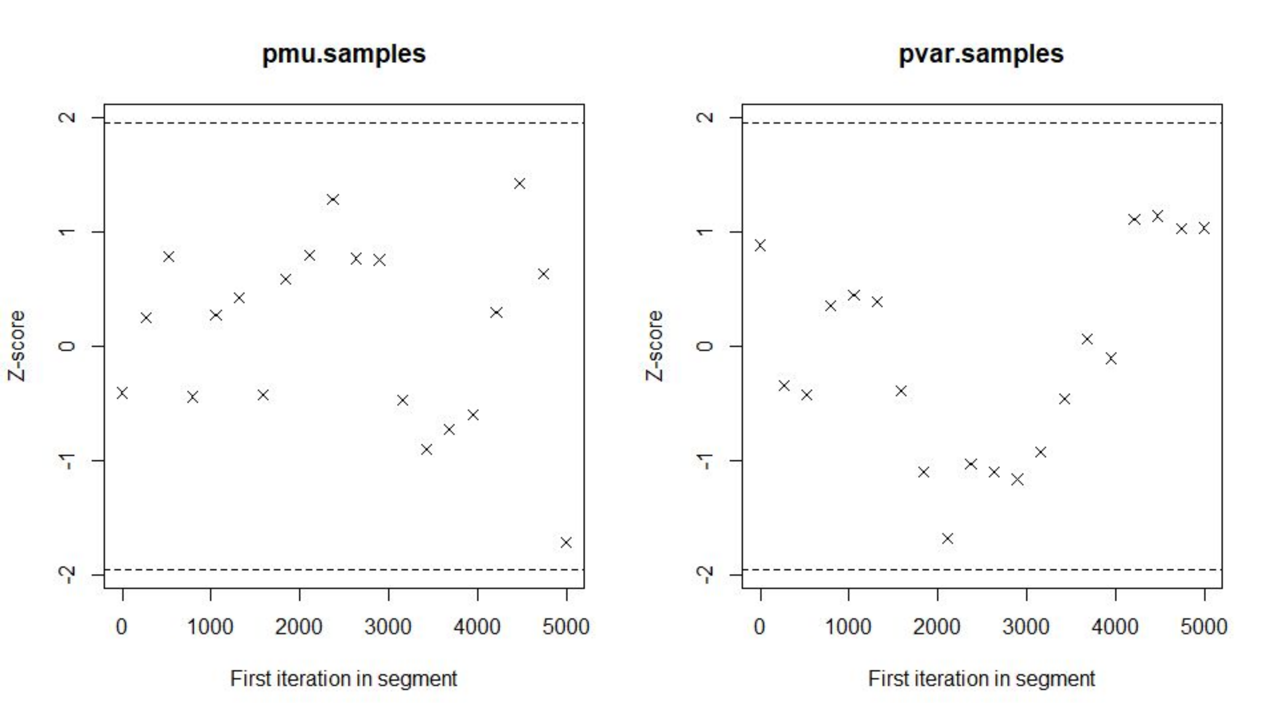
If the samples are drawn from the stationary distribution of the chain, the two means are equal and Geweke’s statistic has an asymptotically standard normal distribution

The test statistic is a standard Z-score: the difference between the two sample means divided by its estimated standard error

The Z-score is calculated under the assumption that the two parts of the chain are asymptotically independent, which requires that the sum of frac1 and frac2 be strictly less than 1.

Geweke’s convergence test (Geweke, 1992)

CODA: `geweke.diag`; `geweke.plot`



Raftery and Lewis

Diagnostic based on a criterion of accuracy of estimation of the quantile q

It is intended for use on a short pilot run of a Markov chain

The number of iterations required to estimate the quantile q (2.5%) to within an accuracy of +/- r (0.005) with probability p (0.95) is calculated (Nmin and N)

Positive autocorrelation will increase the required sample size above this minimum value.

An estimate I (the ‘dependence factor’, I=N/Nmin, estimate of the thinning interval) of the extent to which autocorrelation inflates the required sample size is also provided:

Values of I larger than 5 indicate strong autocorrelation

Raftery and Lewis

```
Quantile (q) = 0.025
Accuracy (r) = +/- 0.005
Probability (s) = 0.95

                Burn-in   Total Lower bound   Dependence
                (M)       (N)   (Nmin)        factor (I)
 pmu.samples  2        3666  3746           0.979
 pvar.samples 2        3666  3746           0.979
```

Heidelberger -Welch diagnostic

Based on the Brownian bridge theory

It tests whether the stationarity of the Markov chain is attained using the values from the MCMC output

It is first applied to the whole chain, then after discarding the first 10%, 20%, 30%..., till either the null hypothesis is accepted or 50% of the chain has been discarded → failure of the stationary test and a longer chain is needed

CODA package: `heidel.diag`

Heidelberger -Welch diagnostic

```
            Stationarity start     p-value
             test         iteration
pmu.samples  passed       1         0.885
pvar.samples passed       1         0.568
             Halfwidth Mean Halfwidth
             test
pmu.samples  passed    26.5 0.00695
pvar.samples passed    99.5 0.09430
```

Gelman and Rubin diagnostic

It implies checking the convergence of two chains

It is an ANOVA-type diagnostic, calculating a shrinking factor (*R*):

- When *R* close to 1 indicate convergence

CODA: `gelman.diag; gelman.plot`

High Posterior Density interval

Credible intervals: interval within which a parameter value falls with a particular probability

- There is a 95% of probability that the parameter lies in the interval
- Narrowest credible interval is the most interesting one → highest density interval

Coda: `HPDinterval`

High Posterior Density interval

```
                lower     upper
pmu.samples  25.89846  27.11789
pvar.samples 90.98152 108.62931
attr(,"Probability")
[1] 0.95
```