Estimating AR(2) parameters in WinBUGS and JAGS

RK

June 10, 2015

Abstract

This document gives the details behind estimating an AR(2) process in Bayesian framework. The estimation is done in WinBUGS and JAGS.

Contents

1. Context

One gets to appreciate the power of WinBUGS or JAGS or Stan only when one tries to do Bayesian estimation the hard way, i.e coding one's own sampler. In this document, I will derive the basic equations relating to AR(2) process and illustrate the use of WinBUGS and JAGS to do parameter estimation.

2. AR(2) math

Consider the following AR(2) process with parameters, $(\phi_1, \phi_2, \sigma^2)$:

$$X_t = \phi_1 X_{t-1} + \phi_2 X_{t-2} + \epsilon_t,\ \epsilon_t \sim N(0, \sigma^2) \tag{1}$$

The likelihood function for the data is

$$f(X_1, X_2, \ldots, X_t) = f(X_1) \prod_{i=2}^{t} f(X_i | X_1, \ldots X_{i-1})$$

Since the joint distribution is a multivariate normal, so are conditional distributions. Denote the conditional means and conditional variances as follows

$$\begin{aligned} f(X_1) &\sim N(\mu_{X_1}, \sigma^2_{X_1}) \\ f(X_2|\cdot) &\sim N(\mu_{X_2|\cdot}, \sigma^2_{X_2|\cdot}) \\ &\vdots \\ f(X_t|\cdot) &\sim N(\mu_{X_t|\cdot}, \sigma^2_{X_t|\cdot}) \end{aligned}$$

where $X_i|\cdot$ denotes the X_i conditional on $X_1, X_2, \ldots X_{i-1}$

To obtain these conditional distributions, it is important to know the autocovariance structure of the process. Let $\gamma(0), \gamma(1), \gamma(2)$ represent the autocovariances at lag $0, 1, 2$ respectively. One of the standard procedures is via *Yule walker equations*.

- Multiplying equation(1) by X_t and taking expectations gives

$$\gamma(0) = \phi_1 \gamma(1) + \phi_2 \gamma(2) + \sigma^2$$

- Multiplying equation(1)by X_{t-1} and taking expectations gives

$$\gamma(1) = \phi_1 \gamma(0) + \phi_2 \gamma(1)$$

- Multiplying equation(1) by X_{t-2} and taking expectations gives

$$\gamma(2) = \phi_1 \gamma(1) + \phi_2 \gamma(0)$$

Solving the above three equations, one obtains

$$\gamma(0) = \frac{\sigma^2(1-\phi_2)}{(1+\phi_2)[(1-\phi_2)^2-\phi_1^2]}$$
$$\gamma(1) = \frac{\phi_1}{1-\phi_2}\gamma(0)$$

The above covariance structure can be used to obtain conditional distributions via *Levinson-Durbin* algorith. The basic idea of the algorith is explore the connection between conditionality and orthogonal projection. It is elegant in the sense that it is recursive. There is no need to invert matrices to get to forecast estimates. To be more precise, the aim is to forecast X_{t+1} based on the $X_t, X_{t-1}, \ldots, X_1$.

$$X_{t+1|\cdot} = \sum_{i=1}^{t} \alpha_{t,j} X_{t+1-j}$$

The algo is about recursively estimating $\{\alpha_{t,i}; i = 1, \ldots, t\}$ given $\{\alpha_{t-1,i}; i = 1, \ldots, t-1\}$(which are the coefficients of $(X_t|\cdot)$ Let the autocovariance function of the process be $\gamma(k)$. The recursive equations are

$$\alpha_{1,1} = \gamma(1)/\gamma(0), r(0) = \gamma(0)$$
$$\alpha_{t,t} = \frac{\gamma(t) - \sum_{i=1}^{t-1} \alpha_{t-1,i}\gamma(t-i)}{r(t-1)}$$
$$\alpha_{t,i} = \alpha_{t-1,i} - \alpha_{t,t}\,\alpha_{t-1,i},\ 1 \le i \le t-1$$
$$r(t) = r(t-1)[1-\alpha_{tt}^2]$$

Given the above equations one can derive the conditional distributions

For $X_1 \sim N(\mu_{X_1}, \sigma^2_{X_1})$,

$$\mu_{X_1} = 0$$
$$\sigma^2_{X_1} = \gamma(0)$$

For $X_2|\cdot \sim N(\mu_{X_2}, \sigma^2_{X_2})$,

$$\mu_{X_2|\cdot} = \alpha_{1,1} - \frac{\gamma(1)}{\gamma(0)}X_1 = \frac{\phi_1}{1-\phi_2}X_1$$
$$\sigma^2_{X_2|\cdot} = r(0) - \alpha_{1,1}\gamma(1) = \frac{1}{(1-\phi_2^2)}$$

For $X_i|\cdot \sim N(\mu_{X_i}, \sigma^2_{X_i})$, $3 \le i \le t$

$$\mu_{X_i|\cdot} = \phi_1 X_{i-1} + \phi_2 X_{i-2}$$
$$\sigma^2_{X_i|\cdot} = 1$$

Hence the likelihood function is

$$f(X_1, X_2, \ldots, X_t) = f(X_1)\prod_{i=2}^{t} f(X_i|X_1, \ldots X_{i-1}) = \prod_{i=1}^{t} N(\mu_{X_i}, \sigma^2_{X_i})$$

3. What's the advantage of using WinBUGS or JAGS ?

The standard way to perform Bayesian analysis is to use the likelihood function and prior to obtain posterior distribution.

$$\pi(\sigma^2, \phi_1, \phi_2 | X_1, X_2, \ldots X_t) = f(X_1, X_2, \ldots, X_t | \sigma^2, \phi_1, \phi_2)\pi(\sigma^2)\pi(\phi_1, \phi_2)$$

If one tries to derive full conditional distribution for σ^2 and (ϕ_1, ϕ_2) with a proper choice of conjugate priors, only the former gives rise to a standard density. To be precise, if one assumes an inverse gamma for σ^2 prior, one gets a posterior for σ^2 as inverse gamma. However the full conditional for (ϕ_1, ϕ_2) turns out to be a nonstandard density function. This means one needs to write a Metropolis Hastings sampler. For AR(2) it is not difficult to write an M-H sampler but it becomes tedious. Softwares such as WinBUGS make a modeler's life easy. One has to represent the model by Directed Acyclic Graph (DAG) and the rest of the estimation is taken care by the software. If there is a need for Metropolis Hasting step in the DAG, WinBUGS automatically generates a candidate generating density and tunes the distribution adaptively so that the chain samples from all parts of the distribution.

4. Estimation using WinBUGS

The model file can be written in `R` as follows :

```
{
sink("model.txt")
cat("
model
{
  psi[2]  ~ dunif(-1, 1)
  u       <- abs(1 - psi[2])
  l       <- -u
  psi[1]  ~ dunif(l, u)
  mu[1]   <- 0
  mu[2]   <- (psi[1]/(1-psi[2]))*x[1]
  tau1    <- ((1+psi[2])/(1-psi[2]))*((1-psi[2])*(1-psi[2]) - psi[1]*psi[1])*tau
  tau2    <- tau*(1 - psi[2]*psi[2])
  tau     ~ dgamma(0.001, 0.001)
  x[1]    ~ dnorm(mu[1], tau1)
  x[2]    ~ dnorm(mu[2], tau2)
   for(i in 3:n){
    x[i]  ~ dnorm(mu[i],tau)
    mu[i] <- psi[1]*x[i-1] + psi[2]*x[i-2]
  }
  sigma   <- 1/sqrt(tau)
}
",fill = TRUE)
sink()}
```

Let's simulate some data with true parameters as $(\phi_1, \phi_2) = (1, -0.5)$ and $\sigma = 1$

```
set.seed(1)
n               <- 100
ar2_data        <- arima.sim( n = n, list( order=c(2,0,0), ar = c(1,-0.5)), sd = 1)
```

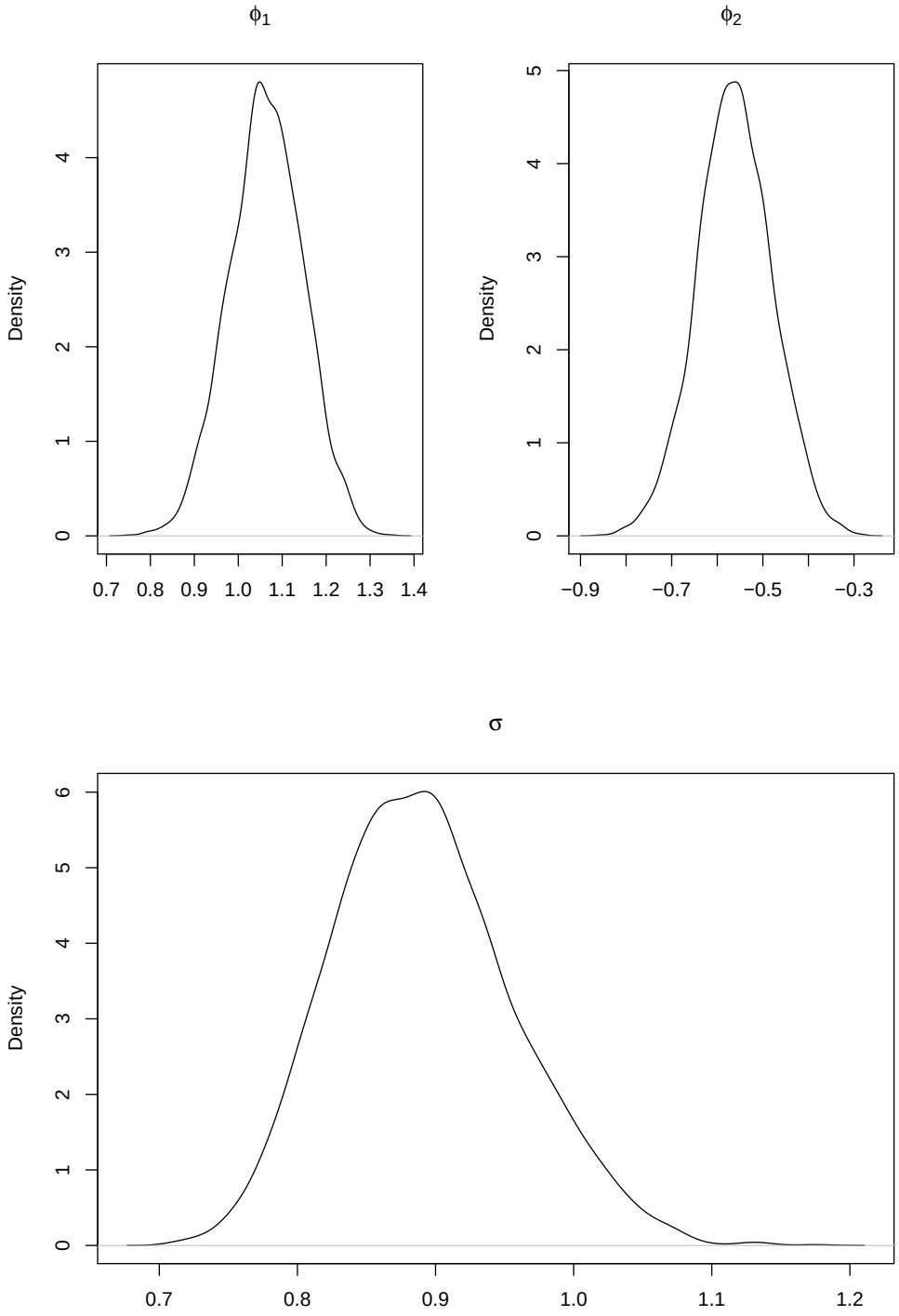
One need not open WinBUGS to do the estimation as packages like `R2WinBUGS` help the modeler work within `R`.

```
#dataset
data            <- list(x = c(ar2_data) , n = n)

# Initialize parameters
inits           <- function(){ list(tau = 1,psi = c(0.4, -0.4)) }

# Parameters to track
params          <- c("psi", "sigma")

# MCMC settings
nc              <- 3
ni              <- 1200
nb              <- 200
nt              <- 1

# bugs directory
bugs.directory <- "C:/Cauldron/garage/WinBUGS14"

# Use WinBUGS
samples         <- bugs(data = data, inits = inits, parameters = params,
                 model="model.txt", n.thin=nt, n.chains=nc, n.burnin=nb,
                 n.iter = ni, debug = FALSE,codaPkg = FALSE,
                 bugs.directory = bugs.directory)
```

```
print(samples$summary[1:3,])
```

```
##              mean         sd       2.5%      25%     50%       75%
## psi[1]  1.0633168 0.08359909  0.8998000  1.00700  1.0630  1.121000
## psi[2] -0.5614634 0.08207536 -0.7224050 -0.61665 -0.5617 -0.506325
## sigma   0.8914375 0.06535886  0.7753925  0.84490  0.8877  0.933525
##             97.5%     Rhat n.eff
## psi[1]  1.2290000 1.001530  2100
## psi[2] -0.3996675 1.001371  2300
## sigma   1.0270250 1.000801  3000
```

Density Estimates of the parameters



5. Estimation using JAGS

```
model <- jags.model("model.txt", data = data, n.chains = 3, n.adapt= 1200, quiet = TRUE)
update(model, 200);
samples_jags <- coda.samples(model, variable.names=c("psi", "sigma"), n.iter=1200)
```

```
print((summary(samples_jags))$statistics)

##              Mean         SD    Naive SE Time-series SE
## psi[1]  1.0631413 0.08394021 0.001399004    0.003087546
## psi[2] -0.5599392 0.08257345 0.001376224    0.002972046
## sigma   0.8907646 0.06410196 0.001068366    0.001068552
```