

A HIERARCHICAL POISSON MODEL FOR CLAIM FREQUENCY DATA

Hing Wu Jing Zhao

December 8, 2007

1 Introduction

This project is about two ways to predict future claim count in a hierarchical model. The first method is the non-parametric Buhlmann-Straub credibility model. The second method is Bayesian Graphical Modelling method. We tried to compare the two methods by looking at the statistics such as mean, standard error and credible set.

2 Credibility Theory

In actuarial field the term credibility was originally attached to experience rating formulas that were convex combinations (weighted averages) of individual and class estimates of the individual risk premium. Credibility theory, thus, was the branch of actuarial mathematics that discovered model-based principles for construction of such formulas. The risk premium m , defined as the expected claims expenses per unit of risk exposed, for an individual risk selected from a portfolio (class) of similar risks. Advocating the combined use of individual risk experience and class risk experience. The premium rate is a weighted average of the form

$$M = m * z + (1 - z) * u \tag{1}$$

where m is the observed mean claim amount per unit of risk exposed for the individual contract and u is the corresponding overall mean in the insurance portfolio. In the language of modern credibility theory, it is a function $m(\Theta)$ of a random element Θ representing the unobservable characteristics of the individual risk. The random nature of Θ expresses the notion of heterogeneity; the individual risk is a random selection from a portfolio of similar but not identical risks, and the distribution of Θ describes the variation of individual risk characteristics across the portfolio. The weight z was soon to be named credibility factor since it measures the amount of credence attached to the individual experience, and M was called the credibility premium.

2.1 Bühlmann-Straub credibility model

In this article, suppose that an insurance company has seven groups of policyholder's loss frequency counts for 50 months as shown in Table 1. The problem is to describe the loss frequencies anticipated in the 51th month. One approach is to make a straightforward application of the Buhlmann-Straub credibility model as described in Herzog (1996, chap. 7) and Klugman, Panjer, and Willmot (1998, sections 5.4.4 and 5.5.1). The non-parametric method described in these texts can be used to estimate the necessary population parameters and credibility factors. In terms of the notation found in Herzog (1996) these calculation of values like m , a , v , z , m_i , u yield the predicted count frequency of the 51th month.

$$u = E(E(X_j|\Theta)) \tag{2}$$

$$v = E(Var(X_j|\Theta)) \tag{3}$$

$$a = Var(E(X_j|\Theta)) \tag{4}$$

$$z = m_i/(m_i - v/a) \tag{5}$$

Table 1:

	Policy Group	Month1	Month2	Month 51
Claim Counts	1	0	4	?
No. in group		236	191	168
Claim Counts	2	28	7	?
No. in group		252	186	184
Claim Counts	3	1	6	?
No. in group		208	184	191
Claim Counts	4	23	2	?
No. in group		243	187	262
Claim Counts	5	0	1	?
No. in group		178	202	186
Claim Counts	6	0	6	?
No. in group		160	181	222
Claim Counts	7	18	4	?
No. in group		207	155	202

3 Bayesian Graphical Modeling by Markov Chain Monte Carlo Simulations

Another approach would be to treat the unknown parameters as random variables and then develop the posterior and predictive distributions of interest using the Bayesian method.

consider a vector random variable $U (U_1, . . . , U_k)$ with joint distribution $f(U_1, . . . , U_k)$. In a Bayesian context, some of these variables are model parameters while others may represent unobserved past or future data. Suppose $f(U)$ has a complicated and analytically intractable form, and the expected value of some integrable function $h(U)$ is sought. Even if this calculation can not be performed analytically, it is still possible that the probabilistic model associated with $f(U)$ may be simple enough to permit independent random draws $U(t), t = 1, . . . , n$, from it. If this is the case, then the desired expectation can be approximated using

$$E[h(u)] \approx \frac{1}{n} \sum_{i=1}^n h(u^t)$$

For instance, the population mean can be estimated using the sample mean. This procedure is called Monte Carlo integration. Unfortunately, many complicated models will not readily permit independent random draws. In this case a MCMC simulation method can be used instead. The main idea behind a MCMC method is to simulate realizations from a Markov chain that has $f(U)$ as its stationary distribution.

we adopt the following complete probability model for the data appearing in Table 1. Let X_{ij} and P_{ij} denote the number of claims and number of people, respectively, for the i -th group policyholder in the j -th policy month. We assume that all X_{ij} are conditionally independent and that $X_{ij} \sim Poisson(P_{ij}\theta_i)$, for all i and j . The parameters λ_i denote the expected number of losses per unit of exposure for group policyholder i . Given α and β , the θ_i are assumed to be conditionally independent with $\theta_i \sim gamma(\alpha, \beta)$, for all i . These gamma distributions are parameterized to have mean α/β . We complete this model by letting $\alpha \sim gamma(5, 5)$ and $\beta \sim gamma(25, 1)$. These last two distributions were selected arbitrarily. However, they imply that each u_i has a prior mean and standard deviation approximately equal to 0.041 and 0.048 respectively, which is not unreasonable for the present context.

We start with a consideration of the following slight extension of the full probability model:

$$\begin{aligned} X_{ij} &\sim Poisson(\lambda_{ij}), \\ \lambda_{ij} &= P_{ij}\theta_i, \\ \theta_i &\sim gamma(\alpha, \beta), \\ \alpha &\sim gamma(5, 5), \\ \beta &\sim gamma(25, 1), \end{aligned}$$

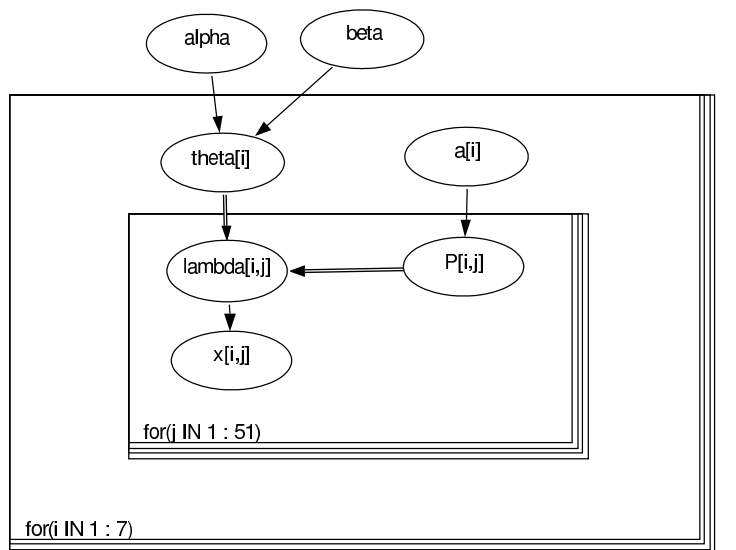
$$P_{ij} \sim \text{Poisson}(a_i),$$

$$a_i \sim \text{Uniform}(100, 300),$$

For the data in Table 1, the values taken on by the indices are $i = 1, 2, \dots, 7$ and $j = 1, \dots, 51$. This is the form of the model we will implement using BUGS/WinBUGS. We assume conditional independence for the variables appearing on the left-hand side of each line given the model parameters appearing on the right. So the variables X_{ij} are conditionally independent given the values of λ_{ij} . The first five lines describe the original hierarchical Poisson model. The other three lines describe a simple probability model for the number of people.

The model has the particularly simple graphical representation given in Figure 1. This graph emphasizes the main qualitative features of the model without the use of cumbersome algebraic formulas. Nodes in the graph denote the data and parameters appearing in the model. Figure 1 is said to be a directed graph as each link between nodes is an arrow. The directed links are of two types, stochastic and logical. Stochastic links or dependencies are indicated with solid arrows. Logical links or dependencies are indicated with hollow ones. The arrows indicate the conditional independence assumptions of the model. They also suggest a parent-child analogy for the linked nodes. For example, we can say that α is a parent of θ_1 , and P_{15} is a child of a_1 . Given its parent nodes, denoted as $\text{parents}[v]$, each node v is independent of all other nodes except for the descendants of v .

Figure 1: Graphical Poisson Model



4 Conclusion

The following table is the comparison of the results of the two approaches.

Table 2: Estimated Summary Statistics

Parameter	Mean		SD		95%	
	Bays.	Cred.	Bays.	Cred.	CS	PI
x[1,51]	6.377	6.585	2.544	1.959	(2,12)	(1.5,11.7)
x[2,51]	10.4	9.459	3.193	2.146	(5,17)	(3.8,15.1)
x[3,51]	6.181	6.767	2.496	2.219	(2,12)	(1.0,12.6)
x[4,51]	14.93	13.468	3.832	3.065	(8,23)	(5.5,21.9)
x[5,51]	4.986	5.951	2.241	2.180	(1,10)	(0.3,11.7)
x[6,51]	9.763	9.509	3.116	2.606	(4,16)	(2.7,16.4)
x[7,51]	7.327	7.693	2.684	2.364	(3,13)	(1.5,13.9)

From the table, it can be seen that the prediction for the bayesian and credibility theory approach are quite close. The standard error for credibility approach is a bit smaller than that of bayesian approach.

5 Appendix(R code and MCMC Model)

5.1 R code

```
stim <- function(n,m) {
x <- numeric()
P <- numeric()
for (i in 1:n) {
a <- 50
b <- 0.25
p <-round(rgamma(1,a,b))
a2 <- rgamma(1,5,5)
b2 <- rgamma(1,25,1)
u <-rgamma(1,a2,b2)
l <- p*u
x <- c(x,rpois(1,1))
P <- c(P,p)
}
x <- matrix(x, nrow = m)
P <- matrix(P, nrow = m)
list ( x.value = x, m.value = P)
}

fut <- function(m) {
x <- numeric()
P <- numeric()
n <- m
for (i in 1:n) {
a <- 50
b <- 0.25
p <-round(rgamma(1,a,b))
a2 <- rgamma(1,5,5)
b2 <- rgamma(1,25,1)
u <-rgamma(1,a2,b2)
l <- p*u
x <- c(x,rpois(1,1))
P <- c(P,p)
}
x <- matrix(x, nrow = m)
P <- matrix(P, nrow = m)
list ( x.value = x, m.value = P)
}

BS <- function(x,m,s) {
r <- nrow(x)
```

```

n <- ncol(x)
ans <- numeric()
va <- numeric()
sde <- numeric()

mm <- 0
mu <- 0
a <- 0

for ( i in 1:r ) {
  for (j in 1:n) {
    if (!is.na(m[i,j])) {
      mm <- mm + m [i,j]
    }
  }
}

for ( i in 1:r ) {
  for (j in 1:n) {
    if (!is.na(x[i,j])) {
      mu <- mu + x[i,j]
    }
  }
}

mu <- mu / mm

v <- 0
for ( i in 1:r ) {
  tm <- 0
  for (j in 1:n) {
    if (!is.na(x[i,j])) {
      tm <- tm + x[i,j]
    }
  }
  mi <- 0
  for (j in 1:n) {
    if (!is.na(m[i,j])) {
      mi <- mi + m[i,j]
    }
  }
  tm <- tm/mi
  for (j in 1:n) {
    if (!is.na(m[i,j])) {
      v <- v + m[i,j] * (x[i,j] / m[i,j]-tm)^2
    }
  }
}

```

```

}
for (j in 1:n) {
  if (is.na(m[i,j])) {
    a <- a +1
  }
}
}
v <- v/(r * (n-1) - a)

ms <- 0
a <- 0
for ( i in 1:r ) {
  tm <- 0
  for (j in 1:n) {
    if (!is.na(x[i,j])) {
      tm <- tm + x[i,j]
    }
  }
  mi <- 0
  for (j in 1:n) {
    if (!is.na(m[i,j])) {
      mi <- mi + m[i,j]
    }
  }
  ms <- ms + mi^2
  tm <- tm/mi
  a <- a + mi * (tm-mu)^2
}
a <- ( a- v * (r-1))/( mm-ms /mm)

k <- v/a
mi <- 0

for ( i in 1:r ) {

mi <- 0
for (j in 1:n) {
  if (!is.na(m[i,j])) {
    mi <- mi + m[i,j]
  }
}

z <- mi/(mi+k)

tm <- 0

```

```

for (j in 1:n) {
  if (!is.na(x[i,j])) {
    tm <- tm + x[i,j]
  }
}

tm <- tm/mi
ans <- c(ans,(tm *z + (1-z) * mu)*s[i])
va <- c(va,(a+v/mi)*s[i]^2)
sde <- c(sde,(tm *z + (1-z) * mu)*s[i]-qt(0.975,r-1)*sqrt((a+v/mi)*s[i]^2)
*sqrt(1+1/r),(tm *z + (1-z) * mu)*s[i]+qt(0.975,r-1)
*sqrt((a+v/mi)*s[i]^2)*sqrt(1+1/r))
}

list ( Expected.Value = ans, Standard.Dev = sqrt(va) ,
Predicted.C.I = matrix(sde, ncol = 2, byrow = TRUE))
}

ans <-stim(350,7)
x <- ans$x.value
m <- ans$m.value
f <- fut(7)
s <- f$m.value

value <- BS (x,m,s)
value

```

5.2 MCMC model

- Mode code

```

model
{
  for( j in 1 : 51 ) {
    for( i in 1 : 7 ) {
      x[i , j] ~ dpois(lambda[i , j])
    }
  }
  for( j in 1 : 51 ) {
    for( i in 1 : 7 ) {
      lambda[i , j] <- p[i , j] * theta[i]
    }
  }
  for( i in 1 : 7 ) {

```

```

    theta[i] ~ dgamma(alpha,beta)
  }
  for( j in 1 : 51 ) {
    for( i in 1 : 7 ) {
      p[i , j] ~ dpois(a[i])
    }
  }
  for( i in 1 : 7 ) {
    a[i] ~ dunif(100,300)
  }

  alpha ~ dgamma(5,5)
  beta ~ dgamma(25,1)
}

```

Data

```

list( x = structure(
  .Data = c(0 , 4 , 3 , 4 , 4 , 9 , 14 , 0 , 0 , 10 ,
5 , 3 , 0 , 3 , 7 , 20 , 2 , 0 , 2 , 9 ,
9 , 1 , 11 , 4 , 21 , 0 , 5 , 3 , 0 , 4 ,
12 , 0 , 0 , 18 , 9 , 13 , 10 , 2 , 12 , 11 ,
4 , 0 , 32 , 5 , 1 , 18 , 37 , 15 , 25 , 0 , NA ,
28 , 7 , 4 , 25 , 28 , 1 , 0 , 20 , 3 , 8 , 10 ,
1 , 0 , 0 , 34 , 10 , 39 , 17 , 9 , 7 , 1 ,
13 , 7 , 4 , 0 , 2 , 3 , 12 , 13 , 6 , 7 ,
10 , 3 , 27 , 25 , 12 , 0 , 2 , 21 , 0 , 13 ,
2 , 1 , 11 , 5 , 13 , 0 , 60 , 27 , 18 , NA ,
1 , 6 , 2 , 9 , 6 , 15 , 22 , 2 , 0 , 5 , 2 ,
0 , 0 , 0 , 0 , 9 , 10 , 9 , 9 , 0 , 1 ,
2 , 3 , 3 , 25 , 3 , 5 , 3 , 17 , 5 , 4 ,
1 , 0 , 45 , 0 , 4 , 25 , 3 , 2 , 0 , 2 ,
4 , 0 , 16 , 1 , 1 , 4 , 22 , 3 , 19 , NA ,
23 , 2 , 7 , 1 , 2 , 9 , 10 , 28 , 0 , 68 , 0 ,
14 , 2 , 26 , 12 , 4 , 26 , 6 , 0 , 7 , 36 ,
3 , 12 , 9 , 0 , 21 , 1 , 1 , 1 , 7 , 3 ,
2 , 2 , 6 , 40 , 5 , 27 , 0 , 12 , 1 , 17 ,
6 , 2 , 30 , 11 , 0 , 2 , 1 , 8 , 48 , NA ,
0 , 1 , 3 , 9 , 0 , 0 , 2 , 16 , 2 , 11 , 2 ,
5 , 0 , 4 , 9 , 2 , 5 , 9 , 1 , 24 , 3 ,
4 , 0 , 15 , 15 , 17 , 3 , 0 , 12 , 1 , 1 ,
9 , 8 , 4 , 10 , 2 , 1 , 14 , 0 , 0 , 5 ,
7 , 4 , 5 , 1 , 2 , 4 , 1 , 8 , 0 , NA ,
0 , 6 , 7 , 0 , 2 , 15 , 14 , 3 , 1 , 14 , 6 ,
2 , 6 , 0 , 4 , 7 , 0 , 0 , 3 , 3 , 5 ,
0 , 1 , 21 , 5 , 0 , 31 , 55 , 8 , 1 , 32 ,
24 , 4 , 11 , 0 , 4 , 3 , 32 , 21 , 9 , 1 ,

```

```

8 , 13 , 0 , 24 , 8 , 0 , 1 , 7 , 0 , NA ,
18 , 4 , 1 , 2 , 16 , 30 , 3 , 6 , 0 , 8 , 35 ,
1 , 4 , 0 , 10 , 2 , 1 , 2 , 3 , 8 , 1 ,
3 , 18 , 14 , 1 , 23 , 9 , 3 , 3 , 13 , 0 ,
11 , 0 , 3 , 2 , 1 , 19 , 5 , 11 , 2 , 0 ,
0 , 1 , 9 , 4 , 14 , 2 , 8 , 5 , 17 , NA) ,
.Dim = c( 7 , 51 ) ) ,

p = structure(
.Data = c(
236 , 191 , 219 , 210 , 196 , 197 , 178 , 174 , 179 , 277 , 214 ,
179 , 183 , 203 , 158 , 183 , 202 , 219 , 246 , 169 , 177 ,
195 , 209 , 272 , 223 , 245 , 165 , 263 , 148 , 174 , 191 ,
154 , 175 , 165 , 188 , 186 , 240 , 182 , 175 , 198 , 151 ,
177 , 216 , 255 , 144 , 259 , 257 , 171 , 252 , 216 , 168 ,
252 , 186 , 232 , 200 , 172 , 177 , 152 , 180 , 194 , 193 , 216 ,
258 , 161 , 137 , 218 , 171 , 212 , 185 , 205 , 183 , 191 ,
235 , 190 , 186 , 213 , 241 , 194 , 178 , 200 , 160 , 182 ,
149 , 200 , 210 , 216 , 215 , 184 , 169 , 173 , 222 , 202 ,
272 , 259 , 218 , 189 , 198 , 197 , 239 , 217 , 237 , 184 ,
208 , 184 , 205 , 198 , 207 , 177 , 277 , 204 , 186 , 173 , 216 ,
238 , 227 , 220 , 223 , 191 , 218 , 179 , 200 , 185 , 158 ,
191 , 166 , 161 , 215 , 274 , 188 , 197 , 254 , 230 , 188 ,
203 , 217 , 231 , 158 , 202 , 208 , 244 , 150 , 179 , 165 ,
226 , 240 , 126 , 202 , 184 , 199 , 268 , 252 , 249 , 191 ,
243 , 187 , 222 , 173 , 192 , 185 , 204 , 172 , 165 , 187 , 215 ,
195 , 210 , 227 , 179 , 240 , 214 , 216 , 191 , 216 , 237 ,
179 , 267 , 188 , 156 , 253 , 170 , 162 , 226 , 185 , 206 ,
176 , 206 , 188 , 194 , 224 , 204 , 169 , 207 , 175 , 151 ,
170 , 173 , 218 , 148 , 221 , 183 , 172 , 165 , 259 , 262 ,
178 , 202 , 191 , 190 , 204 , 164 , 197 , 203 , 171 , 145 , 165 ,
191 , 200 , 253 , 191 , 201 , 192 , 233 , 145 , 183 , 180 ,
213 , 204 , 228 , 189 , 244 , 205 , 195 , 178 , 145 , 184 ,
208 , 199 , 220 , 226 , 159 , 162 , 189 , 177 , 187 , 167 ,
219 , 229 , 172 , 246 , 229 , 215 , 211 , 193 , 194 , 186 ,
160 , 181 , 143 , 168 , 204 , 147 , 206 , 214 , 177 , 205 , 216 ,
232 , 234 , 231 , 203 , 182 , 145 , 197 , 194 , 170 , 188 ,
179 , 204 , 217 , 224 , 187 , 188 , 223 , 239 , 194 , 181 ,
185 , 180 , 194 , 197 , 202 , 186 , 208 , 218 , 190 , 199 ,
225 , 179 , 163 , 159 , 217 , 203 , 187 , 171 , 184 , 222 ,
207 , 155 , 211 , 170 , 184 , 236 , 222 , 165 , 182 , 158 , 231 ,
179 , 160 , 198 , 204 , 159 , 187 , 194 , 166 , 237 , 194 ,
203 , 193 , 178 , 186 , 256 , 173 , 187 , 195 , 220 , 214 ,
235 , 228 , 137 , 224 , 193 , 183 , 174 , 211 , 205 , 191 ,
223 , 190 , 165 , 209 , 149 , 240 , 208 , 222 , 241 , 202) ,
.Dim = c( 7 , 51 ) ) )

```

Inits

```
list( alpha = 2, beta = 10,  
      theta=c(  
1,1,1,1,1,1,1) )  
list( alpha = 1, beta = 2,  
      theta=c(  
0.05, 0.05, 0.05, 0.05, 0.05, 0.05) )  
list( alpha = 1, beta = 25,  
      theta=c(  
10,10,10,10,10,10,10) )
```

- MCMC Model output and diagnostics.

References

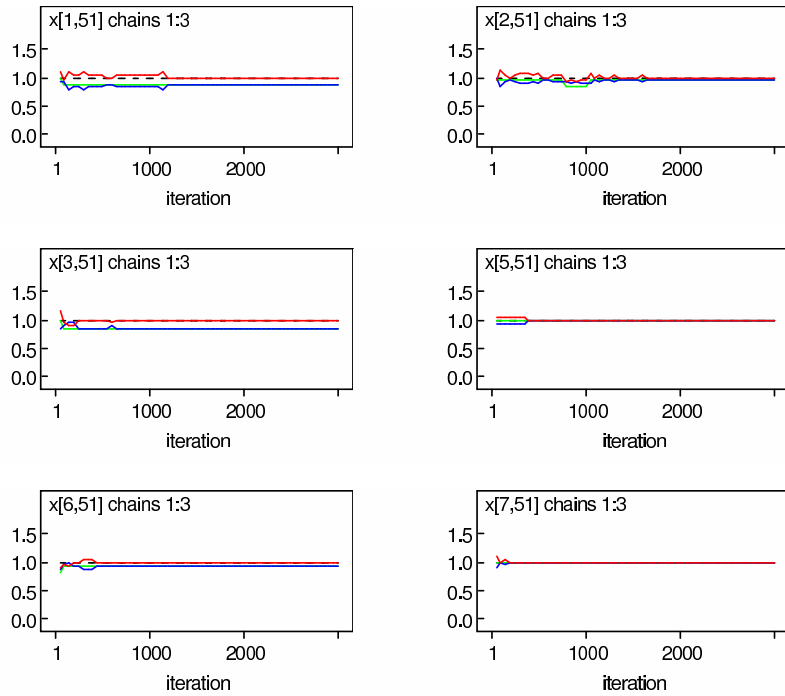
- [1] Ragnar Norberg, <http://stats.lse.ac.uk/norberg/links/papers/CRED-eas.pdf> Credibility Theory
- [2] David P. M. Scollnik, "ACTUARIAL MODELING with MCMC and BUGS"

Figure 2: Model stat. and Diagnostic

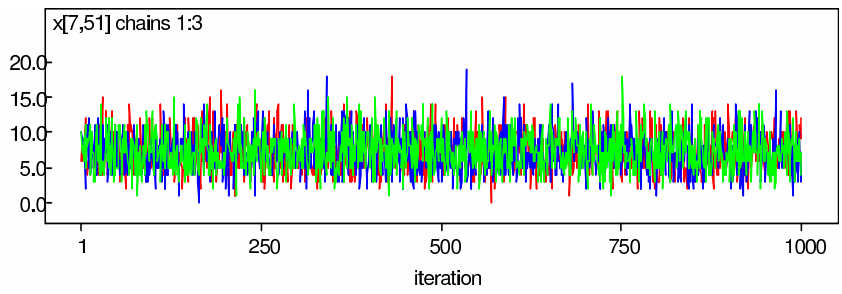
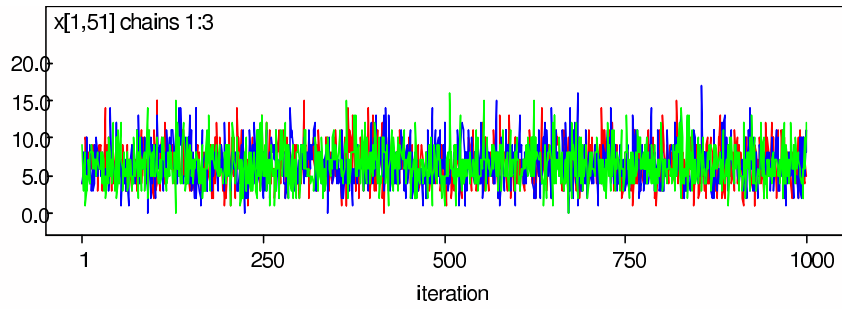
Node Statistics

node	mean	sd	MC error	2.5%	median	97.5%	start	sample
alpha	1.327	0.3327	0.005079	0.7506	1.299	2.046	1200	5403
beta	26.6	4.875	0.07225	17.8	26.23	37.21	1200	5403
theta[1]	0.03799	0.001959	2.543E-5	0.03423	0.03796	0.04187	1200	5403
theta[2]	0.05673	0.002341	3.19E-5	0.05209	0.05669	0.06137	1200	5403
theta[3]	0.03226	0.001761	2.324E-5	0.02889	0.03224	0.03567	1200	5403
theta[4]	0.05683	0.002411	3.403E-5	0.05221	0.05681	0.06179	1200	5403
theta[5]	0.02678	0.001642	2.202E-5	0.02367	0.02676	0.03015	1200	5403
theta[6]	0.04364	0.002102	2.776E-5	0.03964	0.0436	0.04787	1200	5403
theta[7]	0.03625	0.001925	2.949E-5	0.03259	0.03624	0.04014	1200	5403
x[1,51]	6.377	2.544	0.03636	2.0	6.0	12.0	1200	5403
x[2,51]	10.4	3.193	0.04622	5.0	10.0	17.0	1200	5403
x[3,51]	6.181	2.496	0.03471	2.0	6.0	12.0	1200	5403
x[4,51]	14.93	3.832	0.05393	8.0	15.0	23.0	1200	5403
x[5,51]	4.986	2.241	0.02947	1.0	5.0	10.0	1200	5403
x[6,51]	9.763	3.116	0.04432	4.0	10.0	16.0	1200	5403
x[7,51]	7.327	2.684	0.03519	3.0	7.0	13.0	1200	5403

Galman Rubin Statistics



History Trace



Autocorrelation Diagnostics

