Lab 10: Change Point Model

Wayne Stewart

March 27, 2018

Contents

Change point regression models

In this lab we will investigate another class of models which are extremely flexible in their application.

This topic can be seen as piecewise regression.

You can answer the questions and carry out the lab by filling in this RMD document.

Stagnant Water: Change point model

y_i is the log flow rate down an inclined channel, and x_i is the log height of stagnant surface layers of different surfactants. The rate of decline in flow rate seems to suddenly increase around $x = 0$.

The change point model

$$y_i \sim N(\mu_i, \sigma^2)$$

$$\mu_i = \beta_0 + \beta_1 x_i + \beta_2(x_i - \theta)I_{(1, x_i \geq \theta,\, 0\ else)}$$

The data

```
dataList = list(y = c(1.12, 1.12, 0.99, 1.03, 0.92, 0.90, 0.81, 0.83, 0.65, 0.67, 0.60,  0.59, 0.51, 0.4
    x = c(-1.39, -1.39, -1.08, -1.08, -0.94, -0.80, -0.63, -0.63, -0.25, -0.25, -0.12, -0.12, 0.01, 0.1
    N = 29)
```

Task 1

Plot the data using ggplot

```
df = data.frame(x = dataList$x, y=dataList$y)

head(df)
```

```
##       x    y
## 1 -1.39 1.12
## 2 -1.39 1.12
## 3 -1.08 0.99
## 4 -1.08 1.03
## 5 -0.94 0.92
## 6 -0.80 0.90
```

```
library(ggplot2)
```

```
## Warning in register(): Can't find generic 'scale_type' in package ggplot2 to
## register S3 method.
```

```
# place code here
```

Comment on the plot

Task 2

Inspect the jags model below and answer the questions

```
model {
  for (i in 1:N) {
    y[i]    ~ dnorm(mu[i], tau)
    mu[i]  <- alpha + beta[1]*x[i] + beta[2]*(x[i] - theta)
            * step(x[i] - theta)
  }
  tau       ~ dgamma(0.001, 0.001)
  alpha     ~ dnorm(0.0, 1.0E-6)
  for (j in 1:2) {
    beta[j] ~ dnorm(0.0, 1.0E-6)
  }
  sigma    <- 1/sqrt(tau)
  theta     ~ dunif(-1.3, 1.1)
}
```

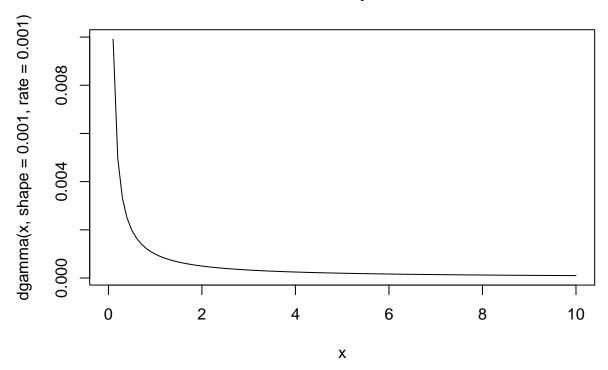**In the expression of the analytical model we used** $I_{(1, x_i \geq \theta, \, 0 \, else)}$

What jags function is used for I? `step()`

A prior is placed on τ

What is the mean and variance of the prior distribution for τ?

```
# What it looks like
curve(dgamma(x, shape = 0.001, rate = 0.001), xlim = c(0, 10))
title(main = 'Daniel Carpenter')
```

Daniel Carpenter


```r
# Mean
0.001/0.001
```

```
## [1] 1
```

```r
# Variance
0.001/0.001^2
```

```
## [1] 1000
```

```r
# Variance
```

Task 3

Make a Jags script

Make a complete Jags script to run the model. You can use the script below and alter it to fit.

```r
library(rjags)
#Define the model:
modelString = "
model {
```

```r
  for (i in 1:N) {
    y[i]     ~ dnorm(mu[i], tau)
    mu[i]  <- beta0 + beta[1]*x[i] + beta[2]*(x[i] - theta)
             * step(x[i] - theta)
  }
  tau       ~ dgamma(0.001, 0.001)
  beta0     ~ dnorm(0.0, 1.0E-6)

  for (j in 1:2) {
    beta[j] ~ dnorm(0.0, 1.0E-6)
  }
  sigma    <- 1/sqrt(tau)
  theta     ~ dunif(-1.3, 1.1)

  int2 <- beta0 - beta[1]*theta
  slope2 <- beta[1] + beta[2]
}
" # close quote for modelString
writeLines( modelString , con="TEMPmodel.txt" )

initsList = list(tau = 1, beta0 = 2, beta = c(1, 2), theta = 1)

# Run the chains:
jagsModel = jags.model( file="TEMPmodel.txt" , data=dataList , inits=initsList ,
                        n.chains=3 , n.adapt=500 )

# Show how the model is ran - realSlicer is slow. Conjugate is fast
list.samplers(jagsModel)

update( jagsModel , n.iter=500 )
codaSamples = coda.samples( jagsModel , variable.names=c("beta0", "beta1"),
                            n.iter=33340 )
save( codaSamples , file=paste0("lab10","Mcmc.Rdata") )

# Output of model
summary(codaSamples)

# Plot the model output
library(ggmcmc)
s = ggs(codaSamples)
ggs_density(s)

ggs_crosscorrelation(s)
```

Task 4

Run the model and make point and interval estimates.

Interpret these estimates.

Task 5

Plot the estimating lines onto the data