MATH 559: BAYESIAN THEORY AND METHODS

AUXILIARY VARIABLE METHODS AND MISSING DATA

Auxiliary variable methods can also be used to handle *missing data* problems. Missing data can arise in problems due to drop-out, data handling issues or censoring/truncation. The missingness can occur

- *completely at random* (independently of the observed and unobserved data)
- *at random* (dependent on the observed data, but independent of the unobserved data)
- *not at random* (dependent on the observed data and unobserved data)

Missingness can occur in a *response* variable Y or *covariate* X. We denote the missing values by $X^\star$ and $Y^\star$.

We consider a Bayesian formulation of the missing data inference problem; maximum likelihood estimation can be handled using the EM algorithm. We consider the joint likelihood of the observed and unobserved data, and treat the unobserved values as quantities to be updated during the MCMC.

The parameter vector θ can now also include parameters in the stochastic missingness mechanism. In the case of i.i.d. data, where the missing data are considered independent of the observed data, the **complete** data $(\mathbf{Y}, \mathbf{Y}^\star)$ have likelihood

$$\mathcal{L}_n^\star(\theta) = \left\{\prod_{i=1}^{n} f_Y(y_i;\theta)\right\}\left\{\prod_{i=1}^{n^\star} f_Y(y_i^\star;\theta)\right\}$$

In this case, a Gibbs sampler can be used to update the quantities $(\theta, \mathbf{Y}^\star)$ from their full conditionals. For θ,

$$\pi_n(\theta|\mathbf{y}, \mathbf{y}^\star) \propto \mathcal{L}_n^\star(\theta)\pi_0(\theta)$$

that is, we perform Bayesian inference using the complete data likelihood. For elements of $\mathbf{Y}^\star$, the full conditional distribution is merely the density $f_Y(y_i^\star;\theta)$. Most typically, however, this kind of missing data occurs due to censoring, that is, we have information that

$$Y_i^\star > c_i$$

say. In this case, the full conditional distribution used in the Gibbs sampler must respect the censoring constraint, that is, the full conditional is

$$f_Y^\star(y_i^\star;\theta,c_i) \;\propto\; \mathbb{1}_{(c_i,\infty)}(y_i^\star) f_Y(y_i^\star;\theta)$$

Conditional on $(\mathbf{y}, \mathbf{y}^\star)$, the full conditional distribution of θ is not affected by the censoring, so is identical to above.

Example: Censored data and the Exponential distribution.
The exponential distribution has pdf and cdf

$$f(y;\lambda) = \lambda\exp\{-\lambda y\} \qquad F(y;\lambda) = 1-\exp\{-\lambda y\} \qquad y>0$$

for parameter $\lambda > 0$. Suppose that n data are available, but n_1 of them are censored at $c > 0$; without loss of generality assume that the observations labelled $i = 1, \ldots, n_0$ are uncensored, where $n_0 = n - n_1$. In this case the likelihood for λ is

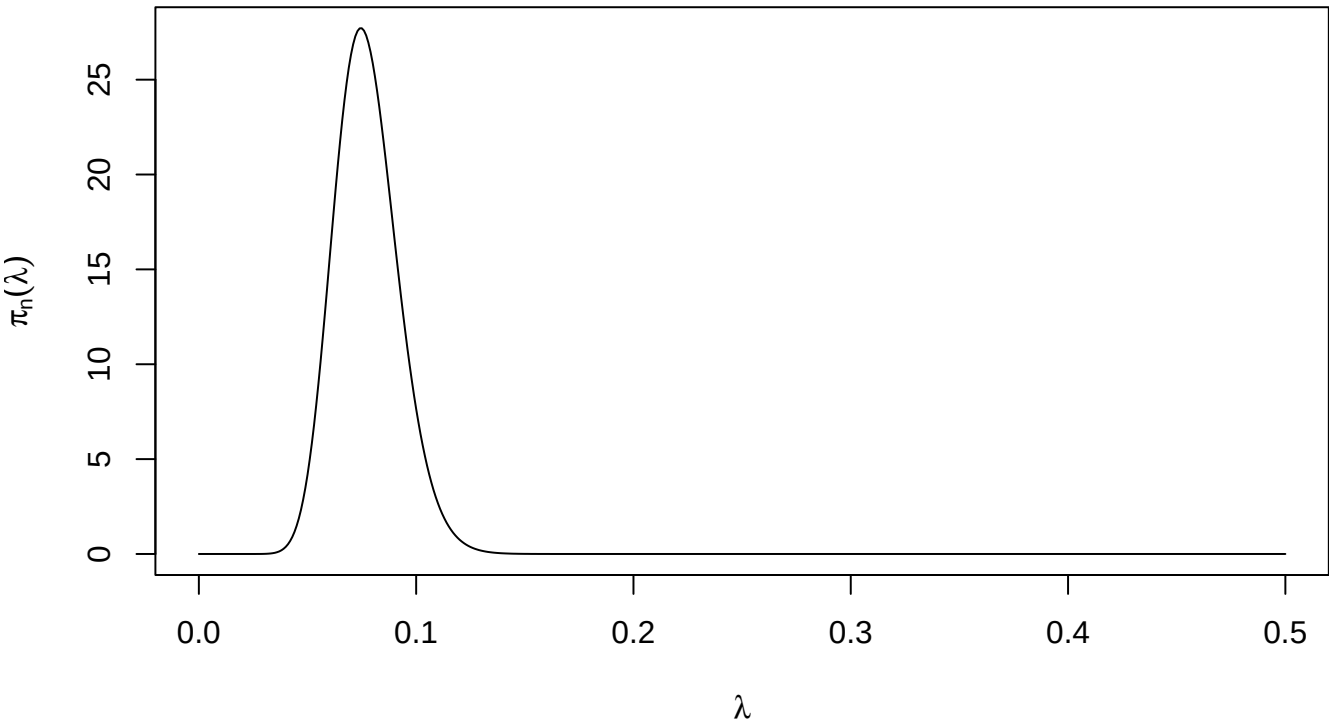
$$\begin{aligned}\mathcal{L}_n(\lambda) &= \prod_{i=1}^{n_0} f(y_i;\lambda) \prod_{i=n_0+1}^{n} (1-F(c;\lambda)) \\ &= \lambda^{n_0}\exp\left\{-\lambda\sum_{i=1}^{n} y_i\right\}\exp\{-\lambda n_1 c\} \\ &= \lambda^{n_0}\exp\left\{-\lambda\left(\sum_{i=1}^{n} y_i + n_1 c\right)\right\}.\end{aligned}$$

Using a conjugate $Gamma(\alpha,\beta)$ prior, we see that the posterior distribution is $Gamma(\alpha_n,\beta_n)$ where

$$\alpha_n = n_0 + \alpha \qquad \beta_n = \sum_{i=1}^{n_0} y_i + n_1 c + \beta$$

which can be plotted as follows for an example data set with $n = 50$.

```r
set.seed(1410)
lam<-0.1
n<-50
cval<-10
y<-rexp(n,lam)
z<-rep(0,n);z[y > cval]<-1
y[y>cval]<-cval
n1<-sum(z)
n0<-n-n1
xv<-seq(0,0.5,by=0.001)
al<-1;be<-1
pi.n<-dgamma(xv,n0+al,sum(y[z==0])+n1*cval+be)
par(mar=c(4,4,1,0))
plot(xv,pi.n,type='l',xlab=expression(lambda),ylab=expression(pi[n](lambda)))
```



Here $n_1 = 23$. We can reproduce the same posterior by augmenting the missing data; we introduce

$$y^\star_{n_0+1}, \ldots, y^\star_n,$$

as the unobserved "true" versions of the censored values. Conditional on the "complete" data, the posterior distribution is straightforwardly obtained as

$$\pi^\star_n(\lambda) \equiv Gamma(\alpha^\star_n, \beta^\star_n)$$

where

$$\alpha^\star_n = n + \alpha \qquad \beta^\star_n = \sum_{i=1}^{n_0} y_i + \sum_{i=n_0+1}^{n} y^\star_i + \beta.$$

To sample the unobserved true values from their full conditional distribution, we have that

$$f^\star(y^\star; \lambda, c) \;\propto\; \mathbb{1}_{(c,\infty)}(y^\star) f(y^\star; \lambda)$$

with corresponding cdf

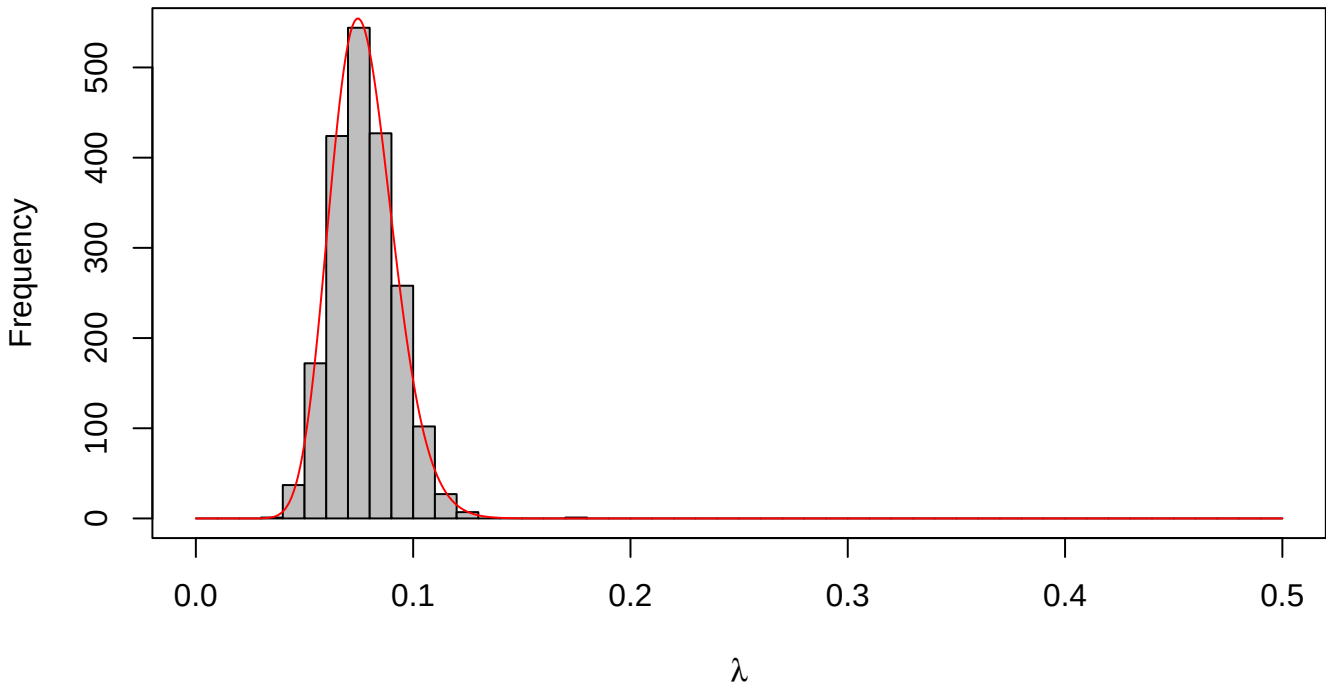
$$F^\star(y^\star; \lambda, c) = \frac{F(y^\star; \lambda) - F(c; \lambda)}{1 - F(c; \lambda)} = 1 - \frac{\exp\{-\lambda y^\star\}}{\exp\{-\lambda c\}},$$

We may therefore sample the unobserved true values using *cdf-inversion* or *probability integral transform* approach: it is well-known that if X is a continuous random variable, with $X \sim F_X(x)$, then $U = F_X(X) \sim Uniform(0,1)$. This allows simulation of X by generating $U \sim Uniform(0,1)$, and setting $X = F_X^{-1}(U)$. Here then we have that if u is a sample from a $Uniform(0,1)$, we compute

$$y^\star = -\frac{1}{\lambda}\log\left((1-u)\exp\{-\lambda c\}\right)$$

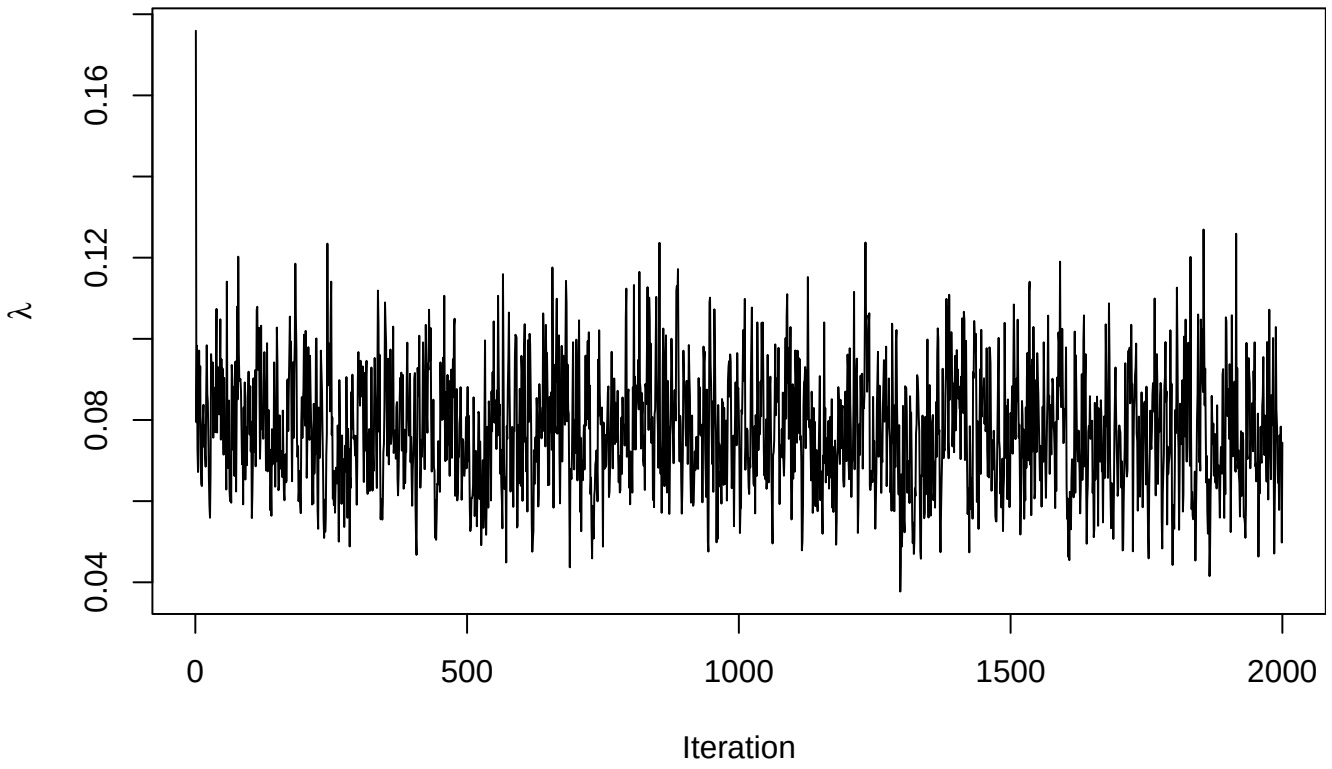
and do this independently for all censored values. If we proceed in this way, using a Gibbs sampler strategy, then the original posterior is recovered.

```
nits<-2000
old.lam<-1/mean(y[z==0])
yall<-y                                #Starting value for the augmented data
lambda.samp<-rep(0,nits)
y.samp<-matrix(0,nrow=nits,ncol=n)
for(iter in 1:nits){
    old.lam<-rgamma(1,n+al,sum(yall)+be)    #Sample lambda given the augmented values
    yall[z==1]<--log((1-runif(sum(z==1)))*exp(-old.lam*cval))/old.lam  #sample the y values given lambda
    lambda.samp[iter]<-old.lam
    y.samp[iter,]<-yall
}
par(mar=c(4,4,2,0))
hist(lambda.samp,br=seq(0,0.5,by=0.01),col='gray',main='',xlab=expression(lambda));box()
lines(xv,pi.n*nits*0.01,col='red')
```

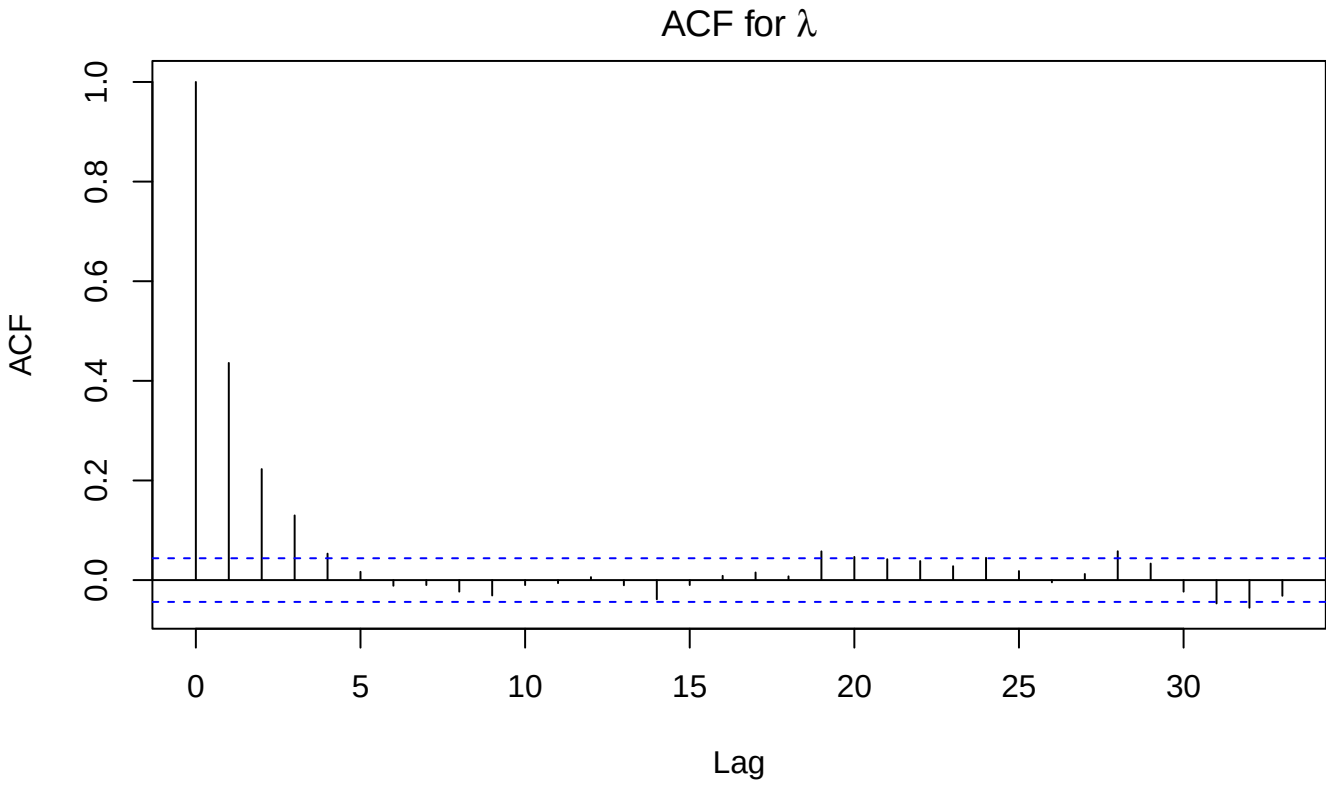


The λ samples are being collected with low autocorrelation.

```
par(mar=c(4,4,1,0))
plot(lambda.samp,type='l',xlab='Iteration',ylab=expression(lambda))
```




```
par(mar=c(4,4,2,0))
acf(lambda.samp,main=' ');title(expression(paste('ACF for ',lambda)),line=1)
```



Example: Censored data and the Burr distribution.
The *Burr* distribution is a three-parameter probability distribution on $\mathbb{R}^+$ with cumulative distribution function

$$F(y;\alpha,\gamma,\lambda) = 1 - \frac{1}{(1+\lambda x^{\alpha})^{\gamma}} \qquad y > 0$$

for parameters $\alpha, \lambda, \gamma > 0$. This distribution is useful for modelling right-skewed data.

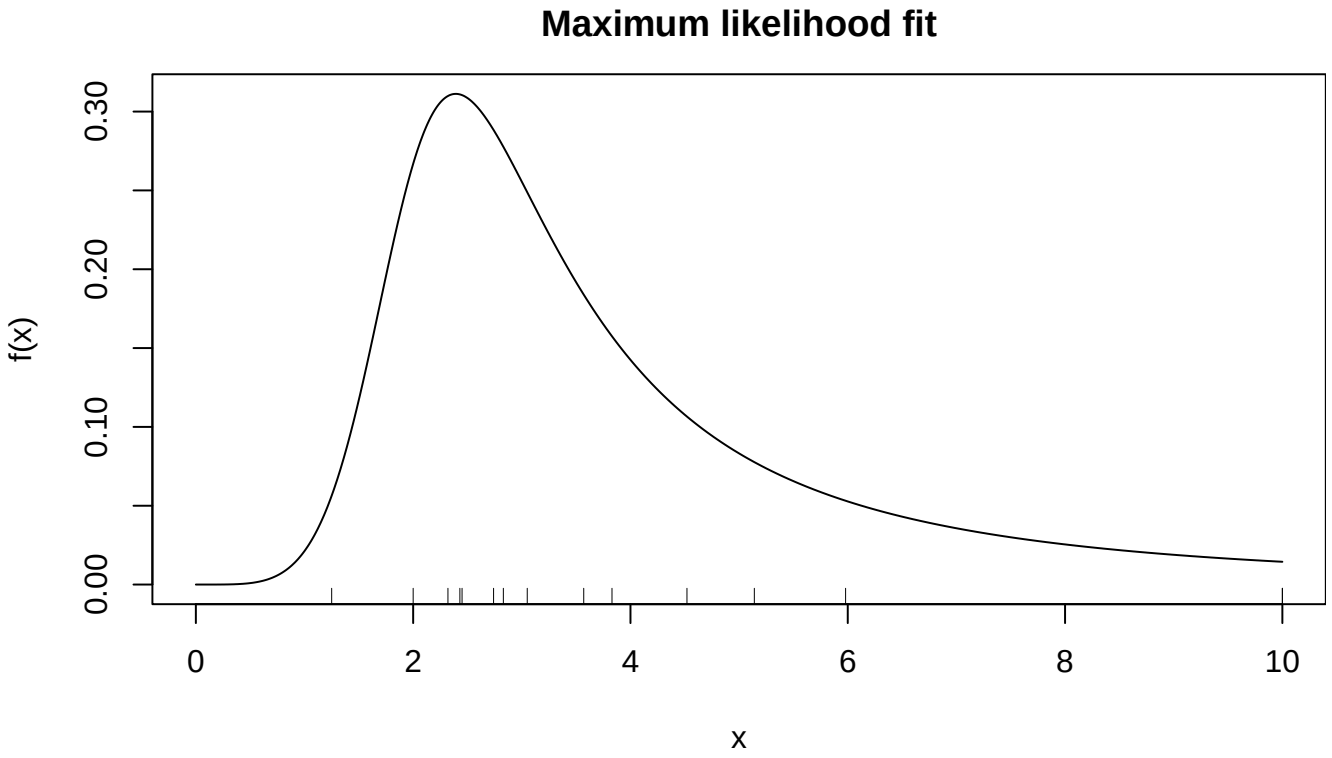
Suppose that the following data are to be modelled using a Burr distribution where underlined observations are

2.83	2.00	5.14	3.83	4.52	1.25	3.57	2.43
2.45	5.98	2.32	3.05	<u>10.00</u>	<u>10.00</u>	2.74	

right-censored (that is, we know that the actual value for that datum is **greater than** the underlined value). The likelihood contribution of a right-censored value y drawn from distribution $F(.)$ is

$$P(Y > y) = 1 - F(y) = \frac{1}{(1+\lambda y^{\alpha})^{\gamma}}.$$

```
Burr.pdf<-function(xvec,a1,g1,l1){
        yvec<-a1*g1*l1*xvec^(a1-1)*exp(-(g1+1)*log(1+l1*xvec^a1))
        return(yvec)
}

#Data
y<-c(2.83,2.00,5.14,3.83,4.52,1.25,3.57,2.43,2.45,5.98,2.32,3.05,10,10,2.74)
n<-length(y)
z<-rep(0,n);z[y==10]<-1   #Censoring indicator

#Likelihood
Burr.like<-function(yv,zv,a1,g1,l1){
    #Written to prevent numerical underflow
    #uncensored
        t1<-log(l1)+a1*log(yv[zv==0])
        term<-yv[zv==0]
        term[t1 < 300]<-log(1+exp(t1[t1<300]))
        term[t1 > 300]<-t1[t1>300]
        Bv<-sum(log(a1*g1*l1)+(a1-1)*log(yv[zv==0])-(g1+1)*term)

    #censored
        t1<-log(l1)+a1*log(yv[zv==1])
        term<-yv[zv==1]
        term[t1 < 300]<-log(1+exp(t1[t1<300]))
        term[t1 > 300]<-t1[t1>300]
        Bv<-Bv+sum(-g1*term)
        return(Bv)
}
Burr.optim.func<-function(x,yv,zv){
        a1<-exp(x[1]);  g1<-exp(x[2]);  l1<-exp(x[3])
        return(-Burr.like(yv,zv,a1,g1,l1))
}
xhat.ml<-optim(c(0,0,0),method="BFGS",fn=Burr.optim.func,yv=y,zv=z,hessian=T)

#The ML estimates
Burr.ml.est<-theta<-xhat.ml$par
Burr.hess<-xhat.ml$hess
xvec<-seq(0,10,by=0.01)
yvec<-Burr.pdf(xvec,exp(theta[1]),exp(theta[2]),exp(theta[3]))
par(mar=c(4,4,2,0))
plot(xvec,yvec,type="l",xlab='x',ylab='f(x)');rug(y)
title('Maximum likelihood fit',line=1)
```



For the Bayesian analysis, we need a prior distribution. First, we consider a reparameterization and define

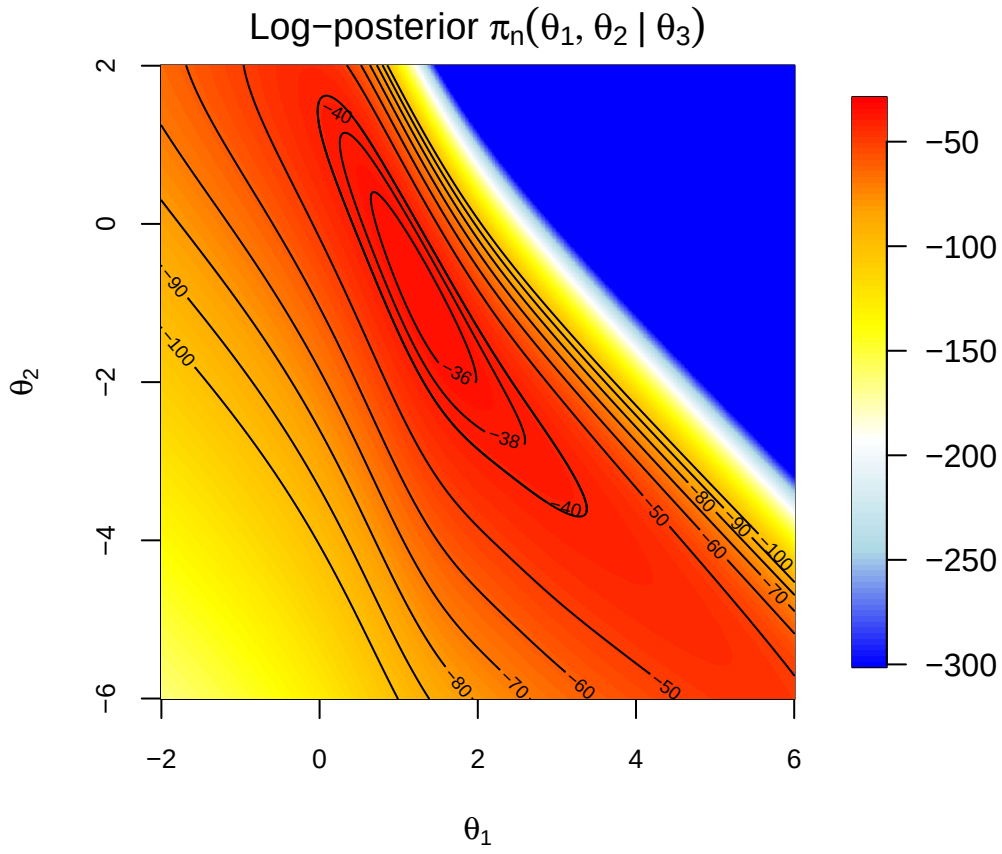
$$\theta_1 = \log \alpha \qquad \theta_2 = \log \gamma \qquad \theta_3 = \log \lambda$$

and then place independent $Normal(0, \sigma_\theta^2)$ priors on the new parameters. We may then compute the posterior mode using optimization methods.

```
#Bayesian Mode estimates
Burr.optim.post.func<-function(x,yv,zv,psd){
        a1<-exp(x[1])
        g1<-exp(x[2])
        l1<-exp(x[3])
        p1<-sum(dnorm(x,0,psd,log=T))
        return(-Burr.like(yv,zv,a1,g1,l1)-p1)
}
#lognormal prior for parameters
prior.sd<-2

xhat.Bayes<-optim(c(0,0,0),method="BFGS",fn=Burr.optim.post.func,yv=y,zv=z,psd=prior.sd,hessian=T)
th.post<-xhat.Bayes$par
```

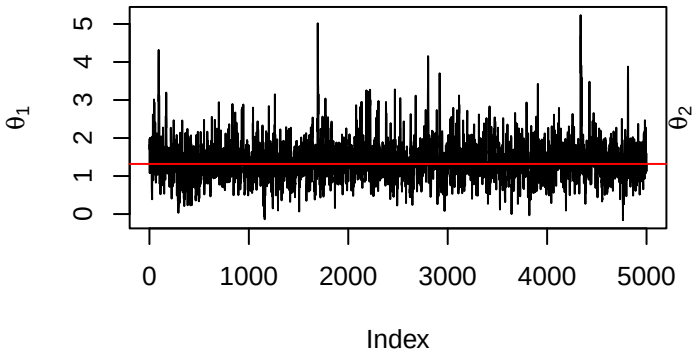
```
#Plot the posterior at the posterior mode for lambda
th1<-seq(-2,6,by=0.01)
th2<-seq(-6,2,by=0.01)
like.mat<-outer(th1,th2,"+")*0
post.mat<-outer(th1,th2,"+")*0
for(i in 1:length(th1)){
        for(j in 1:length(th2)){
        th<-c(th1[i],th2[j],th.post[3])
                like.mat[i,j]<-Burr.like(y,z,exp(th1[i]),exp(th2[j]),exp(th.post[3]))
                post.mat[i,j]<-max(like.mat[i,j]+sum(dnorm(th,0,prior.sd,log=T)),-300)

        }
}
plevs<-c(seq(-100, -40, by = 10),seq(-40,-30,by=2))
```

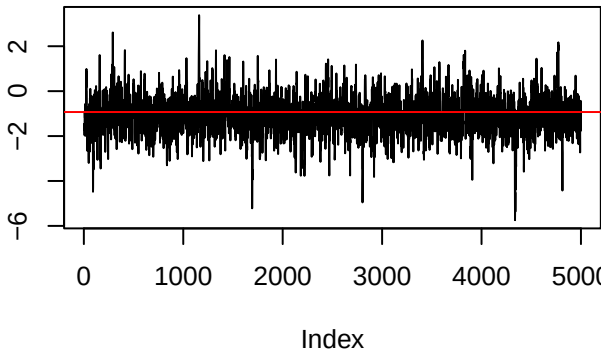
```
library(fields,quietly=TRUE)
par(pty='s',mar=c(4,4,2,2))
colfunc <- colorRampPalette(c("blue","lightblue","white","yellow","orange","red"))
image.plot(th1,th2,post.mat,col=colfunc(100),zlim=range(-300,-30),
           xlab=expression(theta[1]),ylab=expression(theta[2]),cex.axis=0.8)
contour(th1,th2,post.mat,add=T,levels=plevs)
title(expression(paste('Log-posterior ',pi[n](theta[1],theta[2]~"|"~theta[3]))))
```

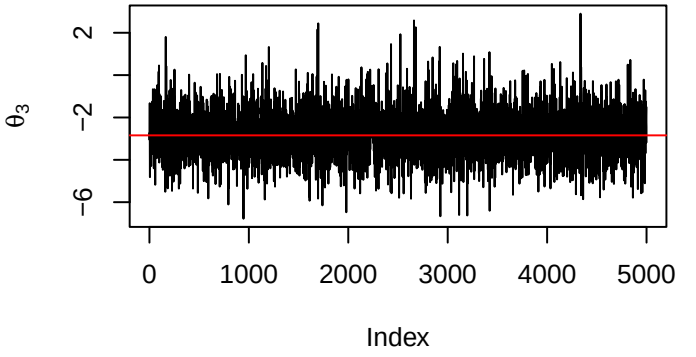


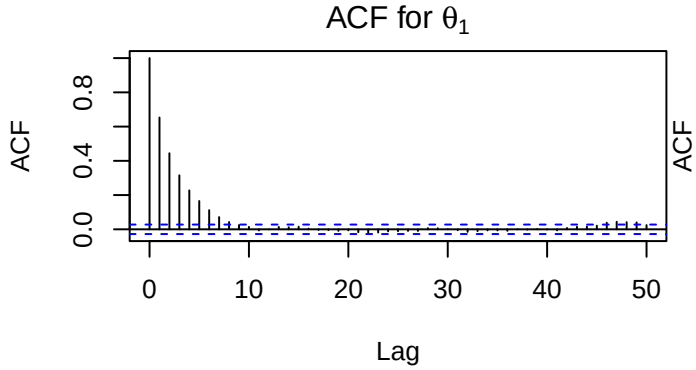

```
#Metropolis-Hastings MCMC
old.a1<-exp(th.post[1]);old.g1<-exp(th.post[2]);old.l1<-exp(th.post[3])
old.th1<-(th.post[1]);old.th2<-(th.post[2]);old.th3<-(th.post[3])

old.like<-Burr.like(y,z,old.a1,old.g1,old.l1)
old.prior<-sum(dnorm((c(old.th1,old.th2,old.th3)),0,prior.sd,log=T))

nburn<-1000
nsamp<-5000
nthin<-20
nits<-nburn+nsamp*nthin
sig.1<-1.0;sig.2<-1.0;sig.3<-1.0

t0<-proc.time()[3]
Burr.samp1<-matrix(0,nrow=nsamp,ncol=3)
ico<-i1<-i2<-i3<-0
for(iter in 1:nits){

        #Metropolis step for each parameter separately

        new.th1<-rnorm(1,old.th1,sig.1)
        new.a1<-exp(new.th1)
```

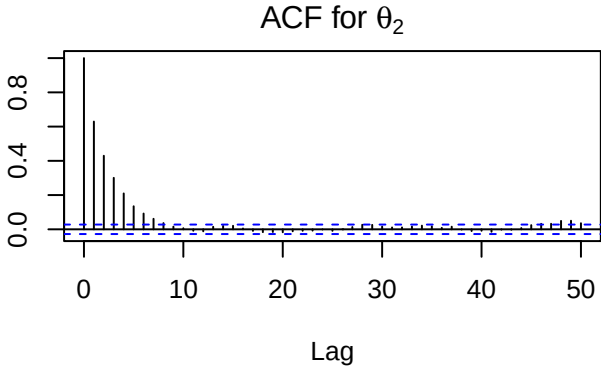
```
        new.like<-Burr.like(y,z,new.a1,old.g1,old.l1)
        new.prior<-sum(dnorm((c(new.th1,old.th2,old.th3)),0,prior.sd,log=T))
        if(log(runif(1)) < new.like+new.prior-old.like-old.prior){
                old.th1<-new.th1
                old.a1<-new.a1
                old.like<-new.like
                old.prior<-new.prior
                i1<-i1+1
        }
        new.th2<-rnorm(1,old.th2,sig.2)
        new.g1<-exp(new.th2)
        new.like<-Burr.like(y,z,old.a1,new.g1,old.l1)
        new.prior<-sum(dnorm((c(old.th1,new.th2,old.th3)),0,prior.sd,log=T))
        if(log(runif(1)) < new.like+new.prior-old.like-old.prior){
                old.th2<-new.th2
                old.g1<-new.g1
                old.like<-new.like
                old.prior<-new.prior
                i2<-i2+1
        }
        new.th3<-rnorm(1,old.th3,sig.3)
        new.l1<-exp(new.th3)
        new.like<-Burr.like(y,z,old.a1,old.g1,new.l1)
        new.prior<-sum(dnorm((c(old.th1,old.th2,new.th3)),0,prior.sd,log=T))
        if(log(runif(1)) < new.like+new.prior-old.like-old.prior){
                old.th3<-new.th3
                old.l1<-new.l1
                old.like<-new.like
                old.prior<-new.prior
                i3<-i3+1
        }
        if(iter > nburn & iter %% nthin == 0){
                ico<-ico+1
                Burr.samp1[ico,]<-c(old.a1,old.g1,old.l1)
        }
}
th.samp1<-log(Burr.samp1)
```

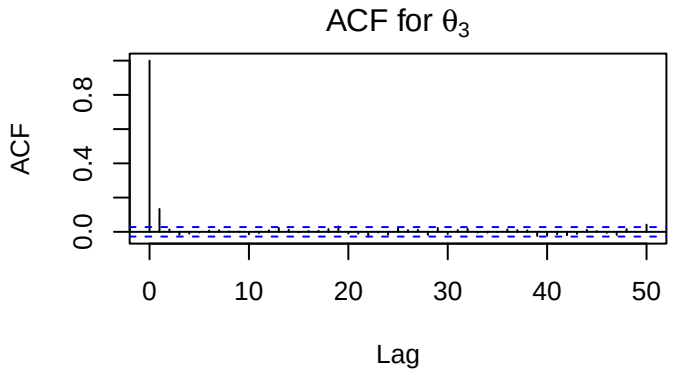




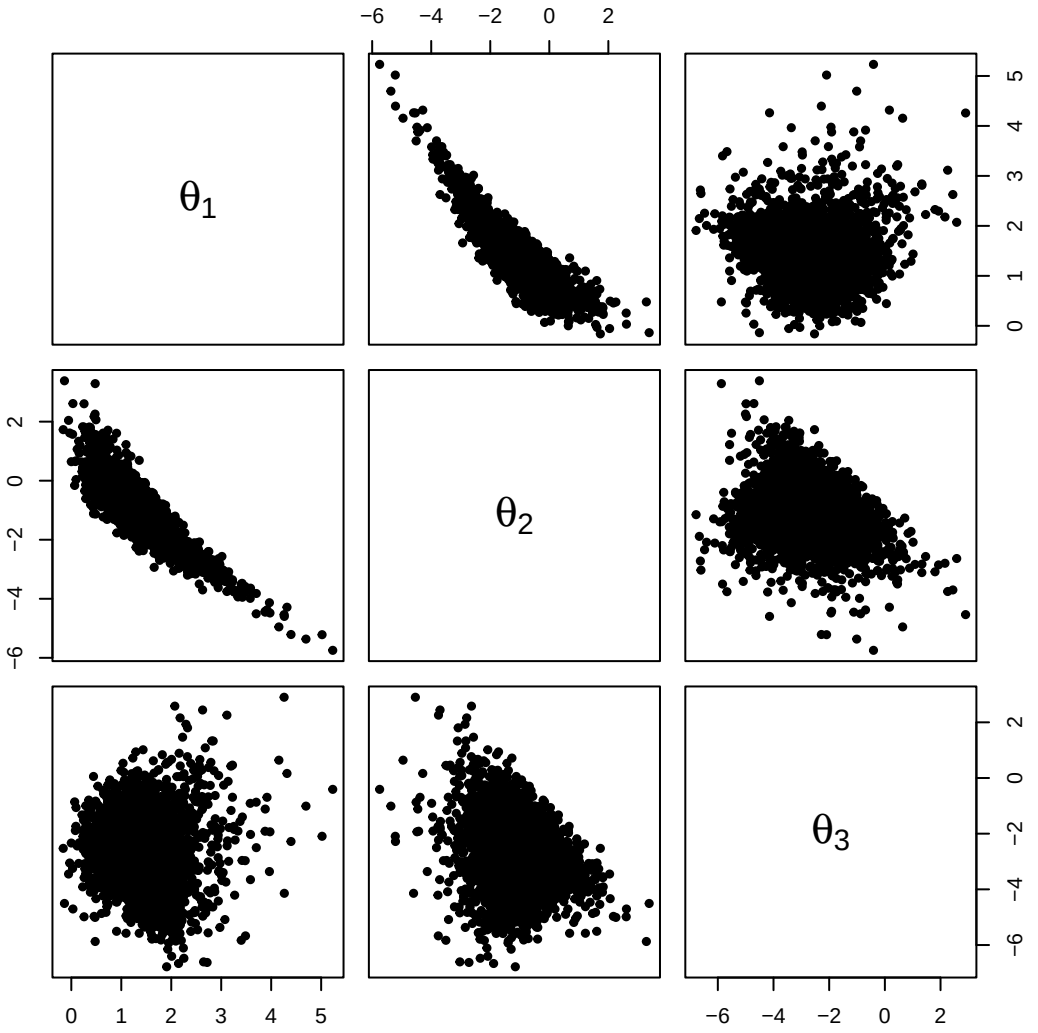






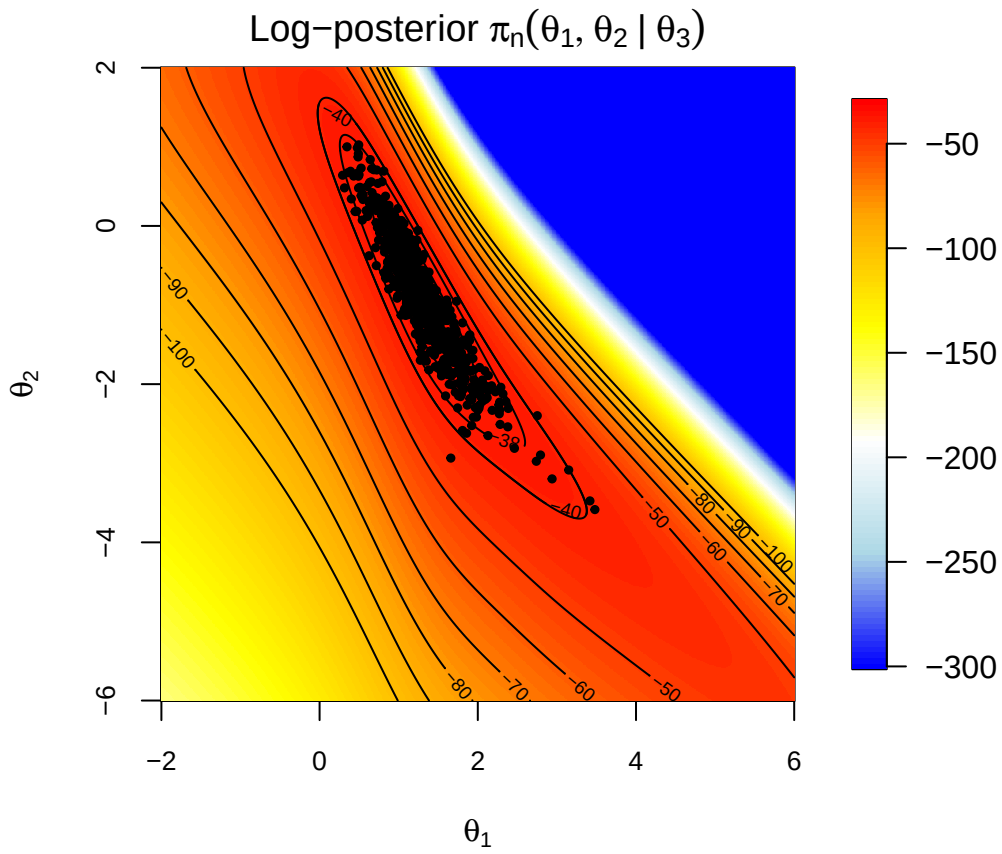



```
nvec<-c(expression(theta[1]),expression(theta[2]),expression(theta[3]))
par(pty='s')
pairs(th.samp1,labels=nvec,pch=19,cex=0.75)
```



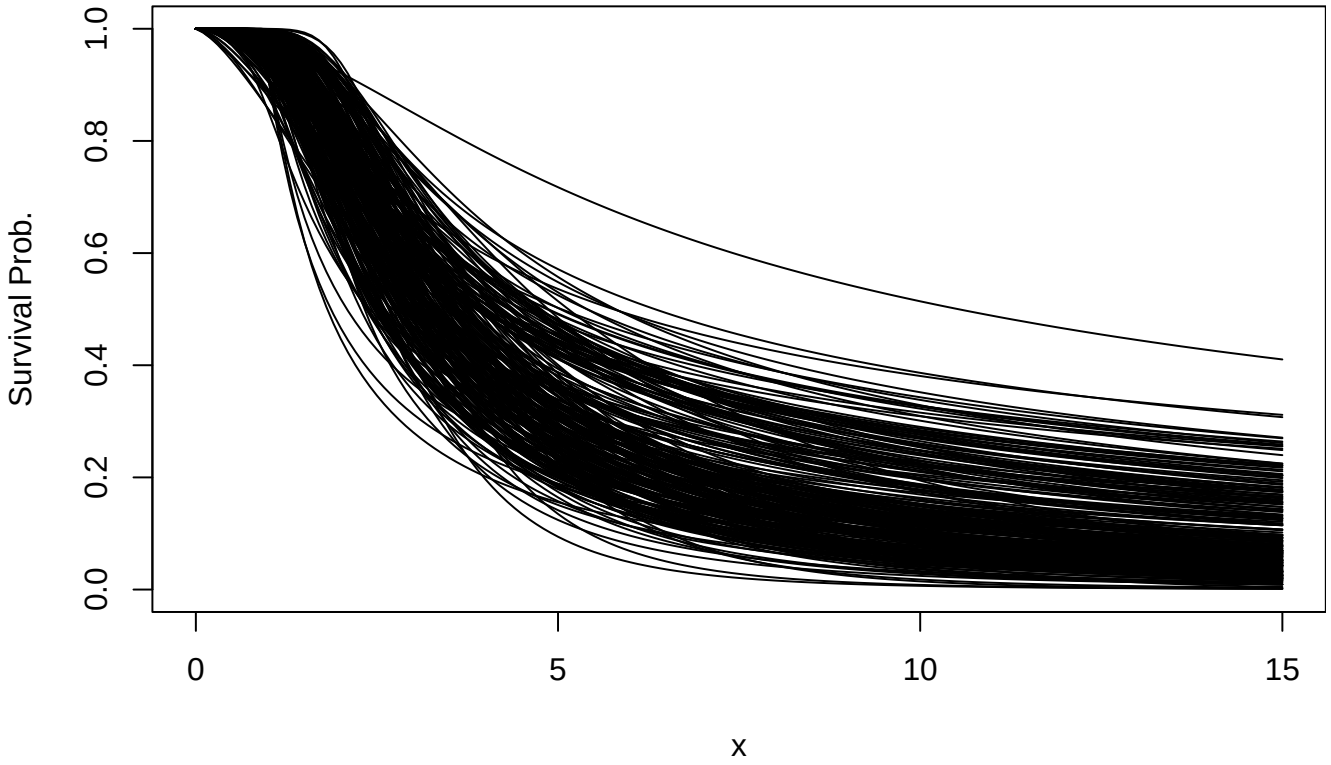

```
require(coda)
effectiveSize(th.samp1)

+      var1      var2      var3
+  938.3542 1017.2708 3821.3521
```

```
ivec<-abs(th.samp1[,3]-th.post[3])< 0.2
par(pty='s',mar=c(4,4,2,2))
image.plot(th1,th2,post.mat,col=colfunc(100),zlim=range(-300,-30),
           xlab=expression(theta[1]),ylab=expression(theta[2]),cex.axis=0.8)
contour(th1,th2,post.mat,add=T,levels=plevs)
title(expression(paste('Log-posterior ',pi[n](theta[1],theta[2]~"|"~theta[3]))))
points(th.samp1[ivec,1:2],pch=19,cex=0.5)
```

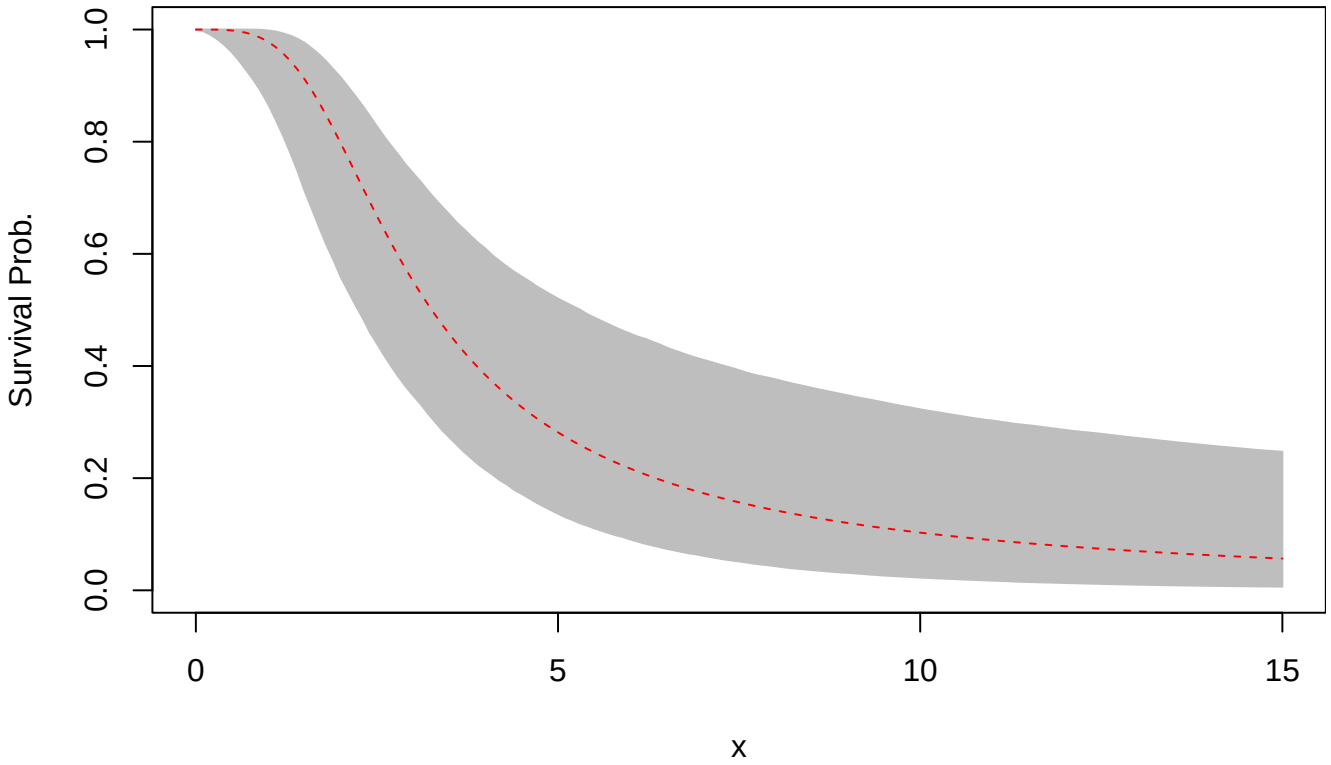


The posterior is being sampled fairly effectively using the standard Gibbs sampler in the log parameterization. From the posterior samples, we can compute sampled survivor functions.

```
Surv.x<-seq(0,15,by=0.1)
Surv.samp1<-matrix(0,nrow=nsamp,ncol=length(Surv.x))
a1<-exp(th.post[1])
g1<-exp(th.post[2])
l1<-exp(th.post[3])
Surv.y<-exp(-g1*log(1+l1*Surv.x^a1))
Surv.y.ml<-Surv.y
par(mar=c(4,4,1,0))
plot(Surv.x,Surv.y,type="l",ylim=range(0,1),ylab="Survival Prob.",xlab="x")
for(isamp in 1:nsamp){
        a1<-Burr.samp1[isamp,1]
        g1<-Burr.samp1[isamp,2]
        l1<-Burr.samp1[isamp,3]
        Surv.y<-exp(-g1*log(1+l1*Surv.x^a1))
        Surv.samp1[isamp,]<-Surv.y
        if(isamp %% 20 == 0){lines(Surv.x,Surv.y)}
}
```




```
Surv.q<-apply(Surv.samp1,2,quantile,prob=c(0.025,0.975))
par(mar=c(4,4,1,0))
plot(Surv.x,Surv.y,type="n",ylim=range(0,1),ylab="Survival Prob.",xlab="x")
polygon(c(Surv.x,rev(Surv.x)),c(Surv.q[1,],rev(Surv.q[2,])),col="gray",border="gray")
lines(Surv.x,Surv.y.ml,col="red",lty=2)
```



Reparameterization: We may reparameterize the model and rewrite

$$F(y) = 1 - \frac{1}{(1 + (y/\phi)^{\xi})^{\eta/\xi}}.$$

```
#Likelihood
Burr.like2<-function(yv,zv,xiv,etav,phiv){
    #Written to prevent numerical underflow
    a1<-xiv
    g1<-etav/xiv
    l1<-(1/phiv)^xiv
    #uncensored
        t1<-log(l1)+a1*log(yv[zv==0])
        term<-yv[zv==0]
        term[t1 < 300]<-log(1+exp(t1[t1<300]))
        term[t1 > 300]<-t1[t1>300]
        Bv<-sum(log(a1*g1*l1)+(a1-1)*log(yv[zv==0])-(g1+1)*term)
    #censored
        t1<-log(l1)+a1*log(yv[zv==1])
        term<-yv[zv==1]
        term[t1 < 300]<-log(1+exp(t1[t1<300]))
        term[t1 > 300]<-t1[t1>300]
        Bv<-Bv+sum(-g1*term)
    if(is.na(Bv)){Bv<--Inf}
        return(Bv)
}
```

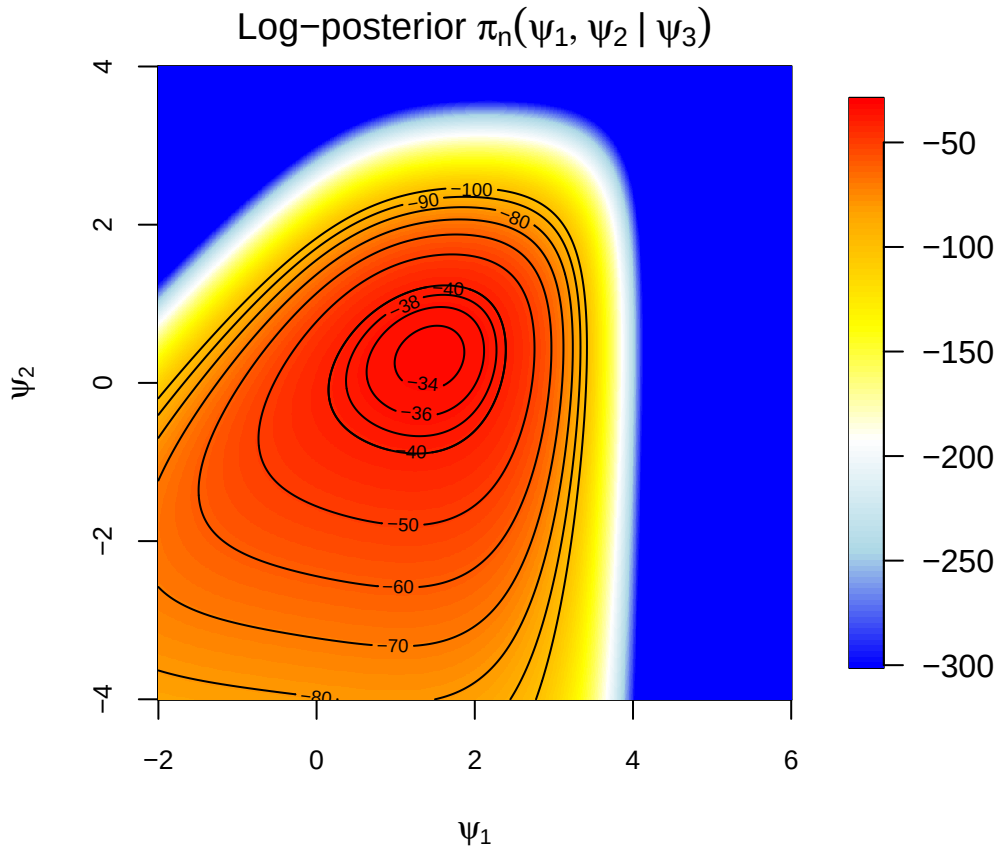
We then consider a further reparameterization and define

$$\psi_1 = \log \xi \equiv \log \alpha \qquad \psi_2 = \log \eta \equiv \log \gamma + \log \alpha \qquad \psi_3 = \log \phi = -\frac{1}{\alpha} \log \lambda$$

and with the independent $Normal(0, \sigma_\theta^2)$ priors on the θ parameters, we note the reparameterization $\theta_1 = \psi_1, \theta_2 = \psi_2 - \psi_1$, and $\theta_3 = -e^{-\psi_1}\psi_3$, yielding the Jacobian e^{ψ_1}, and the induces prior on the ψ parameters. We may again compute the posterior mode using optimization methods.

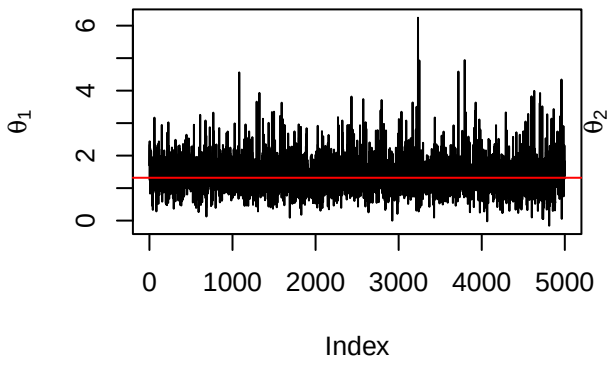
```
#Bayesian Mode estimates
Burr.optim.post.func2<-function(x,yv,zv,psd){
        xiv<-exp(x[1])
        etav<-exp(x[2])
        phiv<-exp(x[3])
    th<-c(x[1],x[2]-x[1],-x[3]*xiv)
        p1<-sum(dnorm(th,0,psd,log=T))+x[1]
        return(-Burr.like2(yv,zv,xiv,etav,phiv)-p1)
}
#lognormal prior for parameters
xhat.Bayes2<-optim(c(0,0,0),method="BFGS",fn=Burr.optim.post.func2,yv=y,zv=z,psd=prior.sd,hessian=T)
psi.post<-xhat.Bayes2$par
#Plot the posterior at the posterior mode for phi
psi1<-seq(-2,6,by=0.01)
psi2<-seq(-4,4,by=0.01)
like.mat2<-outer(psi1,psi2,"+")*0
post.mat2<-outer(psi1,psi2,"+")*0
for(i in 1:length(psi1)){
        for(j in 1:length(psi2)){
        psi<-c(psi1[i],psi2[j],psi.post[3])
        th<-c(psi[1],psi[2]-psi[1],-exp(psi[1])*psi[3])
        xiv<-exp(psi[1])
        etav<-exp(psi[2])
        phiv<-exp(psi[3])
                like.mat2[i,j]<-Burr.like2(y,z,xiv,etav,phiv)
```

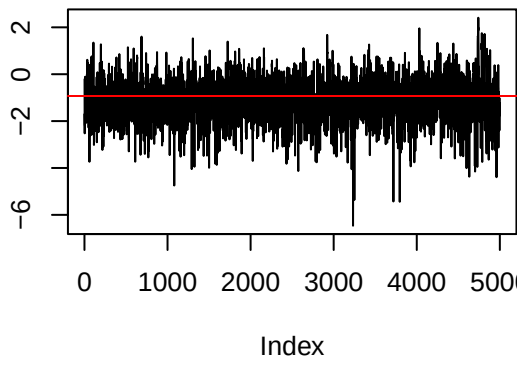
```
                post.mat2[i,j]<-max(like.mat2[i,j]+sum(dnorm(th,0,prior.sd,log=T))+psi[1],-300)

        }
}
plevs<-c(seq(-100, -40, by = 10),seq(-40,-30,by=2))
library(fields,quietly=TRUE)
par(pty='s',mar=c(4,4,2,2))
image.plot(psi1,psi2,post.mat2,col=colfunc(100),zlim=range(-300,-30),
           xlab=expression(psi[1]),ylab=expression(psi[2]),cex.axis=0.8)
contour(psi1,psi2,post.mat2,add=T,levels=plevs)
title(expression(paste('Log-posterior ',pi[n](psi[1],psi[2]~"|"~psi[3]))))
```

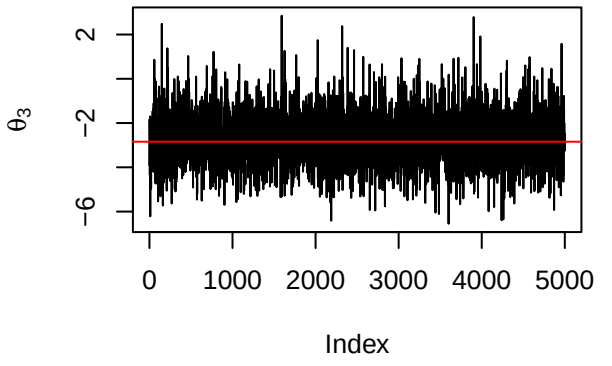


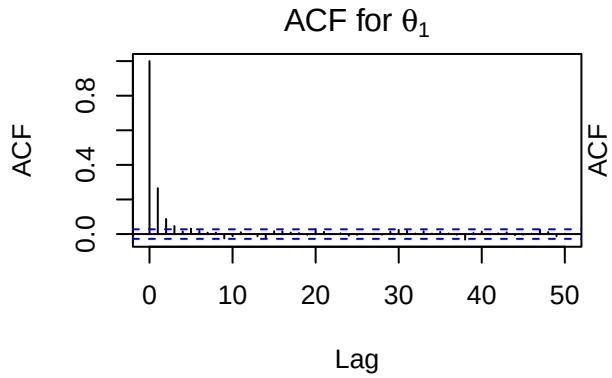

```
old.xi<-exp(psi.post[1]);old.eta<-exp(psi.post[2]);old.phi<-exp(psi.post[3])
old.th1<-psi.post[1];old.th2<-psi.post[2]-psi.post[1];old.th3<--exp(psi.post[1])*psi.post[3]
old.psi1<-psi.post[1];old.psi2<-psi.post[2];old.psi3<-psi.post[3]
old.like<-Burr.like2(y,z,old.xi,old.eta,old.phi)
old.prior<-sum(dnorm((c(old.th1,old.th2,old.th3)),0,prior.sd,log=T))+old.psi1

nburn<-1000
nsamp<-5000
nthin<-20
nits<-nburn+nsamp*nthin
sig.1<-1.0;sig.2<-1.0;sig.3<-1.0
t0<-proc.time()[3]
Burr.samp2<-Burr.samp2.psi<-matrix(0,nrow=nsamp,ncol=3)
ico<-i1<-i2<-i3<-0
for(iter in 1:nits){

        #Metropolis step for each parameter separately in psi parameterization
```

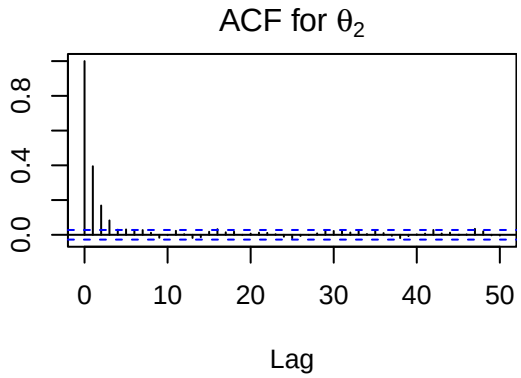
```
    new.psi1<-rnorm(1,old.psi1,sig.1)
    new.xi<-exp(new.psi1)
    new.th1<-new.psi1
    new.th2<-old.psi2-new.psi1
    new.th3<--exp(new.psi1)*old.psi3
    new.a1<-exp(new.th1)
    new.g1<-exp(new.th2)
    new.l1<-exp(new.th3)
        new.like<-Burr.like2(y,z,new.xi,old.eta,old.phi)
        new.prior<-sum(dnorm((c(new.th1,new.th2,new.th3)),0,prior.sd,log=T))+new.psi1
        if(log(runif(1)) < new.like+new.prior-old.like-old.prior){
                old.psi1<-new.psi1
        old.xi<-new.xi
        old.th1<-new.th1
        old.th2<-new.th2
        old.th3<-new.th3
                old.a1<-new.a1
        old.g1<-new.g1
        old.l1<-new.l1
                old.like<-new.like
                old.prior<-new.prior
                i1<-i1+1
        }
        new.psi2<-rnorm(1,old.psi2,sig.2)
    new.eta<-exp(new.psi2)
    new.th2<-new.psi2-old.psi1
        new.g1<-exp(new.th2)
        new.like<-Burr.like2(y,z,old.xi,new.eta,old.phi)
        new.prior<-sum(dnorm((c(old.th1,new.th2,old.th3)),0,prior.sd,log=T))+old.psi1
        if(log(runif(1)) < new.like+new.prior-old.like-old.prior){
        old.psi2<-new.psi2
        old.eta<-new.eta
                old.th2<-new.th2
                old.g1<-new.g1
                old.like<-new.like
                old.prior<-new.prior
                i2<-i2+1
        }
        new.psi3<-rnorm(1,old.psi3,sig.3)
    new.phi<-exp(new.psi3)
    new.th3<--exp(old.psi1)*new.psi3
        new.l1<-exp(new.th3)
        new.like<-Burr.like2(y,z,old.xi,old.eta,new.phi)
        new.prior<-sum(dnorm((c(old.th1,old.th2,new.th3)),0,prior.sd,log=T))+old.psi1
        if(log(runif(1)) < new.like+new.prior-old.like-old.prior){
        old.psi3<-new.psi3
        old.phi<-new.phi
        old.th3<-new.th3
                old.l1<-new.l1
                old.like<-new.like
                old.prior<-new.prior
                i3<-i3+1
        }
        if(iter > nburn & iter %% nthin == 0){
                ico<-ico+1
                Burr.samp2[ico,]<-c(old.a1,old.g1,old.l1)
        Burr.samp2.psi[ico,]<-c(old.psi1,old.psi2,old.psi3)
        }
}
th.samp2<-log(Burr.samp2)
psi.samp2<-Burr.samp2.psi
```

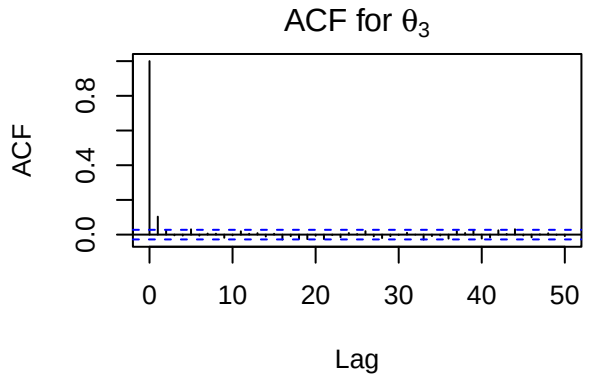

θ1
0
2
4
6
0
1000
2000
3000
4000
5000
Index

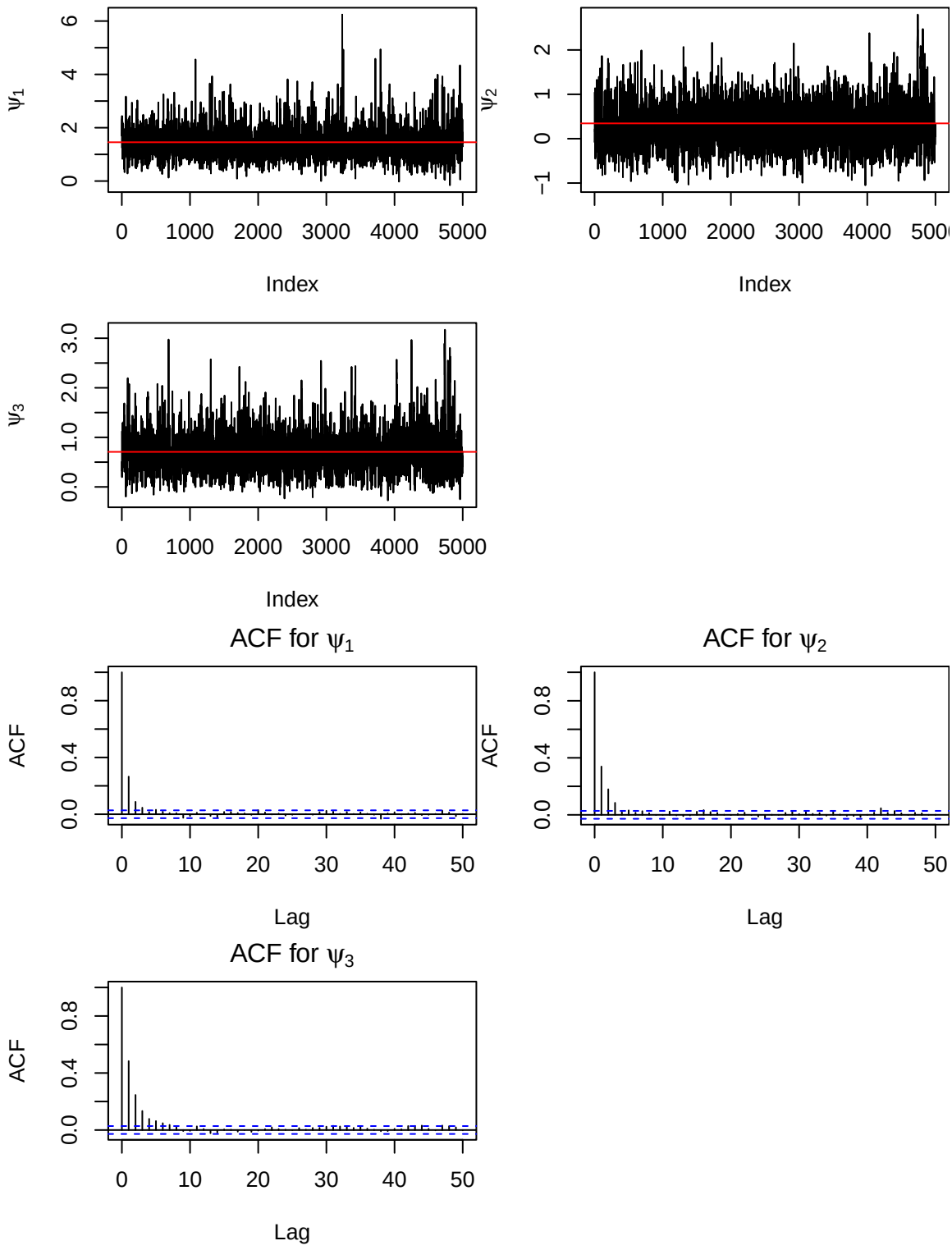

θ2
−6
−2
0
2
0
1000
2000
3000
4000
500
Index


θ3
−6
−2
2
0
1000
2000
3000
4000
5000
Index

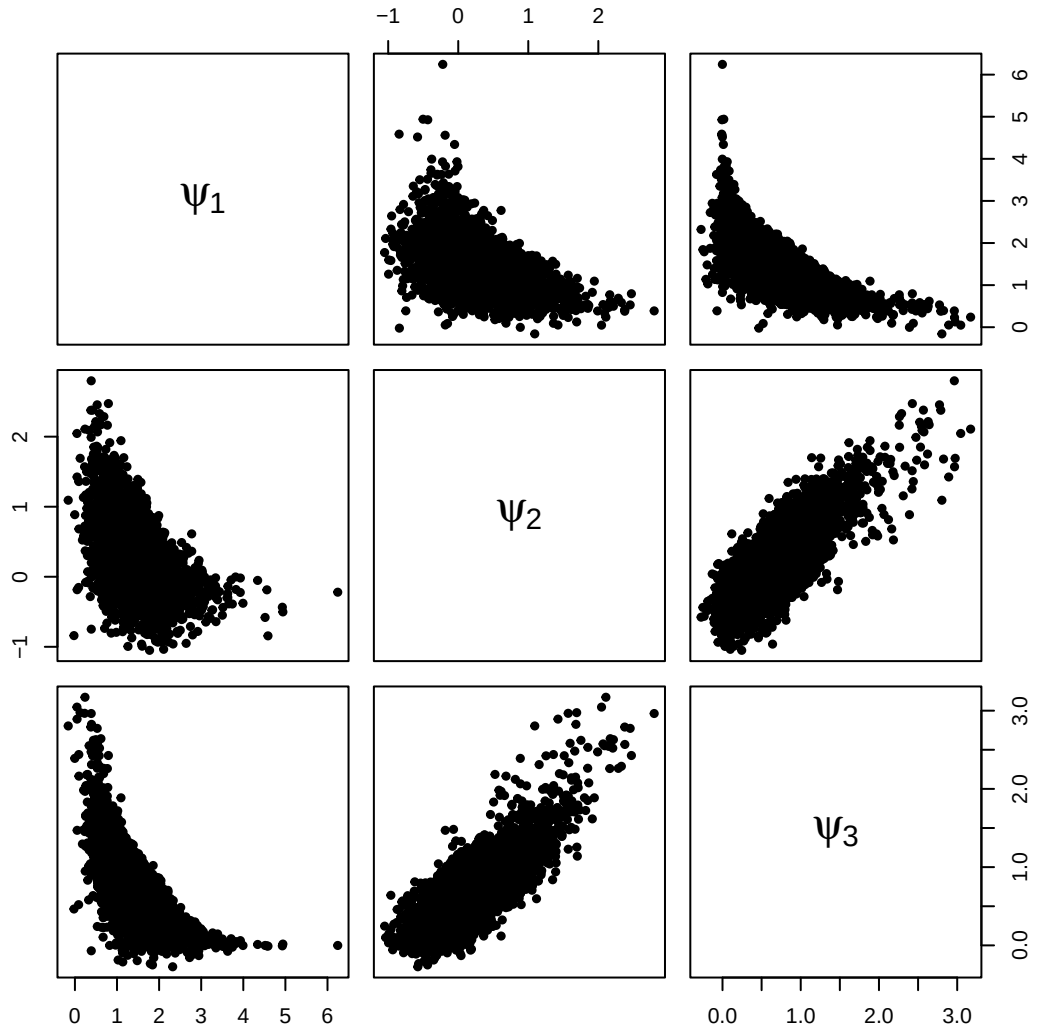

ACF for θ1
ACF
0.0
0.4
0.8
0
10
20
30
40
50
Lag


ACF for θ2
ACF
0.0
0.4
0.8
0
10
20
30
40
50
Lag

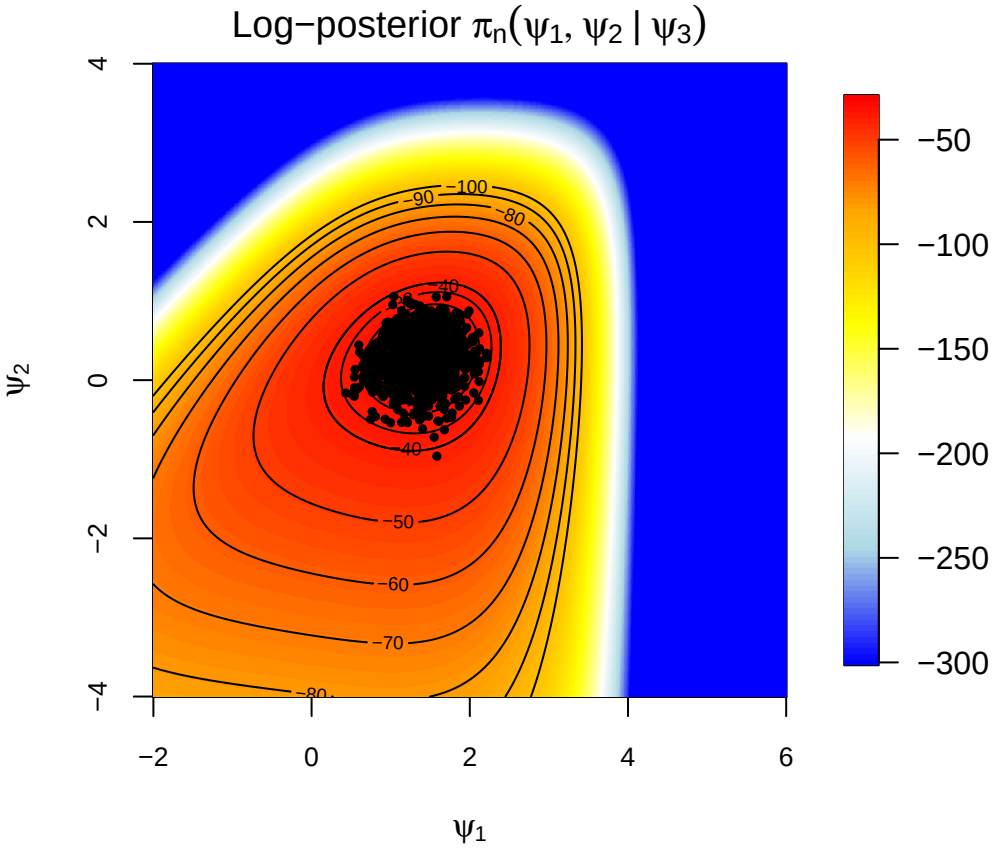

ACF for θ3
ACF
0.0
0.4
0.8
0
10
20
30
40
50
Lag




```
nvec<-c(expression(psi[1]),expression(psi[2]),expression(psi[3]))
par(pty='s')
pairs(psi.samp2,labels=nvec,pch=19,cex=0.75)
```




```
ivec<-abs(psi.samp2[,3]-psi.post[3])< 0.1
par(pty='s',mar=c(4,4,2,2))
image.plot(psi1,psi2,post.mat2,col=colfunc(100),zlim=range(-300,-30),
           xlab=expression(psi[1]),ylab=expression(psi[2]),cex.axis=0.8)
contour(psi1,psi2,post.mat2,add=T,levels=plevs)
title(expression(paste('Log-posterior ',pi[n](psi[1],psi[2]~"|"~psi[3]))))
points(psi.samp2[ivec,1:2],pch=19,cex=0.5)
```



This new parameterization appears to work better than the original one, so we retain it. The final step is to attempt the use of auxiliary variables to remove the need to use the censored density in the likelihood construction. We 'impute' the missing values for the two censored data points by sampling from the conditional density

$$f^\star(y;\psi) = \frac{f(y;\psi)}{P(Y>10)} = \frac{f(y;\psi)}{1-F(y;\psi)}.$$

Here we have

$$F^\star(y;\psi) = \frac{F(y;\psi)}{P(Y>c)} = \frac{F(y;\psi)}{1-F(c;\psi)} = \frac{1}{1-F(c;\psi)}\frac{1}{(1+(y/\phi)^\xi)^{\eta/\xi}} \qquad y>c.$$

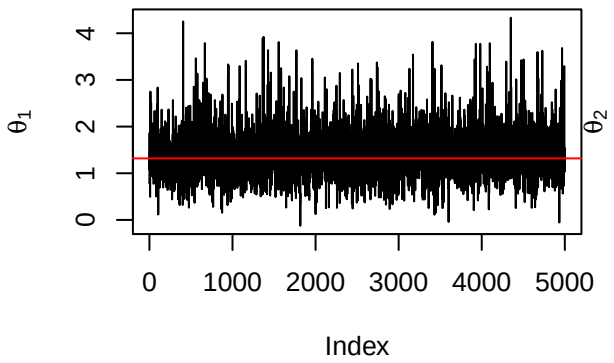
Thus to generate from $F^\star(y;\psi)$, we generate u from $Uniform(0,1)$, and set

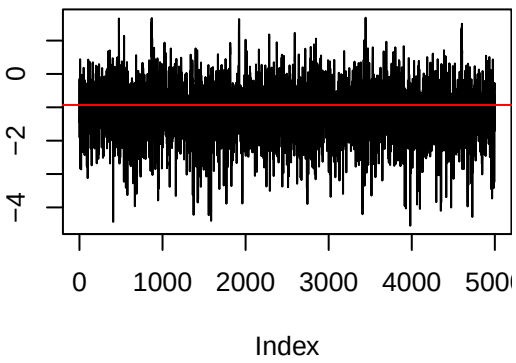
$$y^\star = \phi\left\{((1-F(c;\psi))u)^{-\xi/\eta}-1\right\}^{1/\xi}.$$

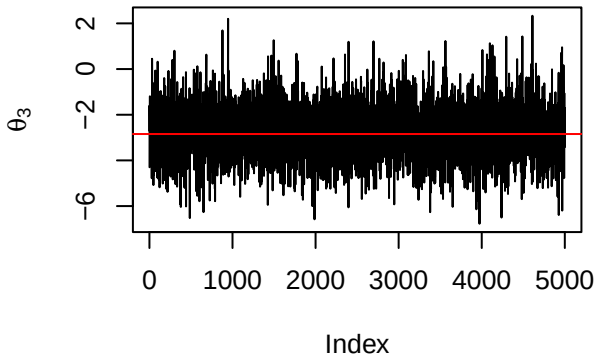
```
#Likelihood
Burr.like3<-function(yv,zv,xiv,etav,phiv){
    #Written to prevent numerical underflow
    a1<-xiv
    g1<-etav/xiv
    l1<-(1/phiv)^xiv
    #uncensored
        t1<-l1*(yv[zv==0])^a1
        term<-log(1+t1)
        Bv<-sum(log(a1*g1*l1)+(a1-1)*log(yv[zv==0])-(g1+1)*term)
    if(is.na(Bv)){Bv<--1e10}
        return(Bv)
}

Burr.surv2<-function(xvec,xiv,etav,phiv){
```

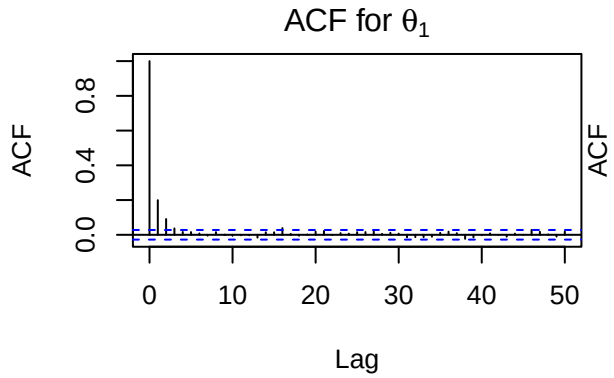
```
    a1<-xiv
    g1<-etav/xiv
    l1<-(1/phiv)^xiv
    t1<-l1*(xvec)^a1
        term<-log(1+t1)
        yvec<-exp(-g1*term)
        return(yvec)
}
#Fill in the missing data
yall<-y
zall<-z;zall[z==1]<-0
old.xi<-exp(psi.post[1]);old.eta<-exp(psi.post[2]);old.phi<-exp(psi.post[3])
old.th1<-psi.post[1];old.th2<-psi.post[2]-psi.post[1];old.th3<--exp(psi.post[1])*psi.post[3]
old.psi1<-psi.post[1];old.psi2<-psi.post[2];old.psi3<-psi.post[3]

old.like<-Burr.like3(yall,zall,old.xi,old.eta,old.phi)
old.prior<-sum(dnorm((c(old.th1,old.th2,old.th3)),0,prior.sd,log=T))+old.psi1

nburn<-1000
nsamp<-5000
nthin<-20
nits<-nburn+nsamp*nthin
sig.1<-1.0;sig.2<-1.0;sig.3<-1.0

t0<-proc.time()[3]
Burr.samp3<-Burr.samp3.psi<-matrix(0,nrow=nsamp,ncol=3)
y.samp3<-matrix(0,nrow=nsamp,ncol=n)
ico<-i1<-i2<-i3<-0
for(iter in 1:nits){
        #Metropolis step for each parameter separately in psi parameterization

        new.psi1<-rnorm(1,old.psi1,sig.1)
        new.xi<-exp(new.psi1)
        new.th1<-new.psi1
        new.th2<-old.psi2-new.psi1
        new.th3<--exp(new.psi1)*old.psi3
        new.a1<-exp(new.th1)
        new.g1<-exp(new.th2)
        new.l1<-exp(new.th3)
        new.like<-Burr.like3(yall,zall,new.xi,old.eta,old.phi)
        new.prior<-sum(dnorm((c(new.th1,new.th2,new.th3)),0,prior.sd,log=T))+new.psi1
        if(log(runif(1)) < new.like+new.prior-old.like-old.prior){
                old.psi1<-new.psi1
        old.xi<-new.xi
        old.th1<-new.th1
        old.th2<-new.th2
        old.th3<-new.th3
                old.a1<-new.a1
        old.g1<-new.g1
        old.l1<-new.l1
                old.like<-new.like
                old.prior<-new.prior
                i1<-i1+1
        }
        new.psi2<-rnorm(1,old.psi2,sig.2)
    new.eta<-exp(new.psi2)
    new.th2<-new.psi2-old.psi1
        new.g1<-exp(new.th2)
        new.like<-Burr.like3(yall,zall,old.xi,new.eta,old.phi)
        new.prior<-sum(dnorm((c(old.th1,new.th2,old.th3)),0,prior.sd,log=T))+old.psi1
        if(log(runif(1)) < new.like+new.prior-old.like-old.prior){
```

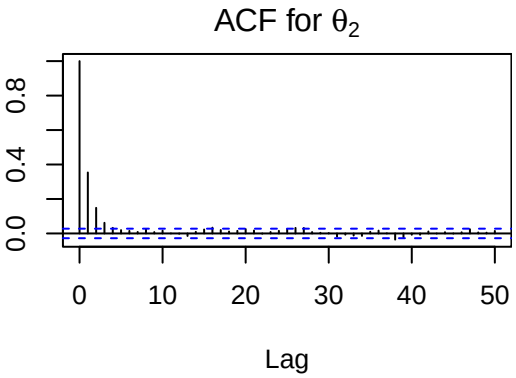
```
        old.psi2<-new.psi2
        old.eta<-new.eta
                old.th2<-new.th2
                old.g1<-new.g1
                old.like<-new.like
                old.prior<-new.prior
                i2<-i2+1
        }
        new.psi3<-rnorm(1,old.psi3,sig.3)
    new.phi<-exp(new.psi3)
    new.th3<--exp(old.psi1)*new.psi3
        new.l1<-exp(new.th3)
        new.like<-Burr.like3(yall,zall,old.xi,old.eta,new.phi)
        new.prior<-sum(dnorm((c(old.th1,old.th2,new.th3)),0,prior.sd,log=T))+old.psi1
        if(log(runif(1)) < new.like+new.prior-old.like-old.prior){
        old.psi3<-new.psi3
        old.phi<-new.phi
        old.th3<-new.th3
                old.l1<-new.l1
                old.like<-new.like
                old.prior<-new.prior
                i3<-i3+1
        }
    uvec<-runif(sum(z==1))
    svec<-Burr.surv2(y[z==1],old.xi,old.eta,old.phi)
    yall[z==1]<-old.phi*((svec*(1-uvec))^(-1/old.g1)-1)^(1/old.a1)

        old.like<-Burr.like3(yall,zall,old.xi,old.eta,old.phi)

        if(iter > nburn & iter %% nthin == 0){
                ico<-ico+1
                Burr.samp3[ico,]<-c(old.a1,old.g1,old.l1)
        Burr.samp3.psi[ico,]<-c(old.psi1,old.psi2,old.psi3)
        y.samp3[ico,]<-yall
        }
}
th.samp3<-log(Burr.samp3)
psi.samp3<-Burr.samp3.psi
```

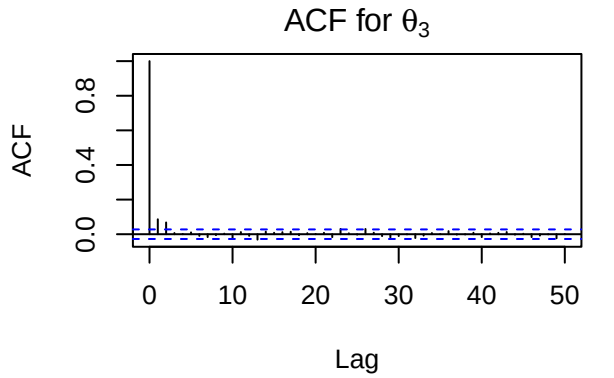

θ1
0 1 2 3 4
0 1000 2000 3000 4000 5000
Index

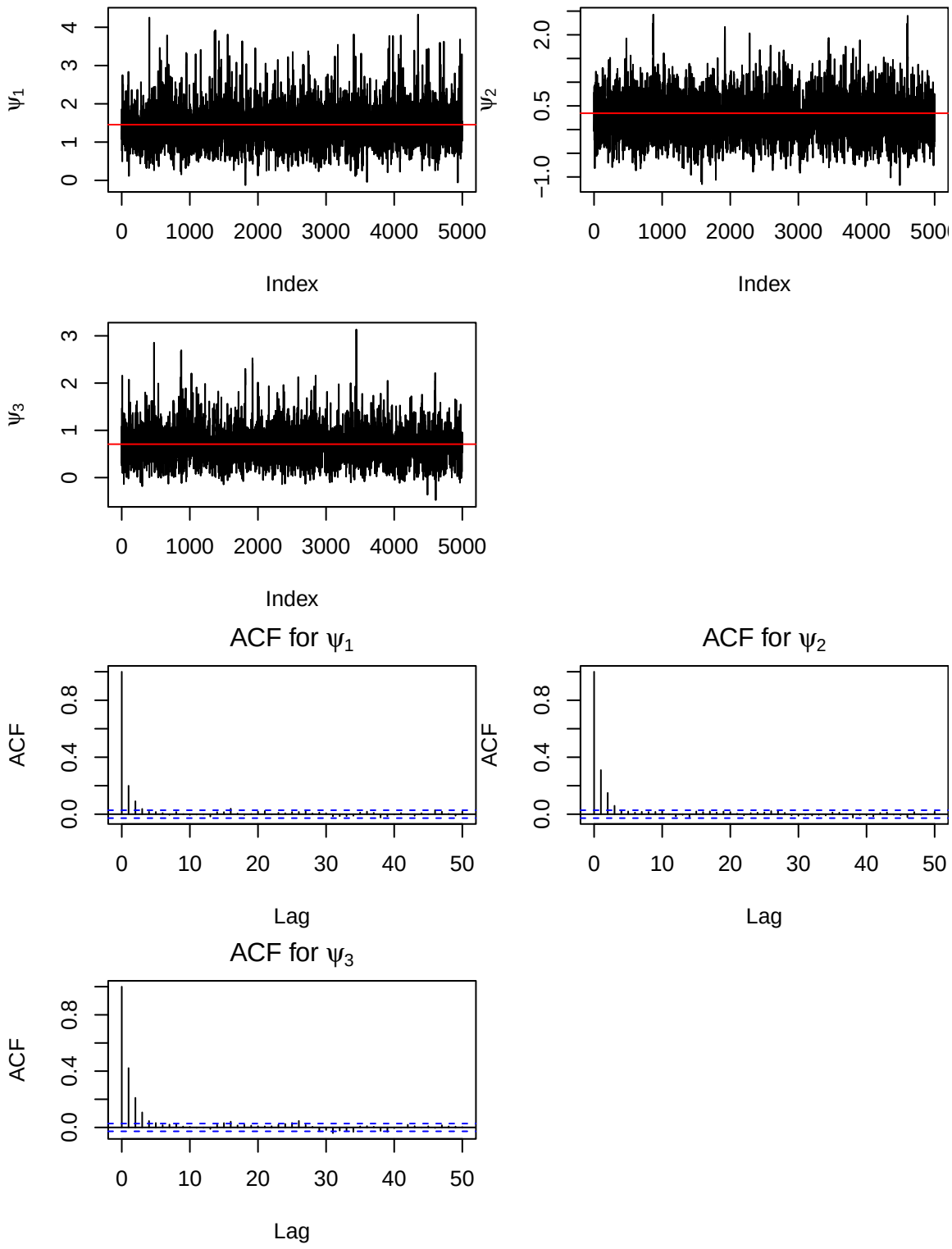

θ2
−4 −2 0
0 1000 2000 3000 4000 500
Index


θ3
−6 −2 0 2
0 1000 2000 3000 4000 5000
Index

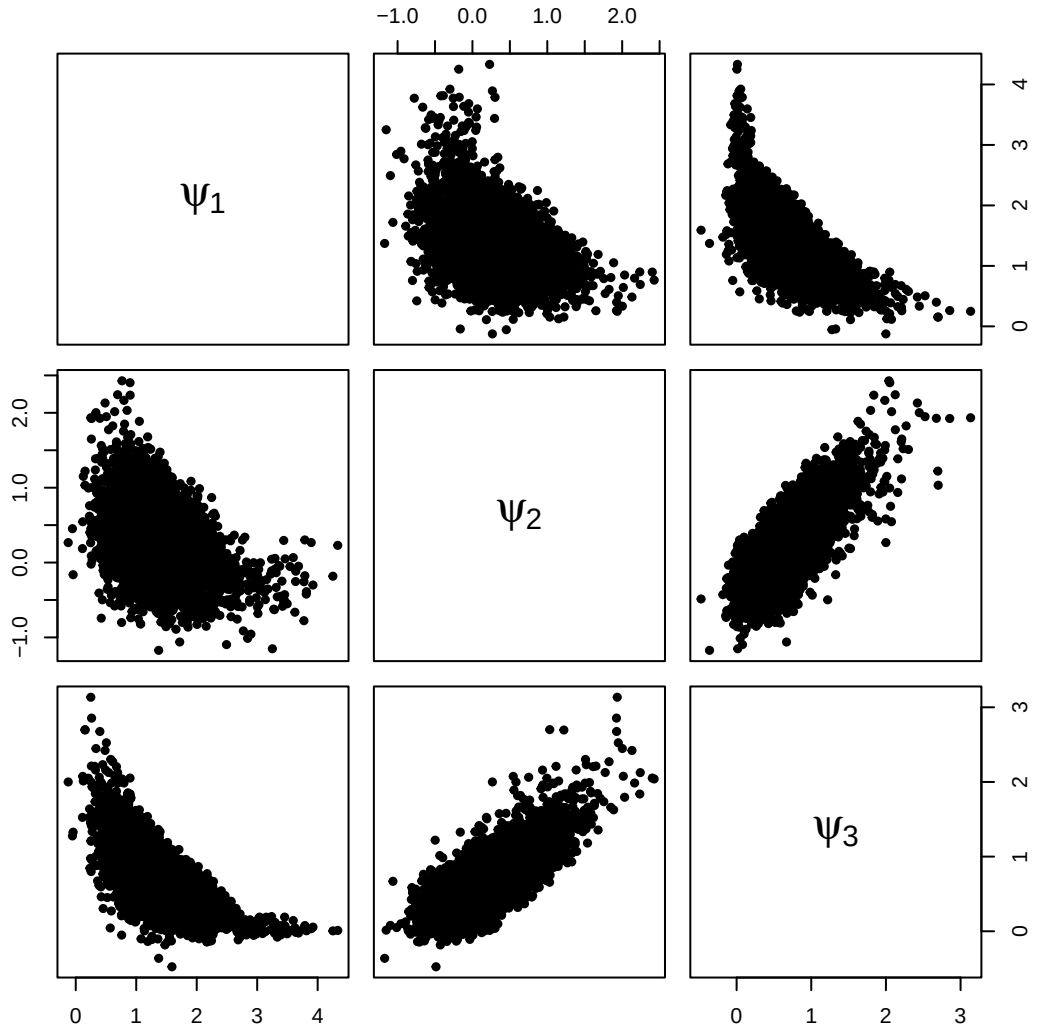

ACF for θ1
ACF
0.0 0.4 0.8
0 10 20 30 40 50
Lag


ACF for θ2
ACF
0.0 0.4 0.8
0 10 20 30 40 50
Lag


ACF for θ3
ACF
0.0 0.4 0.8
0 10 20 30 40 50
Lag

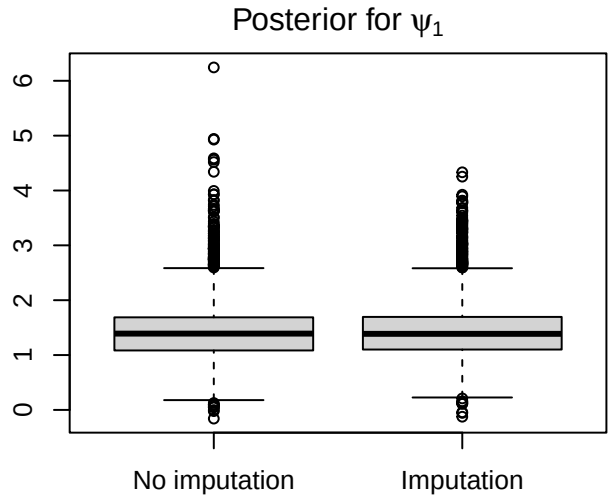


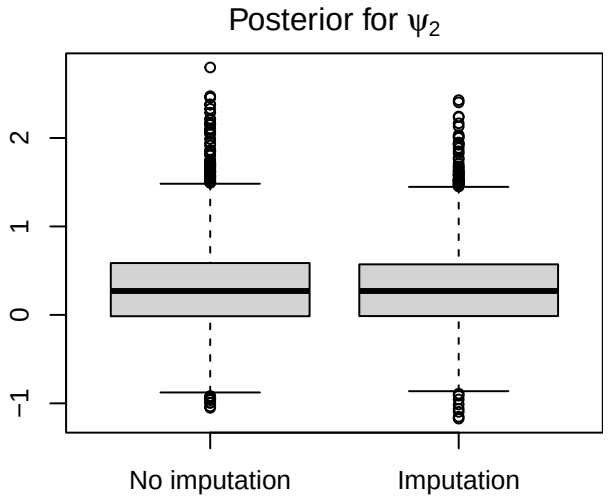

```
nvec<-c(expression(psi[1]),expression(psi[2]),expression(psi[3]))
par(pty='s')
pairs(psi.samp3,labels=nvec,pch=19,cex=0.75)
```

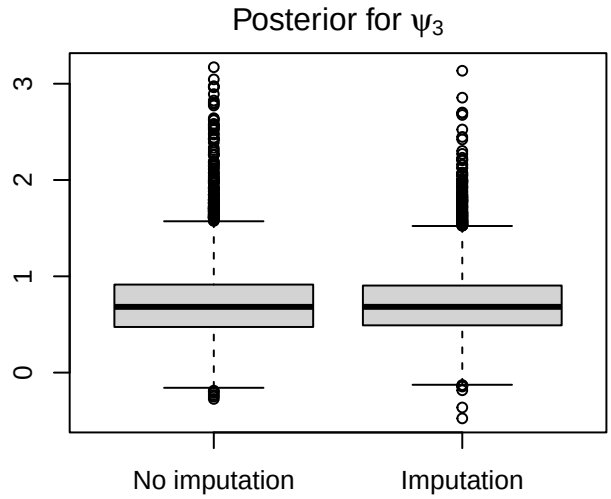



```
effectiveSize(psi.samp2)  #No imputation

+     var1     var2     var3
+ 2904.320 2096.787 1740.076

effectiveSize(psi.samp3)  #Imputation

+     var1     var2     var3
+ 2999.743 2339.939 1878.158
```

```
par(pty='m',mar=c(4,4,2,0),mfrow=c(2,2))
boxplot(cbind(psi.samp2[,1],psi.samp3[,1]),names=c('No imputation','Imputation'))
title(expression(paste('Posterior for ',psi[1])),line=1)
boxplot(cbind(psi.samp2[,2],psi.samp3[,2]),names=c('No imputation','Imputation'))
title(expression(paste('Posterior for ',psi[2])),line=1)
boxplot(cbind(psi.samp2[,3],psi.samp3[,3]),names=c('No imputation','Imputation'))
title(expression(paste('Posterior for ',psi[3])),line=1)
```







Missing data in Regression: For missing covariate data in a regression setting, sampling the missing covariates involves an extra component. Consider the full conditional distribution for missing covariates $\mathbf{X}^\star$

$$\pi_{\mathbf{X}^\star|\mathbf{X},\mathbf{Y},\mathbf{Y}^\star,\theta}(\mathbf{x}^\star|\mathbf{x},\mathbf{y},\mathbf{y}^\star;\theta)$$

By Bayes theorem, this is proportional to

$$f_{\mathbf{Y},\mathbf{Y}^\star|\mathbf{X},\mathbf{X}^\star}(\mathbf{y},\mathbf{y}^\star|\mathbf{x},\mathbf{x}^\star;\theta)\; f_{\mathbf{X}^\star|\mathbf{X}}(\mathbf{x}^\star|\mathbf{x};\theta)$$

The first term is the complete data likelihood for a regression problem. The second term most typically simplifies to

$$f_{\mathbf{X}^\star}(\mathbf{x}^\star;\theta)$$

under the assumption of random sampling of the response/covariate pairs.