Stats4Astro 2017

Model Building Lab

This lab will focus on writing MCMC samplers that account for selection effects. This problem is a simplified version of a cosmological techniques that uses Type Ia super novae (SNIa) as standard(izable) candles in the estimation of cosmological parameters that describe the expansion history of the universe. Again we use simulated data sets rather than real data to avoid technical difficulties that arise in fully accounting for the actual data generation mechanisms. Simulations and model fitting are based on the Λ-CDM model (using $\Omega_m = 0.3$, $\Omega_\kappa = 0$, $H_0 = 67.3$ km/s/Mpc) which is provided in tabulated form.

file name	columns	description
`Lambda-CDM.txt`	z = redshift, $\mu(z)$ = distance modulus	Λ-CDM model as a function of z

Suppose the distribution of the absolute magnitude of SNIa follow a normal distribuiton,

$$M_i \sim \textsc{Norm}(\mu, \sigma^2),$$

where M_i represents absolute magnitude of SNIa i, μ is the mean absolute magnitude, and σ^2 is the variance of the absolute magnitudes of SNIa. We are interested in estimating μ and σ^2 via a Bayesian analysis, using independent prior distributions on μ and σ^2: $\mu \sim \textsc{Norm}(-19.3, 20^2)$ and $\sigma^2 \sim \beta^2/\chi^2_\nu$, with $\beta^2 = \nu = 0.02$.

Unfortunately, we do not observe the absolute magnitudes, but rather observe the apparent magnitudes

$$m_i = \mu(z_i) + M_i,$$

where z_i is the observed redshift of SNIa. Values of $\mu(z_i)$ are given in `Lambda-CDM.txt`. These values assume $\Omega_m = 0.3$, $\Omega_\kappa = 0$, and $H_0 = 67.3$ km/s/Mpc; we take these values as given throughout this exercise. Suppose, owing to instrumental constraints we only observe SNIa with $m_i < 24$.

For given true values of μ and σ^2 we can simulate a set of redshifts as well as absolute and apparent magnitudes for a hypothetical set of SNIa. Let μ_{true} and σ^2_{true} represent these true values. We set $\mu_{\text{true}} = -19.3$ throughout, but consider how the value of σ^2_{true} effect our results by varying its value.

Consider a set of n SNIa and let N be the subset of these with $m_i < 24$. Our observed dataset will be of size N. For given values of n and σ^2_{true} the R-code in Table 1 can be used to simulate a dataset. (This code assumes that the redshifts of SNIa are distributed with density $\propto (1 + z_i)^2$.) The output from the R-code is summarized in Table 2.

1. Simulate a data set with $n = 200$ and $\sigma_{\text{true}} = 3$. Make a plot of your data with redshift on the horizontal axis and absolute magnitude on the vertical axis. Use color coding to indicate which SNIa are observed and plot a line to indicate the observation cut threshold (above

Table 1: R-code for simulating a dataset.

```
# Paramters
n           <- 200  # sample size before selection effects
var.true     <-  9   # intrinsic standard deviation of absolute magnitues.

# sample the redshifts
z.pts   <- 1:100/100
z.prob  <- (1+z.pts)^2
z.sim   <- sample(z.pts, size=n, replace=TRUE, prob=z.prob)

# read in Lambda CDM model
LCDM    <- read.table("LambdaCDM.txt", header=TRUE)

# Simulate absolute magnitued
M.sim   <- rnorm(n, -19.3, sqrt(var.true))
# compute the apparent magnitudes, as abolsute magntiude + mu (i.e., distance modulus)
m.sim   <- M.sim + LCDM[round(z.sim*100),2]

# Select if m.sim < 24
z.sel   <-  z.sim[m.sim<24]
M.sel   <-  M.sim[m.sim<24]
m.sel   <-  m.sim[m.sim<24]

# observed sample size
N       <- length(m.sel)
```

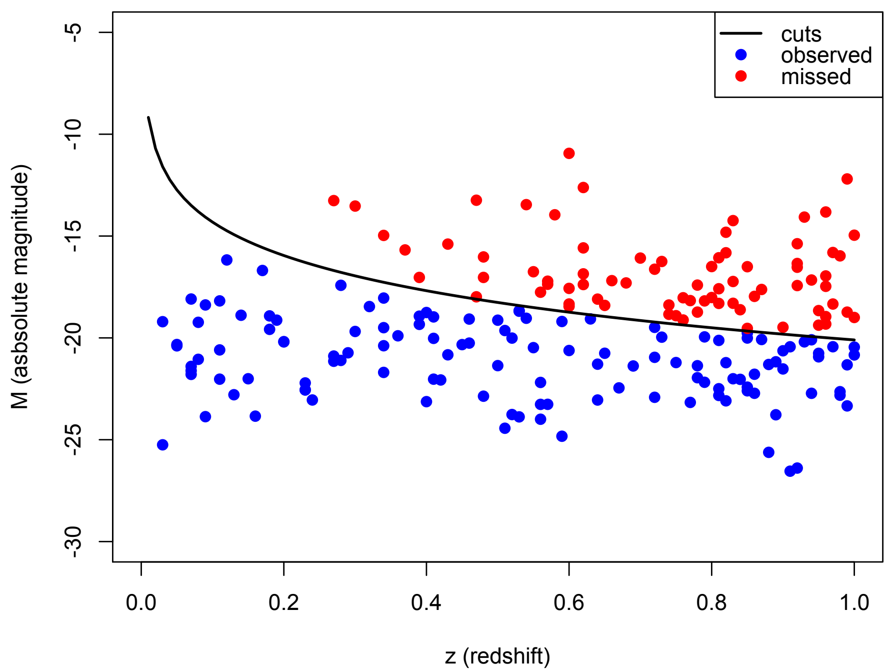
which SNIa are not observed).

Solution:

```
# code to simulate data given in Table 1

z.miss <- z.sim[m.sim>=24]
M.miss <- M.sim[m.sim>=24]
m.miss <- m.sim[m.sim>=24]

M.cutoff <- 24 - LCDM[round(z.pts*100),2]

plot(z.sel,M.sel,col="blue",pch=19,xlab="z (redshift)",ylab="M (asbsolute magnitude)",
            ylim=c(-30,-5),xlim=c(0,1))
points(z.miss,M.miss,col="red",pch=19)
lines(z.pts,M.cutoff,col="black",lwd=2)
legend("topright",c("cuts","observed","missed"),lty=c(1,NA,NA),lwd=c(2,NA,NA),
            pch=c(NA,19,19),col=c("black","blue","red"))
```

Table 2: Output from R-code for simulating a dataset.

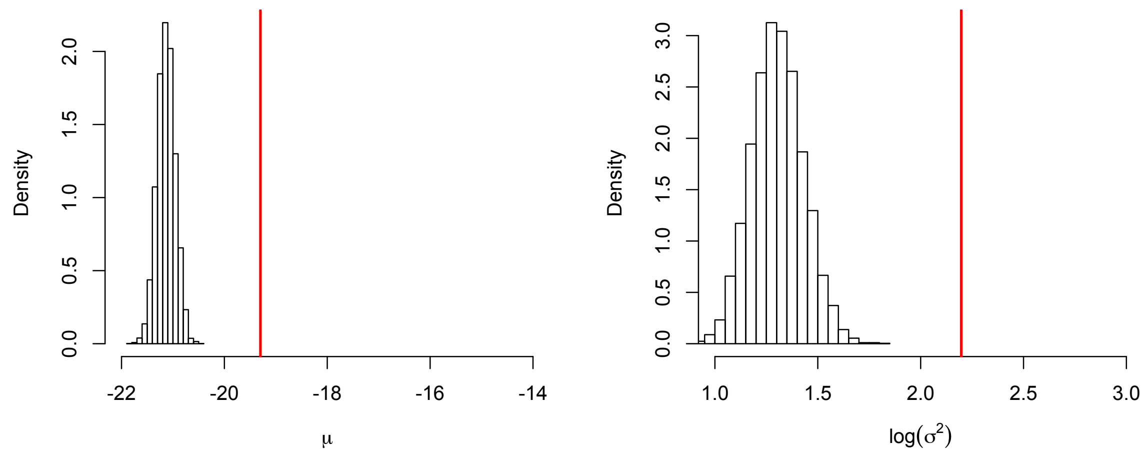
variable name	description
`z.sim`	Simulated redshifts *before* selection effect
`M.sim`	Absolute magnitudes *before* selection effect
`m.sim`	Apparent magnitudes *before* selection effect
`z.sel`	Simulated redshifts *after* selection effect
`M.sel`	Absolute magnitudes *after* selection effect
`m.sel`	Apparent magnitudes *after* selection effect



2. Write a Gibbs sampler to sample from the joint posterior distribution, $p(\mu, \sigma^2 \mid \texttt{M.sel})$ *ignoring selection effect.* How does the posterior distribution compare with the true parameter values?

   Solution:

```
mu <- NULL; sig2 <- NULL
mu[1] <- -19.3; sig2[1] <- 9

# set some priors
#mu0 <- 0; tau2 <- Inf; nu = 0; beta2 = 0

# set some priors
mu0 <- -19.3; tau2 <- 20^2; nu = 0.02; beta2 = 0.02

# Gibbs sampler
num.draws <- 11000
for(i in 2:num.draws){

    # Step 1: update mu from its conditional posterior dist'n given sig2
    a = (sum(M.sel)/sig2[i-1] + mu0/tau2)/(N/sig2[i-1] + 1/tau2)
    b = 1/(N/sig2[i-1] + 1/tau2)
    mu[i] <- rnorm(1,mean=a,sd=sqrt(b))

    # Step 2: update sig2 from its conditional posterior dist'n given mu
    sig2[i] <- sum(((M.sel-mu[i])^2)+beta2)/rchisq(1,df=(N+nu))
}

p2.mu <- mu
p2.sig2 <- sig2

# To compare to true parameter values (after examining trace plots)
# Discarding first 1000 draws as burn-in
par(mfrow=c(1,2))
hist(p2.mu[1001:12000],xlim=c(-22,-14), main="",xlab=expression(mu),prob=TRUE)
abline(v=-19.3,col="red",lwd=2)
hist(log(p2.sig2[1001:12000]),xlim=c(1,3), main="",xlab=expression(log(sigma^2)),prob=TRUE)
abline(v=log(9),col="red",lwd=2)
```



3. Suppose X follows an normal distribution with mean m and standard deviation s, but we only observe X if it is less than some threshold t. The observed variable follows a *truncated normal distribution.* Use the R-functions `dnorm` and `pnorm` to write a function that computes the density of this truncated normal distribution and the natural log of this density.

   Solution:

```
dtnorm <- function(x, mean, sd, truncation, log=FALSE){

value <- 0
if (x <= truncation){
value <-  dnorm(x,mean,sd)/pnorm(truncation,mean,sd)
}

if(log == TRUE) value <- log(value)

value
}
```

   In this code `dtnorm` stands for the density of a truncated `norm`al; `x` is the point at which the (log) density is evaluated, `mean` is the mean of the parent (untruncated) normal, `sd` is the standard deviation of the parent (untruncated) normal, `truncation` is the truncation point, and `log` should be set to `TRUE` to return the natural log of the density.

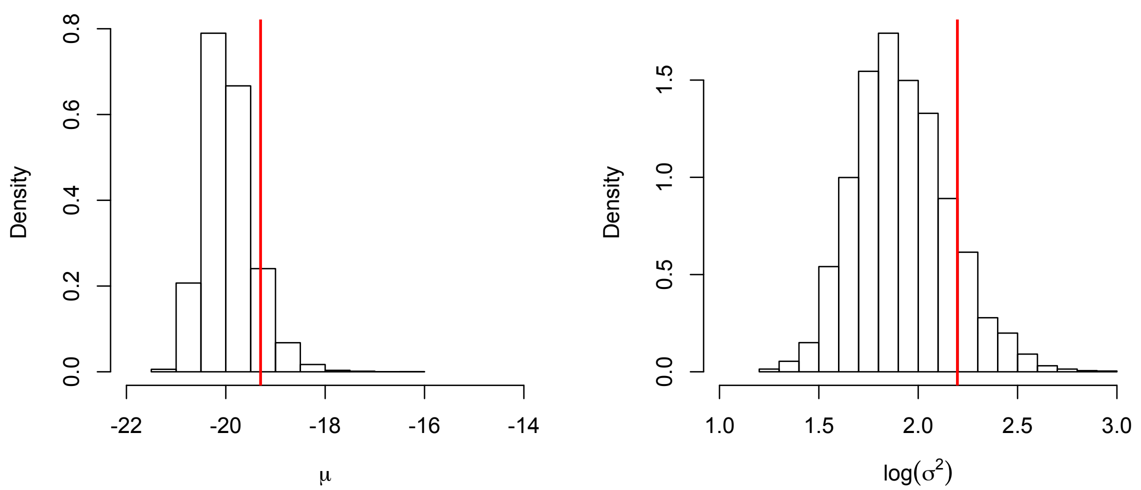
4. Write a Metropolis within Gibbs sampler to sample from the joint posterior distribution, $p(\mu, \sigma^2 \mid \texttt{M.sel})$ *accounting for selection effect.* How does the posterior distribution compare with the true parameter values? Compare these results with your answer to Question 2.

   Solution:

```
Mod2Gibbs <- function(Mags,z,start.vals,draw.num=10000,jump.par){
```

```
accept.mu = 0
accept.sig2 = 0

Draws = matrix(NA,draw.num,2)

#priors (can change)
mu0 <- -19.3
tau2 <- 20^2
beta2 <- 0.02
nu <- 0.02

logPost = function(mu,sig2){
    logLik <- 0
    for(i in 1:length(Mags)){
             logLik <- logLik + dtnorm(x=Mags[i],mean=mu,sd=sqrt(sig2),
                             truncation=(24-LCDM[round(z[i]*100),2]),log=TRUE)
    }
    logPrior <- dnorm(mu,mu0,sqrt(tau2),log=TRUE) + log(beta2)-
                             dchisq(sig2,nu,log=TRUE)
    return(logLik + logPrior)
}

    Draws[1,] <- start.vals

    for(i in 2:draw.num){

            #update mu
            mu.star <- rnorm(1,Draws[i-1,1],jump.par[1])
            log.ratio <- logPost(mu.star,Draws[i-1,2]) - logPost(Draws[i-1,1],Draws[i-1,2])
            ratio = exp( min(log.ratio,100) )
            temp = runif(1)
            if(temp < min(ratio,1)){
                    Draws[i,1] <- mu.star
                    accept.mu = accept.mu + 1
            }else{
                    Draws[i,1] <- Draws[i-1,1]
            }

            #update sig2
            sig2.star <- rlnorm(1,log(Draws[i-1,2]),jump.par[2])
            log.ratio <- logPost(Draws[i,1],sig2.star)-
                            dlnorm(sig2.star,log(Draws[i-1,2]),jump.par[2],log=TRUE)-
                            logPost(Draws[i,1],Draws[i-1,2])+
                            dlnorm(Draws[i-1,2],log(sig2.star),jump.par[2],log=TRUE);
             ratio = exp( min(log.ratio,100) )
             temp = runif(1)
             if(temp < min(ratio,1)){
                     Draws[i,2] <- sig2.star
                     accept.sig2 = accept.sig2 + 1
             }else{
                     Draws[i,2] <- Draws[i-1,2]
```

```
            }
        }
        output <- list(Draws,accept.mu/draw.num,accept.sig2/draw.num)
        names(output) <- c("Draws","accept.ratio.mu","accept.ratio.sig2")
        return(output)
}
}

# Running the Gibbs Sampler
p4 <- Mod2Gibbs(M.sel,z.sel,start.vals=c(-19.3,9),draw.num=11000,jump.par=c(0.82^2,0.62^2))

# Plotting results
par(mfrow=c(1,2))
hist(p4$Draws[1001:11000,1],xlim=c(-22,-14), main="",xlab=expression(mu),prob=TRUE)
abline(v=-19.3,col="red",lwd=2)
hist(log(p4$Draws[1001:11000,2]),xlim=c(1,3), main="",xlab=expression(log(sigma^2)),prob=TRUE)
abline(v=log(9),col="red",lwd=2)
```

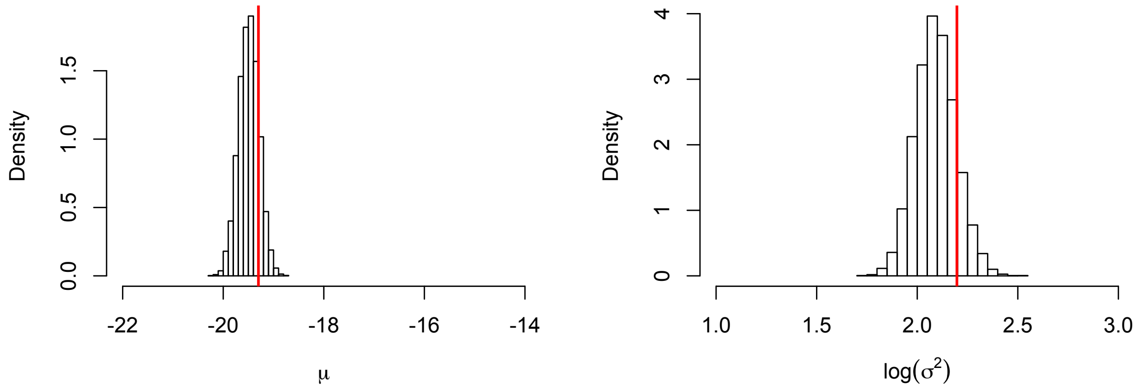


When ignoring selection effects in Question 2, the posterior distributions underestimate the true parameter values; the true parameter values lie outside a 95% credible interval (CI). When accounting for selection effects, the posterior variance grows and the posterior mean/medians shift closer to the true parameter values; the effect is that the true parameter values are captured by the 95% CIs.

5. Suppose there were no selection effects and data were available for the full $n = 200$ SNIa. Run your Gibbs sampler from Question 2 to sample the joint posterior distribution, $p(\mu, \sigma^2 \mid \texttt{M.sim})$ *ignoring selection effect.* (Note you are using the `M.sim` rather than `M.sel` in this analysis.) How does the posterior distribution compare with the true parameter values? Compare these results with your answer to Question 4.

Solution:

```
mu <- NULL; sig2 <- NULL
mu[1] <- -19.3; sig2[1] <- 9

# set some priors
#mu0 <- 0; tau2 <- Inf; nu = 0; beta2 = 0

# set some priors
mu0 <- -19.3; tau2 <- 20^2; nu = 0.02; beta2 = 0.02

# Gibbs sampler
num.draws <- 12000
for(i in 2:num.draws){

        # Step 1: update mu from its conditional posterior dist'n given sig2
        a = (sum(M.sim)/sig2[i-1] + mu0/tau2)/(n/sig2[i-1] + 1/tau2)
        b = 1/(n/sig2[i-1] + 1/tau2)
        mu[i] <- rnorm(1,mean=a,sd=sqrt(b))

        # Step 2: update sig2 from its conditional posterior dist'n given mu
        sig2[i] <- sum(((M.sim-mu[i])^2)+beta2)/rchisq(1,df=(n+nu))
}

p5.mu <- mu
p5.sig2 <- sig2

par(mfrow=c(2,1))
plot(p5.mu,type="l")
plot(p5.sig2,type="l")

par(mfrow=c(1,2))
hist(p5.mu[2001:12000],xlim=c(-22,-14),xlab=expression(mu),prob=TRUE)
abline(v=-19.3,col="red",lwd=2)
hist(log(p5.sig2[2001:12000]),xlim=c(1,3),xlab=expression(log(sigma^2)),prob=TRUE)
abline(v=log(9),col="red",lwd=2)
```



If we suppose that data were available for the full $n = 200$ SNIa and ignore selection effects, the posterior mean/medians are close to the true parameter values. Compared to Question 4, the posterior variances shrink considerably due to the extra information we gain by using the full $n = 200$ SNIa dataset.

6. You should have found the posterior variances of μ and σ^2 to be larger in Question 4 than in Question 5. Experimenting with the value of n, how large must n be for Var(μ | `M.sel`) to be about the same size as Var(μ | `M.sim`) computed with $n = 200$ in Question 5? What is the corresponding value of N?

Solution:

```
# Parameters
n.new <- 2000  # NEW sample size before selection effects
var.true <- 9 # intrinsic standard deviation of absolute magnitudes

# sample the redshifts
z.pts <- 1:100/100
z.prob <- (1+z.pts)^2
z.sim.new <- sample(z.pts,size=n.new,replace=TRUE,prob=z.prob)

# read in Lambda CDM model
LCDM <- read.table("LambdaCDM.txt", header=TRUE)

# Simulate absolute magnitude
M.sim.new <- rnorm(n.new, -19.3, sqrt(var.true))
# Compute the apparent magnitudes, as absolute magnitude + mu (i.e., distance modulus)
m.sim.new <- M.sim.new + LCDM[round(z.sim.new*100),2]

# Select if m.sim < 24
z.sel.new <- z.sim.new[m.sim.new<24]
M.sel.new <- M.sim.new[m.sim.new<24]
m.sel.new <- m.sim.new[m.sim.new<24]
```

```
# observed sample size
N.new <- length(m.sel.new)

p6 <- Mod2Gibbs(M.sel.new,z.sel.new,start.vals=c(-19.3,9),draw.num=11000,
            jump.par=c(0.82^2,0.62^2))

var(p6$Draws[,1])
```

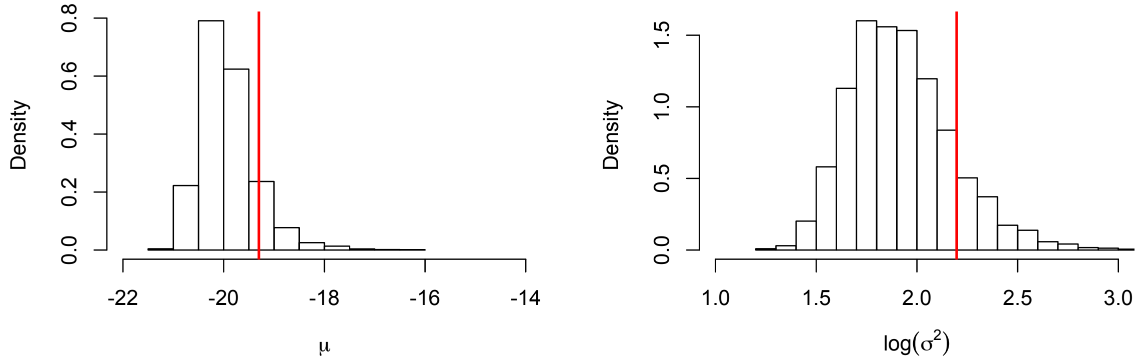
With $n = 2000$, $N = 1203$ and $\text{Var}(\mu \mid \texttt{M.sel})$ is about 0.05, which is similar to $\text{Var}(\mu \mid \texttt{M.sel})$.

7. In practice the apparent, not absolute magnitudes are observed. Write an MCMC sampler for $p(\mu, \sigma^2 \mid \texttt{m.sel})$.

   Solution:

```
AppMagGibbs <- function(Mags,z,start.vals,draw.num=10000,jump.par){

accept.mu = 0
accept.sig2 = 0

Draws = matrix(NA,draw.num,2)

#priors (can change)
mu0 <- -19.3
tau2 <- 20^2
beta2 <- 0.02
nu <- 0.02

logPost = function(mu,sig2){
    logLik <- 0
    for(i in 1:length(Mags)){
            logLik <- logLik + dtnorm(x=Mags[i],mean=mu+LCDM[round(z[i]*100),2],
                        sd=sqrt(sig2),truncation=24,log=TRUE)
     }
     #sum(dtnorm(x=Mags,mean=mu,sd=sqrt(sig2),truncation=LCDM[round(z*100),2]))
     logPrior <- dnorm(mu,mu0,sqrt(tau2),log=TRUE) + log(beta2) - dchisq(sig2,nu,log=TRUE)
     return(logLik + logPrior)
}

    Draws[1,] <- start.vals

    for(i in 2:draw.num){

            #update mu
            mu.star <- rnorm(1,Draws[i-1,1],jump.par[1])
            log.ratio <- logPost(mu.star,Draws[i-1,2]) - logPost(Draws[i-1,1],Draws[i-1,2])
            ratio = exp( min(log.ratio,100) )
            temp = runif(1)
```

```
            if(temp < min(ratio,1)){
            Draws[i,1] <- mu.star
            accept.mu = accept.mu + 1
            }else{
            Draws[i,1] <- Draws[i-1,1]
            }

            #update sig2
            sig2.star <- rlnorm(1,log(Draws[i-1,2]),jump.par[2])
            log.ratio <- logPost(Draws[i,1],sig2.star)-
                         dlnorm(sig2.star,log(Draws[i-1,2]),jump.par[2],log=TRUE)-
                          logPost(Draws[i,1],Draws[i-1,2])+
                         dlnorm(Draws[i-1,2],log(sig2.star),jump.par[2],log=TRUE);
             ratio = exp( min(log.ratio,100) )
             temp = runif(1)
             if(temp < min(ratio,1)){
                    Draws[i,2] <- sig2.star
                    accept.sig2 = accept.sig2 + 1
            }else{
                    Draws[i,2] <- Draws[i-1,2]
            }
            print(i)
            }
    output <- list(Draws,accept.mu/draw.num,accept.sig2/draw.num)
    names(output) <- c("Draws","accept.ratio.mu","accept.ratio.sig2")
    return(output)
}

p7 <- AppMagGibbs(m.sel,z.sel,start.vals=c(-19.3,9),draw.num=11000,jump.par=c(0.9^2,0.65^2))

par(mfrow=c(2,1))
plot(p7$Draws[,1],type="l")
plot(p7$Draws[,2],type="l")

par(mfrow=c(1,2))
hist(p7$Draws[1001:11000,1],xlim=c(-22,-14), main="",xlab=expression(mu),prob=TRUE)
abline(v=-19.3,col="red",lwd=2)
hist(log(p7$Draws[1001:11000,2]),xlim=c(1,3), main="",xlab=expression(log(sigma^2)),prob=TRUE)
abline(v=log(9),col="red",lwd=2)
```



8. *Bonus*: In practice, the observed apparent magnitudes include observation errors. Suppose $m_i^{\text{obs}} = m_i + e_i$, where e_i are independent mean-zero Gaussian observation errors with known variance, τ^2. Generalize the sampler you wrote for Question 7 to account for observation errors. Use a simulation study, varying the values of τ^2 and σ^2 to explore how the observation errors effect the final error bars for μ.

   Solution: Left as an exercise!