Homework - Calculate DIC for single AR model – M2L2

Capstone Project: Bayesian Conjugate Analysis for Autogressive Time Series Models

Oren Bochman

2025-07-02

AR(3)

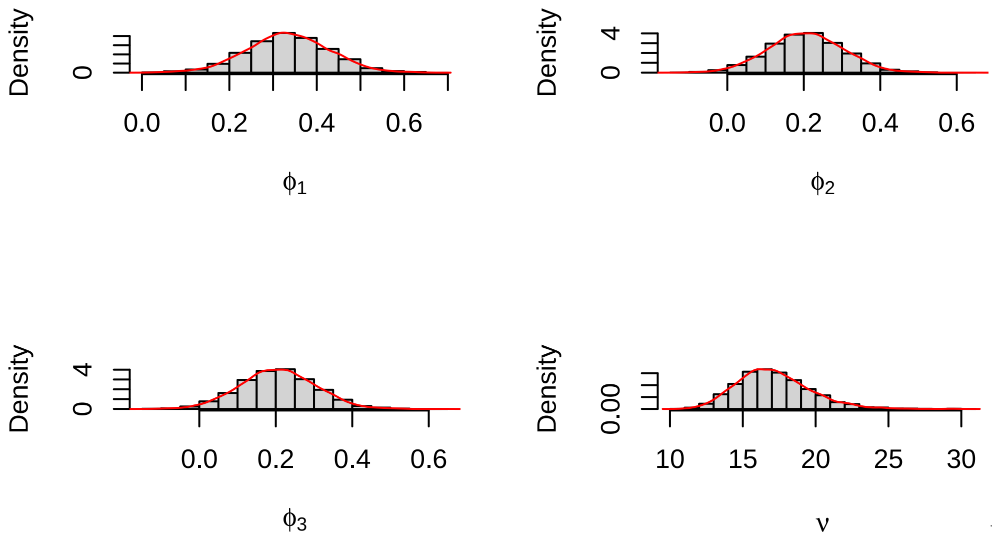
1. For the earthquake data from the previous analysis, you should conclude that the data should be fitted with an AR(3) model. Now fit the data using an AR(3) model, using prior $m_0 = (0,0,0)^T, C_0 = 10I_3, n_0 = d_0 = 0.02$. Obtain 5000 posterior samples for all the model parameters and then calculate the DIC using these 5000 posterior sample. What is the DIC value you get?

```
## read data, you need to make sure the data file is in your current working directory
earthquakes.dat <- read.delim("data/earthquakes.txt")
earthquakes.dat$Quakes=as.numeric(earthquakes.dat$Quakes)

y.dat=earthquakes.dat$Quakes[1:100] ## this is the training data
y.new=earthquakes.dat$Quakes[101:103] ## this is the test data
y.sample=y.dat
```

```
## fit AR(p) with p=3 and sample from posterior

## 1. setup
library(mvtnorm)

n.all=length(y.sample)      # Total number of observations
p=3                         # AR order
m0=matrix(rep(0,p),ncol=1)  # Prior mean of AR coefficients
C0=diag(p)*10               # Prior covariance (diffuse)
n0=0.02; d0=0.02            # Prior parameters for innovation variance (Inverse-Gamma)
```

```r
## 2. prepare data matrices
Y = matrix(y.sample[(p+1):n.all], ncol=1)
#Fmtx = matrix(c(y.sample[2:(n.all-1)],y.sample[1:(n.all-p)]),nrow=p,byrow=TRUE)
Fmtx = t(embed(y.sample, p+1)[, 2:(p+1)])

n=length(Y)

## 3. Posterior update
e = Y - t(Fmtx) %*% m0           # residuals under prior mean
Q = t(Fmtx) %*% C0 %*% Fmtx + diag(n)  # precision-like matrix
Q.inv = chol2inv(chol(Q))        # inverse via Cholesky

A = C0 %*% Fmtx %*% Q.inv       # Kalman gain-like update
m = m0 + A %*% e                # posterior mean of phi
C = C0 - A %*% Q %*% t(A)        # posterior covariance of phi

n.star = n + n0                  # post parameters for innovation variance
d.star = t(Y - t(Fmtx) %*% m0) %*% Q.inv %*% (Y - t(Fmtx) %*% m0) + d0 # post parameters for


n.sample=5000

nu.sample=rep(0,n.sample)
phi.sample=matrix(0,nrow=n.sample,ncol=p)

for (i in 1:n.sample){
  set.seed(i)
  nu.new=1/rgamma(1,shape=n.star/2,rate=d.star/2)
  nu.sample[i]=nu.new
  phi.new=rmvnorm(1,mean=m,sigma=nu.new*C)
  phi.sample[i,]=phi.new
}
```

```r
par(mfrow=c(2,2))
hist(phi.sample[,1],freq=FALSE,xlab=expression(phi[1]),main="")
lines(density(phi.sample[,1]),type='l',col='red')
hist(phi.sample[,2],freq=FALSE,xlab=expression(phi[2]),main="")
lines(density(phi.sample[,2]),type='l',col='red')
hist(phi.sample[,2],freq=FALSE,xlab=expression(phi[3]),main="")
lines(density(phi.sample[,2]),type='l',col='red')
hist(nu.sample,freq=FALSE,xlab=expression(nu),main="")
lines(density(nu.sample),type='l',col='red')
```



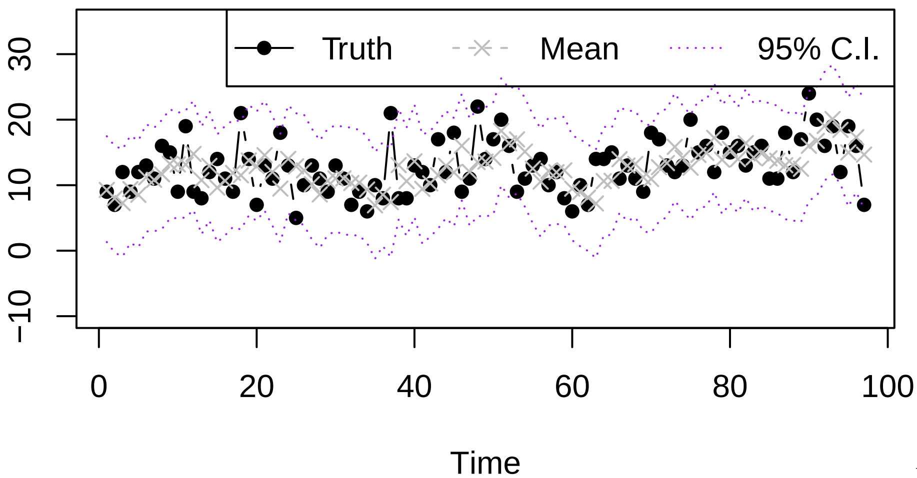
{#fig-capstone-histogram-earthquake-parameters-ar(3) fig-pos='H'}

```
cal_log_likelihood=function(phi,nu){
  mu.y = t(Fmtx)%*%phi
  log.lik=sapply(1:length(mu.y),function(k){dnorm(Y[k,1],mu.y[k],sqrt(nu),log=TRUE)})
  sum(log.lik)
}
phi.bayes=colMeans(phi.sample)
nu.bayes=mean(nu.sample)
log.lik.bayes=cal_log_likelihood(phi.bayes,nu.bayes)
cat(log.lik.bayes)
```

```
-274.4609
```

```
## get in sample prediction
post.pred.y=function(s){

  beta.cur=matrix(phi.sample[s,],ncol=1)
  nu.cur=nu.sample[s]
  mu.y=t(Fmtx)%*%beta.cur
  sapply(1:length(mu.y), function(k){rnorm(1,mu.y[k],sqrt(nu.cur))})
}

y.post.pred.sample=sapply(1:5000, post.pred.y)
```

```r
## show the result
summary.vec95=function(vec){
  c(unname(quantile(vec,0.025)),mean(vec),unname(quantile(vec,0.975)))
}

summary.y=apply(y.post.pred.sample,MARGIN=1,summary.vec95)

plot(Y,type='b',xlab='Time',ylab='',ylim=c(-10,35),pch=16)
lines(summary.y[2,],type='b',col='grey',lty=2,pch=4)
lines(summary.y[1,],type='l',col='purple',lty=3)
lines(summary.y[3,],type='l',col='purple',lty=3)
legend("topright",legend=c('Truth','Mean','95% C.I.'),lty=1:3,col=c('black','grey','purple')
       horiz = T,pch=c(16,4,NA))
```



{#fig-capstone-posterior-inference-earthquake-ar(3)-check fig-pos='H'}

```r
post.log.lik=sapply(1:5000, function(k){cal_log_likelihood(phi.sample[k,],nu.sample[k])})
E.post.log.lik=mean(post.log.lik)

p_DIC=2*(log.lik.bayes-E.post.log.lik)

DIC=-2*log.lik.bayes+2*p_DIC
cat(DIC)
```

```
556.901
```

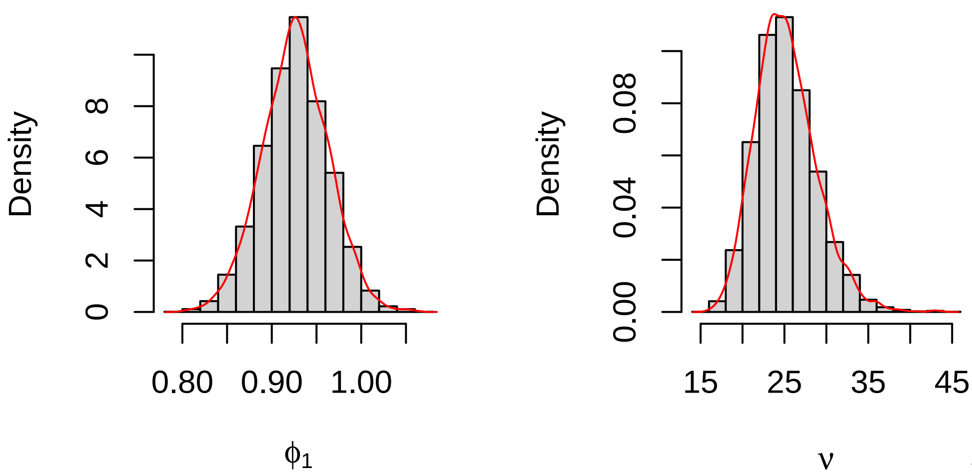
AR(1)

2. For the earthquake data from the previous analysis, you should conclude that the data should be fitted with an AR(3) model. Now fit the data using an AR(3) model, using prior $m_0 = (0), C_0 = 10, n_0 = d_0 = 0.02$. Obtain 5000 posterior samples for all the model parameters and then calculate the DIC using these 5000 posterior sample. What is the DIC value you get?

```
## fit AR(p) with p=3 and sample from posterior

## 1. setup
library(mvtnorm)

n.all=length(y.sample)      # Total number of observations
p=1                         # AR order
m0=matrix(rep(0,p),ncol=1)  # Prior mean of AR coefficients
C0=diag(p)*10               # Prior covariance (diffuse)
n0=0.02; d0=0.02            # Prior parameters for innovation variance (Inverse-Gamma)

## 2. prepare data matrices
Y = matrix(y.sample[(p+1):n.all], ncol=1)
#Fmtx = matrix(c(y.sample[2:(n.all-1)],y.sample[1:(n.all-p)]),nrow=p,byrow=TRUE)
Fmtx = t(embed(y.sample, p+1)[, 2:(p+1)])

n=length(Y)

## 3. Posterior update
e = Y - t(Fmtx) %*% m0           # residuals under prior mean
Q = t(Fmtx) %*% C0 %*% Fmtx + diag(n)  # precision-like matrix
Q.inv = chol2inv(chol(Q))        # inverse via Cholesky

A = C0 %*% Fmtx %*% Q.inv       # Kalman gain-like update
m = m0 + A %*% e                # posterior mean of phi
C = C0 - A %*% Q %*% t(A)        # posterior covariance of phi

n.star = n + n0                  # post parameters for innovation variance
d.star = t(Y - t(Fmtx) %*% m) %*% Q.inv %*% (Y - t(Fmtx) %*% m0) + d0 # post parameters for
#d.star = t(Y - t(Fmtx) %*% m) %*% (Y - t(Fmtx) %*% m) + d0


n.sample=5000
```

```r
nu.sample=rep(0,n.sample)
phi.sample=matrix(0,nrow=n.sample,ncol=p)

for (i in 1:n.sample){
  set.seed(i)
  nu.new=1/rgamma(1,shape=n.star/2,rate=d.star/2)
  nu.sample[i]=nu.new
  phi.new=rmvnorm(1,mean=m,sigma=nu.new*C)
  phi.sample[i,]=phi.new
}
```

```r
par(mfrow=c(1,2))
hist(phi.sample[,1],freq=FALSE,xlab=expression(phi[1]),main="")
lines(density(phi.sample[,1]),type='l',col='red')
hist(nu.sample,freq=FALSE,xlab=expression(nu),main="")
lines(density(nu.sample),type='l',col='red')
```



{#fig-capstone-histogram-earthquake-parameters-ar(1) fig-pos='H'}

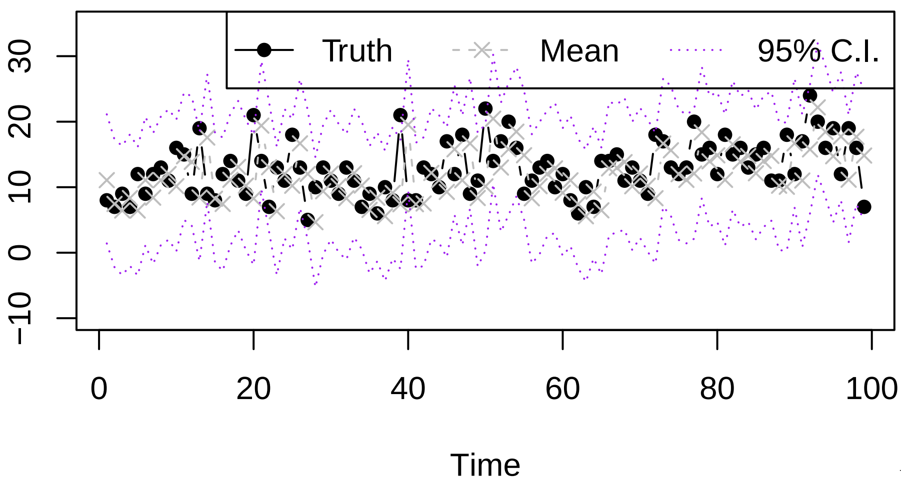
Model Checking by In-sample Point and Interval Estimation

To check whether the model fits well, we plot the posterior point and interval estimate for each point.

```r
## get in sample prediction
post.pred.y=function(s){

  beta.cur=matrix(phi.sample[s,],ncol=1)
```

```
  nu.cur=nu.sample[s]
  mu.y=t(Fmtx)%*%beta.cur
  sapply(1:length(mu.y), function(k){rnorm(1,mu.y[k],sqrt(nu.cur))})
}

y.post.pred.sample=sapply(1:5000, post.pred.y)
```

```
## show the result
summary.vec95=function(vec){
  c(unname(quantile(vec,0.025)),mean(vec),unname(quantile(vec,0.975)))
}

summary.y=apply(y.post.pred.sample,MARGIN=1,summary.vec95)

plot(Y,type='b',xlab='Time',ylab='',ylim=c(-10,35),pch=16)
lines(summary.y[2,],type='b',col='grey',lty=2,pch=4)
lines(summary.y[1,],type='l',col='purple',lty=3)
lines(summary.y[3,],type='l',col='purple',lty=3)
legend("topright",legend=c('Truth','Mean','95% C.I.'),lty=1:3,col=c('black','grey','purple')
       horiz = T,pch=c(16,4,NA))
```



{#fig-capstone-posterior-inference-earthquake-ar(1)-check fig-pos='H'}

```
cal_log_likelihood=function(phi,nu){
  mu.y = t(Fmtx)%*%phi
  log.lik=sapply(1:length(mu.y),function(k){dnorm(Y[k,1],mu.y[k],sqrt(nu),log=TRUE)})
  sum(log.lik)
}
```

```r
phi.bayes=colMeans(phi.sample)
nu.bayes=mean(nu.sample)
log.lik.bayes=cal_log_likelihood(phi.bayes,nu.bayes)
cat(log.lik.bayes)
```

```
-299.3061
```

```r
post.log.lik=sapply(1:5000, function(k){cal_log_likelihood(phi.sample[k,],nu.sample[k])})
E.post.log.lik=mean(post.log.lik)
cat("E.post.log.lik=",E.post.log.lik)
```

```
E.post.log.lik= -300.2802
```

```r
p_DIC=2*(log.lik.bayes-E.post.log.lik)
cat(" p_DIC=",p_DIC)
```

```
 p_DIC= 1.94824
```

```r
DIC=-2*log.lik.bayes+2*p_DIC
cat(" DIC=",DIC)
```

```
 DIC= 602.5086
```