Measurement Error in Linear Regression

Galin Jones

Fall 2021

Measurement error

Consider the usual linear regression setting where we observe a response Y along with a p-vector of predictors (or covariates) X and we model the conditional mean as

$$E[Y|X = x] = x\beta$$

or

$$Y = X\beta + \epsilon \qquad \epsilon \sim \mathrm{N}(0, \sigma^2).$$

The distribution of X is essentially immaterial here.

So far the setting has assumed that the response and covariates are measured perfectly, but, of course, this does not happen in nearly any setting so we need to modify the standard model.

Now suppose that, instead of observing X we instead observe $X^* = X + U$ where U is an independent rv. Then our model becomes, with $X = X^* - U$

$$Y = (X^* - U)\beta + \epsilon = X^*\beta + (\epsilon - U\beta).$$

Since X^* and $\epsilon - U\beta$ are correlated this is *not* the usual regression setting.

Notice that errors in the measurement of the response Y do not need additional attention when X is measured perfectly. Suppose we want to observe Y, but instead observe $Y^* = Y + U$ where U is some independent rv. Then

$$Y^* = X\beta + (\epsilon + U) = X\beta + \epsilon^*$$

which is the usual regression setting where the variance of the error will be larger.

Most commonly we have measurement error in both the response Y and the covariates X so that we observe $Y^* = Y + U$ and $X^* = X + V$ where we assume U and V are independent and independent. Hence

$$Y^* - V = (X^* - U)\beta + \epsilon$$

or

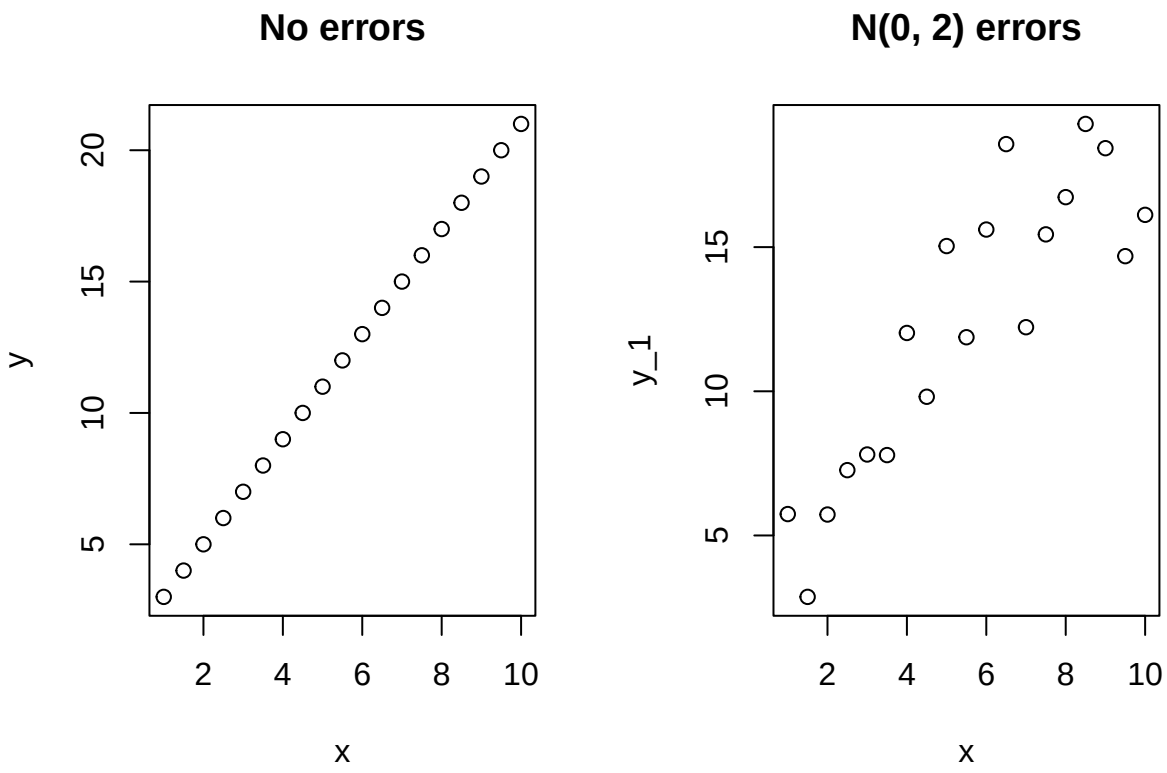
$$Y^* = X^*\beta + (\epsilon + V - U\beta).$$

This is clearly not the usual linear regression setting since X^* and $\epsilon + V - U\beta$ will be correlated. Moreover the error $\epsilon + V - U\beta$ may have a complicated structure that depends on the true regression coefficients β.

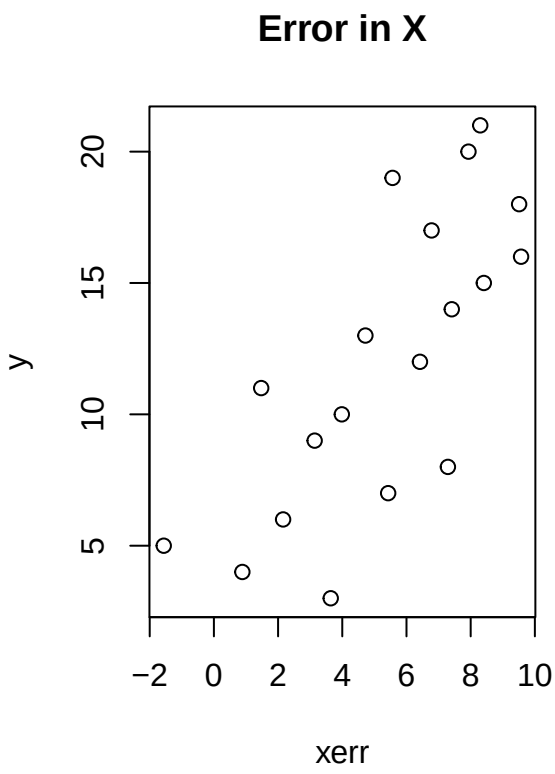
Artifical Example with Simulated Data

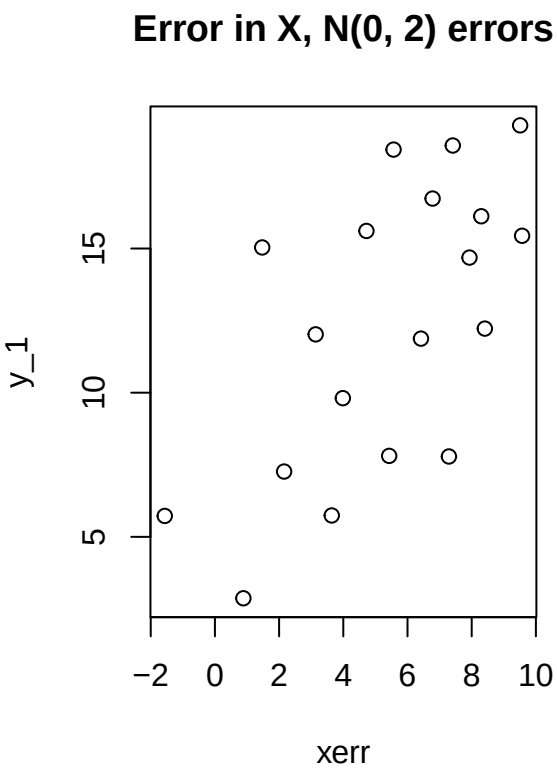
```
set.seed(42)
x<-seq(from=1, to=10, by=0.5)
n<-length(x)
y<-1+2*x
```

```
par(mfrow=c(1,2))
plot(x,y, main="No errors")
y_1<-y+rnorm(n, 0, 2)
plot(x, y_1, main="N(0, 2) errors")
```

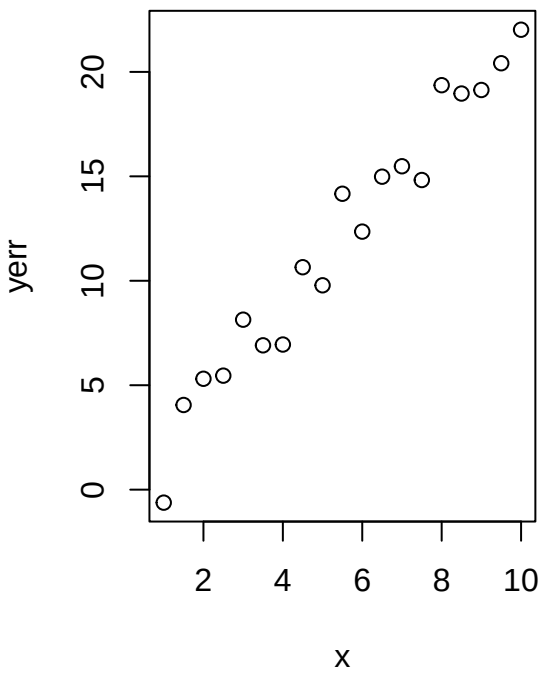


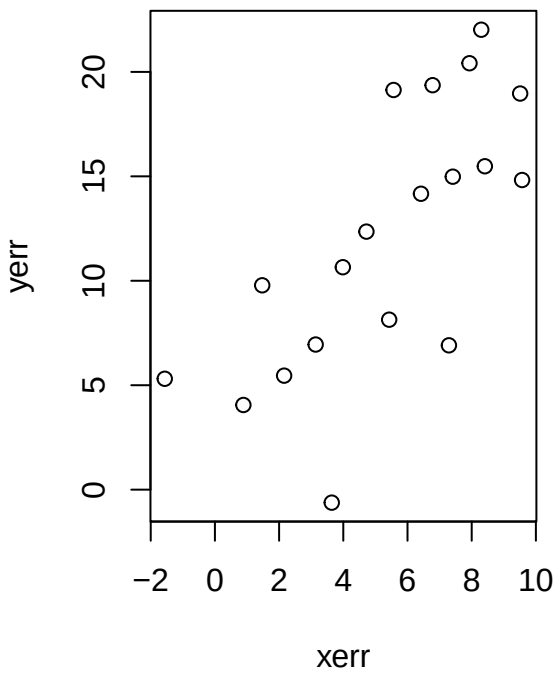

```
par(mfrow=c(1,2))
sx<-2
xerr<-x+rnorm(n,0,sx)
plot(xerr, y, main="Error in X")
plot(xerr, y_1, main="Error in X, N(0, 2) errors")
```






```
par(mfrow=c(1,2))
sy<-1.5
yerr<-y+rnorm(n,0,sy)
plot(x, yerr, main="Error in Y")
plot(xerr, yerr, main="Error in X, Y")
```

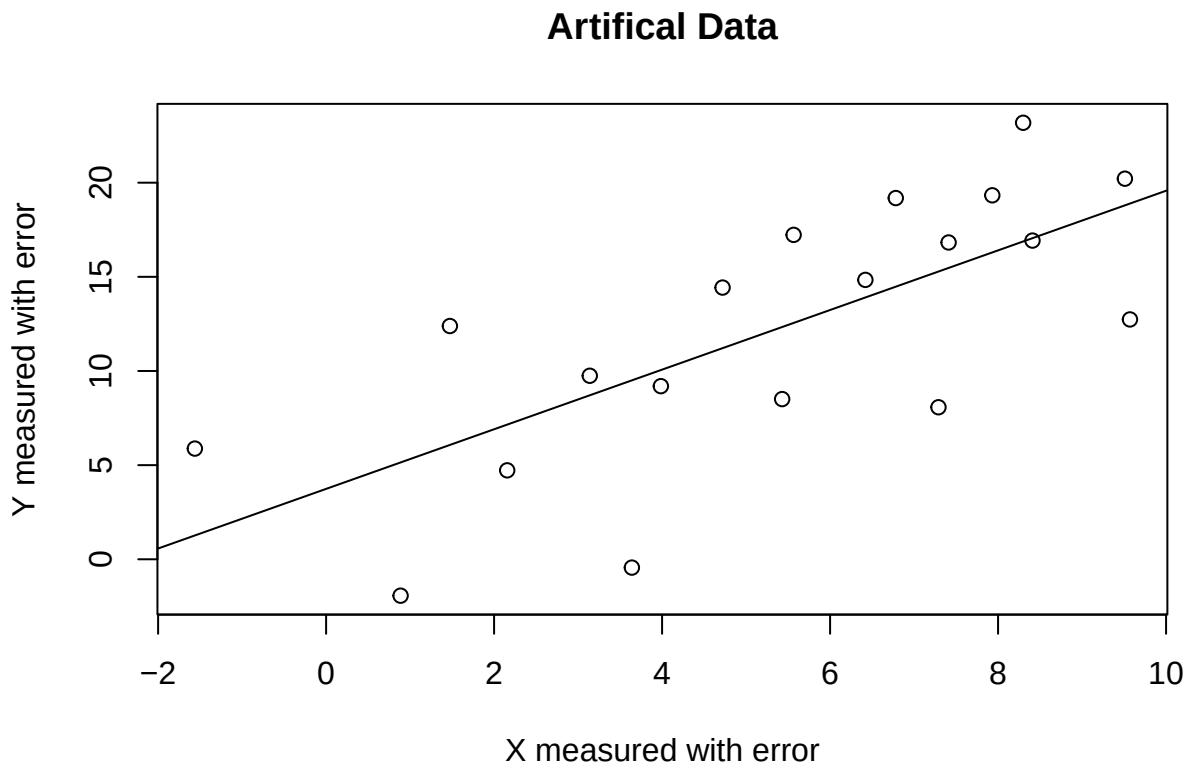




Exercise How do the above plots change if the measurement error in X is always positive? For example, if U is Gamma distributed?

Bayesian linear regression with reference prior ignoring measurement error

```
yerr1<-yerr+rnorm(n,0,2)
lm_mod<-lm(yerr1 ~ xerr)

plot(xerr, yerr1, xlab="X measured with error", ylab="Y measured with error", main="Artifical Data")
abline(lm_mod)
```




```
library(mvtnorm)
sse<-sum(lm_mod$residuals^2)

lam_shape<-lm_mod$df/2

mod_mat<-model.matrix(~ xerr)

XTXinv<-solve(t(mod_mat) %*% mod_mat)

msim<-1e4
post_sample<-matrix(NA_real_, ncol=3, nrow=msim)
for (iter in 1:msim){
  lam<-rgamma(1, shape=lam_shape, scale = 2/sse)
  cmat<-(1/lam)*XTXinv
  post_sample[iter,]<-c(rmvnorm(1, mean = lm_mod$coefficients, sigma = cmat),lam)
}
```

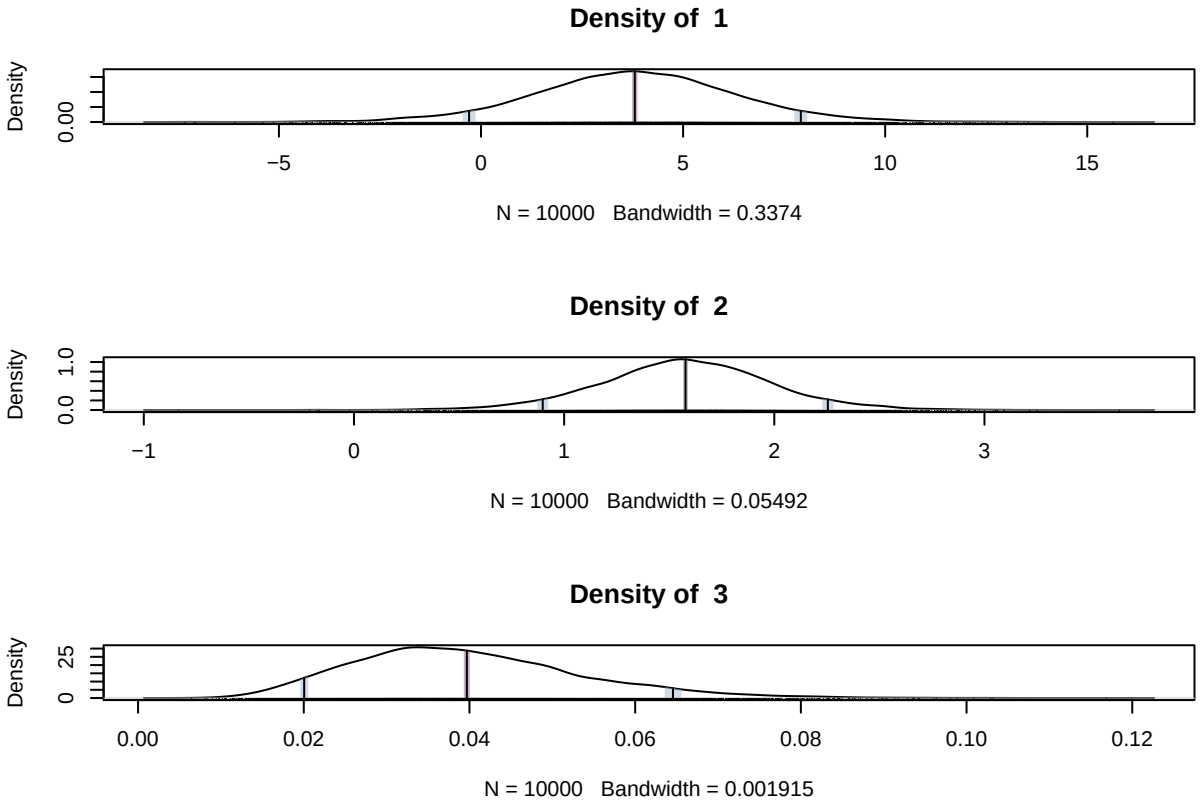
```r
apply(post_sample, 2, summary)
```

```
##              [,1]       [,2]        [,3]
## Min.    -7.326835 -0.8343135 0.006457172
## 1st Qu.  2.220944  1.3228051 0.029929327
## Median   3.799570  1.5775860 0.038166358
## Mean     3.805136  1.5775760 0.039683864
## 3rd Qu.  5.390489  1.8387807 0.047918320
## Max.    15.641120  3.6422028 0.116889146
```

```r
apply(post_sample, 2, quantile, probs = c(0.05, 0.95))
```

```
##           [,1]      [,2]       [,3]
## 5%  -0.2946204 0.8981433 0.02005049
## 95%  7.9134482 2.2548154 0.06456331
```

```r
library(SimTools)
S_post<-Siid(post_sample)

plot(S_post, Q=c(0.05,0.95))
```



Bayesian linear regression with measurement error

Accounting for measurement error in the covariate is, at least conceptually, straightforward:

$$\begin{aligned} Y_i|\beta, X_i, \tau_y &\overset{ind}{\sim} \mathrm{N}(x_i\beta, \tau_y^{-1}) \\ X_i^*|X_i, \tau_x &\overset{ind}{\sim} \mathrm{N}(x_i, \tau_x^{-1}) \\ X_i &\overset{ind}{\sim} \mathrm{N}(a_i, b_i^{-1}) \\ \beta &\sim \mathrm{N}_p(0, d^{-1}I_p) \\ \tau_x &\sim \mathrm{Gamma}(e_x, f_x) \\ \tau_y &\sim \mathrm{Gamma}(e_y, f_y) \end{aligned}$$

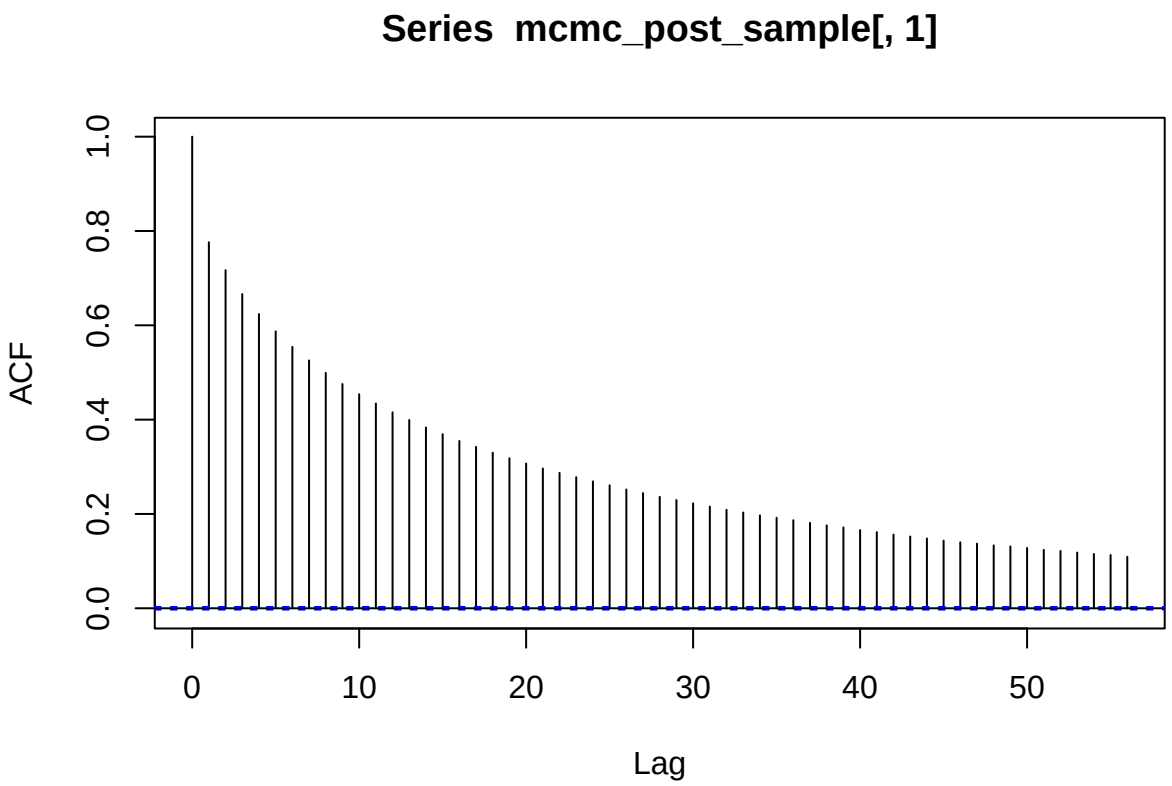
Standard reference priors incorporate the error in X^* with $a_i = 0$ and the b_i estimated, then d^{-1} is taken to be at least 1e3. Finally, the standard recommendation is to choose $e_x = e_y = f_x = f_y = 0.5$ which essentially mimics a unit information prior.

```
library(R2jags)
library(mcmcplots)
library(mcmcse)

# Identify variables
N <- length(yerr1)  # number of data points
obsx <- xerr # covariate measured with error
errx <- sx # standard deviation used to simulate data
obsy <- yerr1 # response

# Prepare data to JAGS
lm_data <- list(
  obsx = obsx,
  obsy = obsy,
  errx = errx,
  N = N
)

# Fit
NORM_errors <-"model{
# Normal priors for predictors
# dnorm in JAGS is different than dnorm in R --dnorm(a,b) means a Normal rv with mean
# a and precision b
beta_int ~ dnorm(0,1e-5)
beta_slope ~ dnorm(0,1e-5)

# Gamma prior for scatter
tau_y ~ dgamma(0.5, 0.5) #precision


# Normal priors for true x
for (i in 1:N){ x[i] ~ dnorm(0,1e-5) }


for (i in 1:N){
obsx[i] ~ dnorm(x[i], 1/errx^2) #errx is the sd used to generate the data
obsy[i] ~ dnorm(mu[i], tau_y)
```

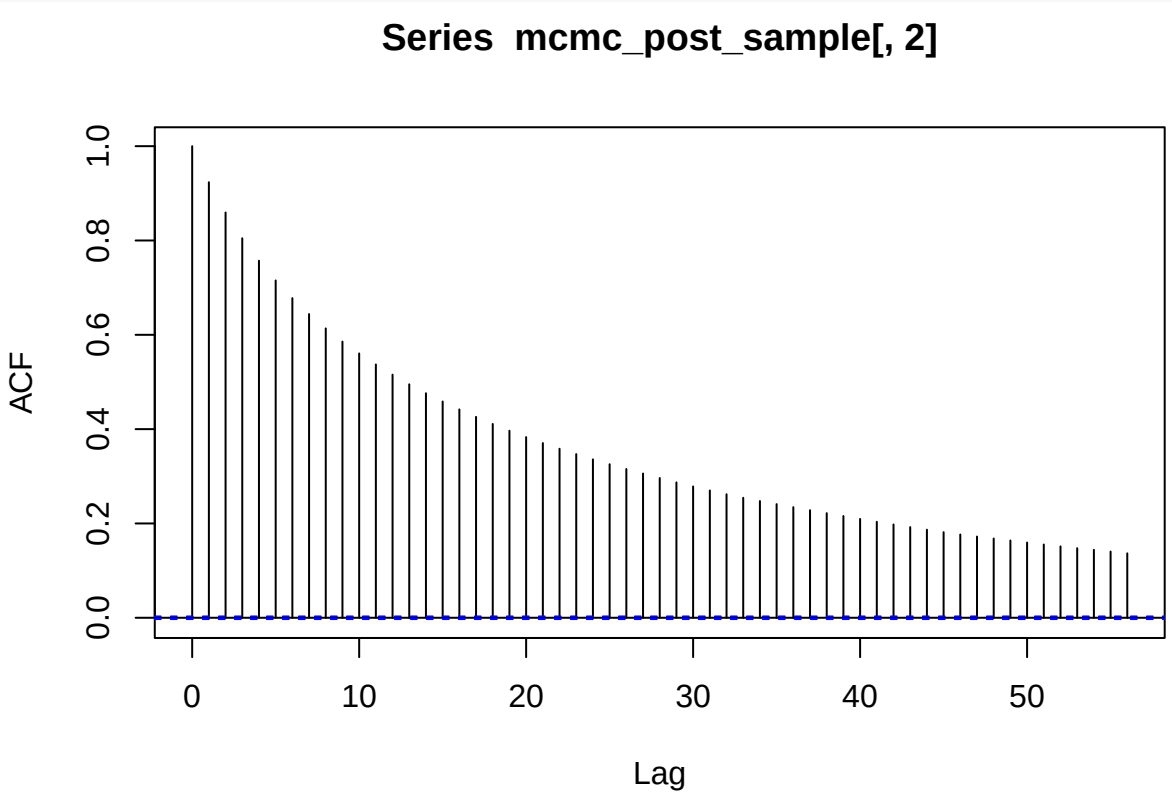
```
mu[i] <- beta_int + beta_slope*x[i] # linear predictor
}
}"

# Define initial values
inits <- function () {
  list(beta_int=lm_mod$coefficients[1],
       beta_slope=lm_mod$coefficients[2])
}

# Identify parameters
params0 <- c("beta_int","beta_slope", "tau_y")

# Fit
NORM_fit <- jags(data = lm_data,
                 inits = inits,
                 parameters = params0,
                 model = textConnection(NORM_errors),
                 n.chains = 1,
                 n.iter = 5e5,
                 n.thin = 1,
                 n.burnin = 0,
                 jags.seed = 528
)
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 38
##    Unobserved stochastic nodes: 22
##    Total graph size: 107
##
## Initializing model
```

```
mcmc_post_sample <- as.mcmc(NORM_fit)

acf(mcmc_post_sample[,1])
```




```
acf(mcmc_post_sample[,2])
```




```
apply(mcmc_post_sample[,1:2], 2, ess)
```

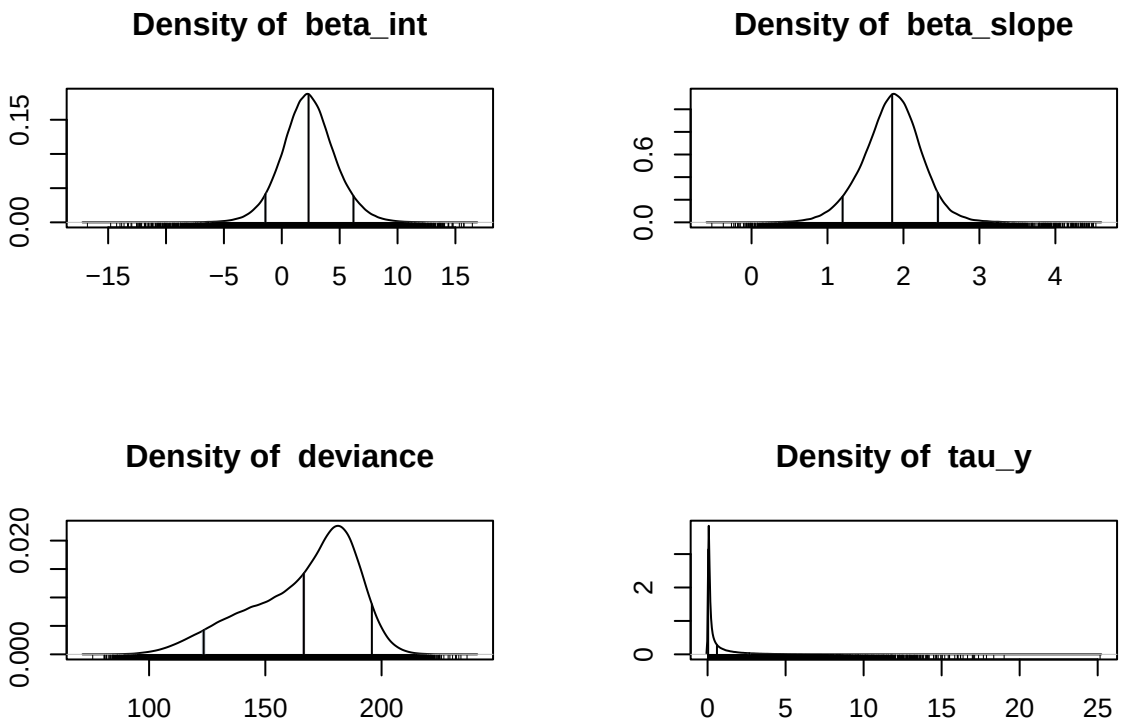
```
##   beta_int beta_slope
```

```
##  10541.970   8514.464
```

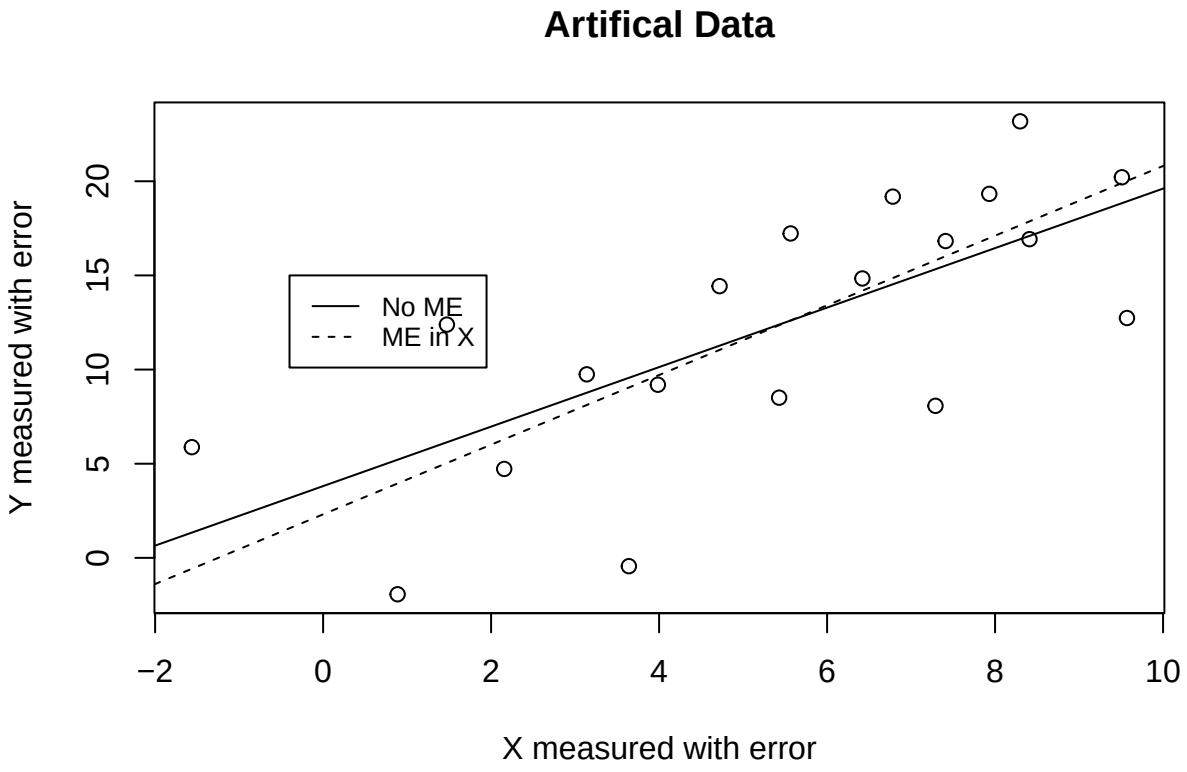
```
# Output
print(NORM_fit,intervals=c(0.05, 0.95), justify = "left", digits=2)
```

```
## Inference for Bugs model at "4", fit using jags,
##  1 chains, each with 5e+05 iterations (first 0 discarded)
##  n.sims = 500000 iterations saved
##            mu.vect sd.vect     5%    95%
## beta_int      2.31    2.33  -1.41   6.19
## beta_slope    1.85    0.39   1.20   2.45
## tau_y         0.60    1.08   0.04   2.67
## deviance    166.55   22.50 123.48 195.83
##
## DIC info (using the rule, pD = var(deviance)/2)
## pD = 253.2 and DIC = 419.8
## DIC is an estimate of expected predictive error (lower deviance is better).
```

```
S_post<-Smcmc(mcmc_post_sample)
plot(S_post, Q=c(0.05, 0.95),  xlab="", ylab="")
```




```
plot(xerr, yerr1, xlab="X measured with error", ylab="Y measured with error", main="Artifical Data")
abline(3.81, 1.58, lty=1)
abline(2.31, 1.85, lty=2)
legend(-0.4, 15, legend=c("No ME", "ME in X"), lty=1:2, cex=0.8)
```



Bayesian linear regression with measurement error

We can also account for measurement error in response and covariate:

$$
\begin{aligned}
Y_i^* | Y_i &\overset{ind}{\sim} \mathrm{N}(Y_i, a^{-1}) \\
Y_i | X^*, \beta &\overset{ind}{\sim} \mathrm{N}(X_i^* \beta, \tau^{-1}) \\
X_i^* | X_i &\overset{ind}{\sim} \mathrm{N}_p(X_i, diag(b_i^{-1})) \\
X_i &\overset{iid}{\sim} \mathrm{N}_p(0, c^{-1} I_p) \\
\beta &\sim \mathrm{N}_p(0, d^{-1} I_p) \\
\tau &\sim \mathrm{Gamma}(e, f)
\end{aligned}
$$

Standard reference priors incorporate the estimated error in Y^* and X^* through the a and the b_i, then c^{-1} and d^{-1} are taken to be at least 1e3. Finally, the standard recommendation is to choose $e = f = 0.5$ which essentially mimics a unit information prior.

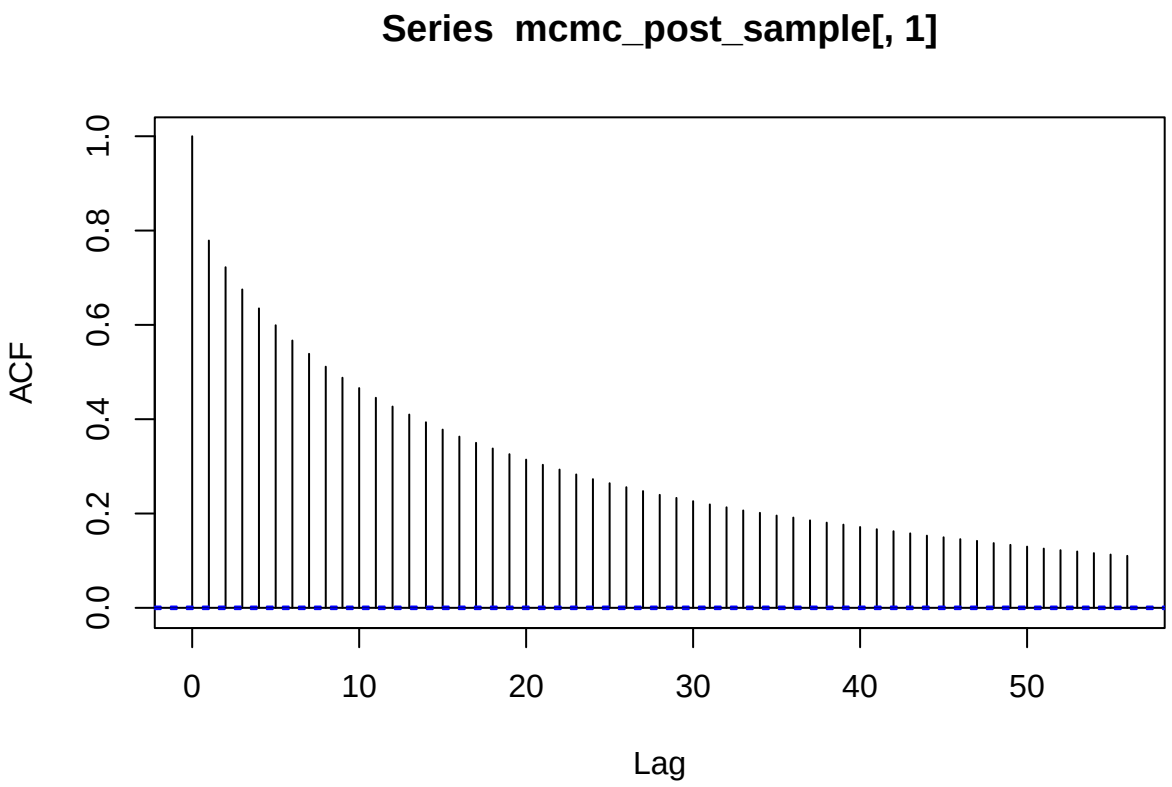
```
library(R2jags)
library(mcmcplots)
library(mcmcse)

# Identify variables
N <- length(yerr1)  # number of data points
obsx <- xerr # covariate measured with error
errx <- sx # standard deviation used to simulate data
obsy <- yerr1 # response measured with error
erry <- sy # standard deviation used to simulate data
```

```r
# Prepare data to JAGS
lm_data <- list(
  obsx = obsx,
  obsy = obsy,
  errx = errx,
  erry = erry,
  N = N
)

# Fit
NORM_errors <-"model{
# Normal priors for predictors
beta_int ~ dnorm(0,1e-5)
beta_slope ~ dnorm(0,1e-5)

# Gamma prior for scatter
tau ~ dgamma(0.5,0.5) # precision


# Normal priors for true x
for (i in 1:N){ x[i] ~ dnorm(0,1e-5) }

for (i in 1:N){
obsx[i] ~ dnorm(x[i], 1/errx^2)
obsy[i] ~ dnorm(y[i], 1/erry^2)
y[i] ~ dnorm(mu[i], tau)
mu[i] <- beta_int + beta_slope*x[i] # linear predictor
}
}"

# Define initial values
inits <- function () {
  list(beta_int=lm_mod$coefficients[1],
       beta_slope=lm_mod$coefficients[2])
}

# Identify parameters
params0 <- c("beta_int","beta_slope", "tau")

# Fit
NORM_fit <- jags(data = lm_data,
                 inits = inits,
                 parameters = params0,
                 model = textConnection(NORM_errors),
                 n.chains = 1,
                 n.iter = 5e5,
                 n.thin = 1,
                 n.burnin = 0,
                 jags.seed = 731
)
```

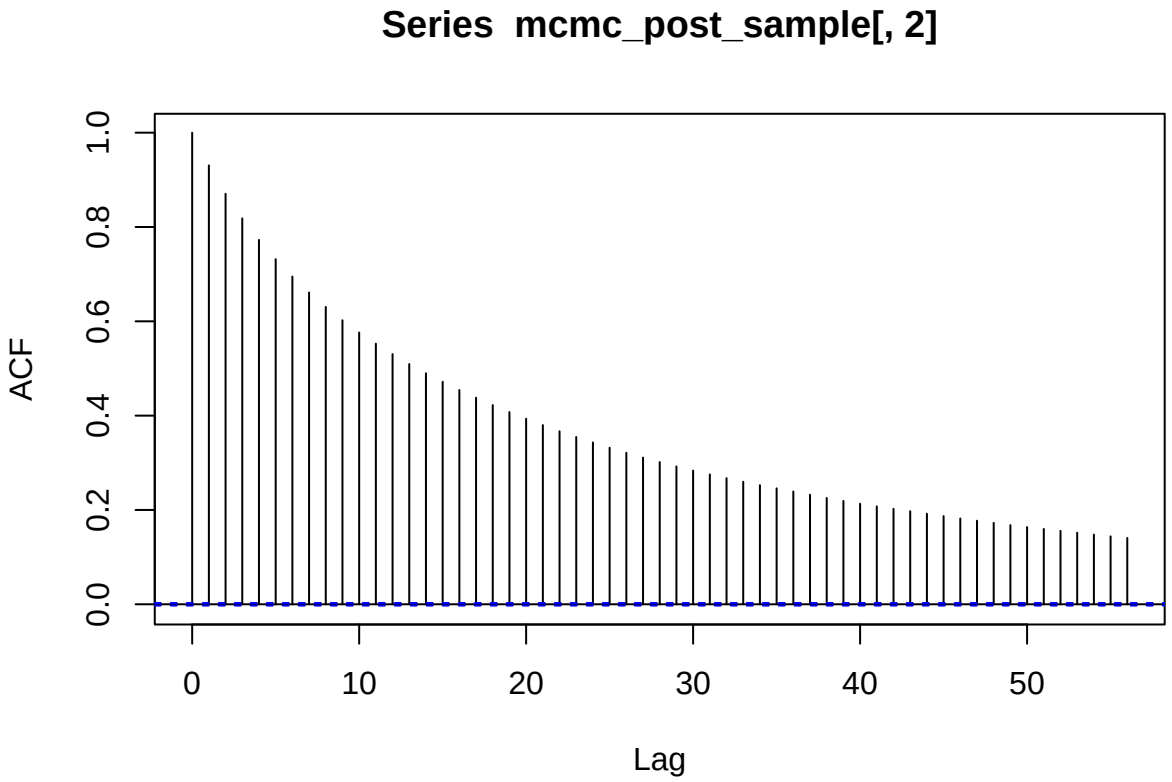
```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
```

```
## Graph information:
##    Observed stochastic nodes: 38
##    Unobserved stochastic nodes: 41
##    Total graph size: 129
##
## Initializing model
```

```
mcmc_post_sample <- as.mcmc(NORM_fit)

acf(mcmc_post_sample[,1])
```




```
acf(mcmc_post_sample[,2])
```



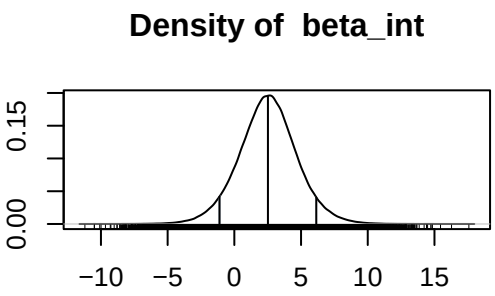

```
apply(mcmc_post_sample[,1:2], 2, ess)
```

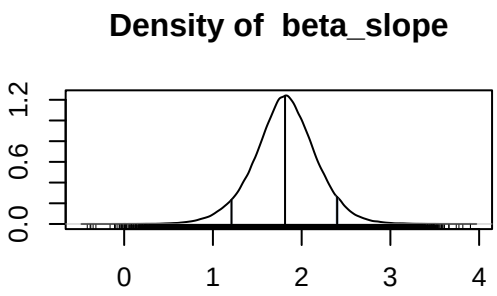
```
##   beta_int beta_slope
##  11180.235   9131.611
```

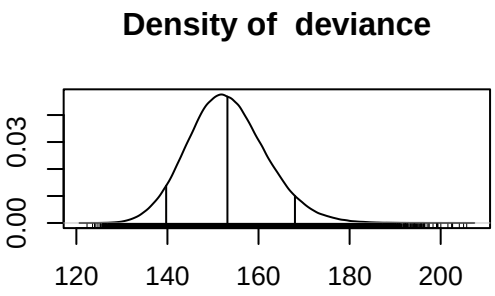
```
# Output
print(NORM_fit,intervals=c(0.05, 0.95), justify = "left", digits=2)
```

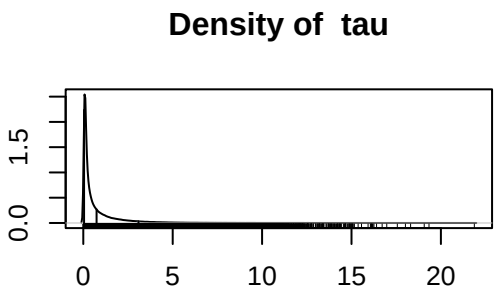
```
## Inference for Bugs model at "5", fit using jags,
##  1 chains, each with 5e+05 iterations (first 0 discarded)
##  n.sims = 500000 iterations saved
##            mu.vect sd.vect     5%    95%
## beta_int      2.52    2.21  -1.11   6.14
## beta_slope    1.81    0.36   1.21   2.40
## tau           0.74    1.19   0.04   3.09
## deviance    153.17    8.63 139.72 168.00
##
## DIC info (using the rule, pD = var(deviance)/2)
## pD = 37.2 and DIC = 190.4
## DIC is an estimate of expected predictive error (lower deviance is better).
```

```
S_post<-Smcmc(mcmc_post_sample)
plot(S_post, Q=c(0.05, 0.95),  xlab="", ylab="")
```

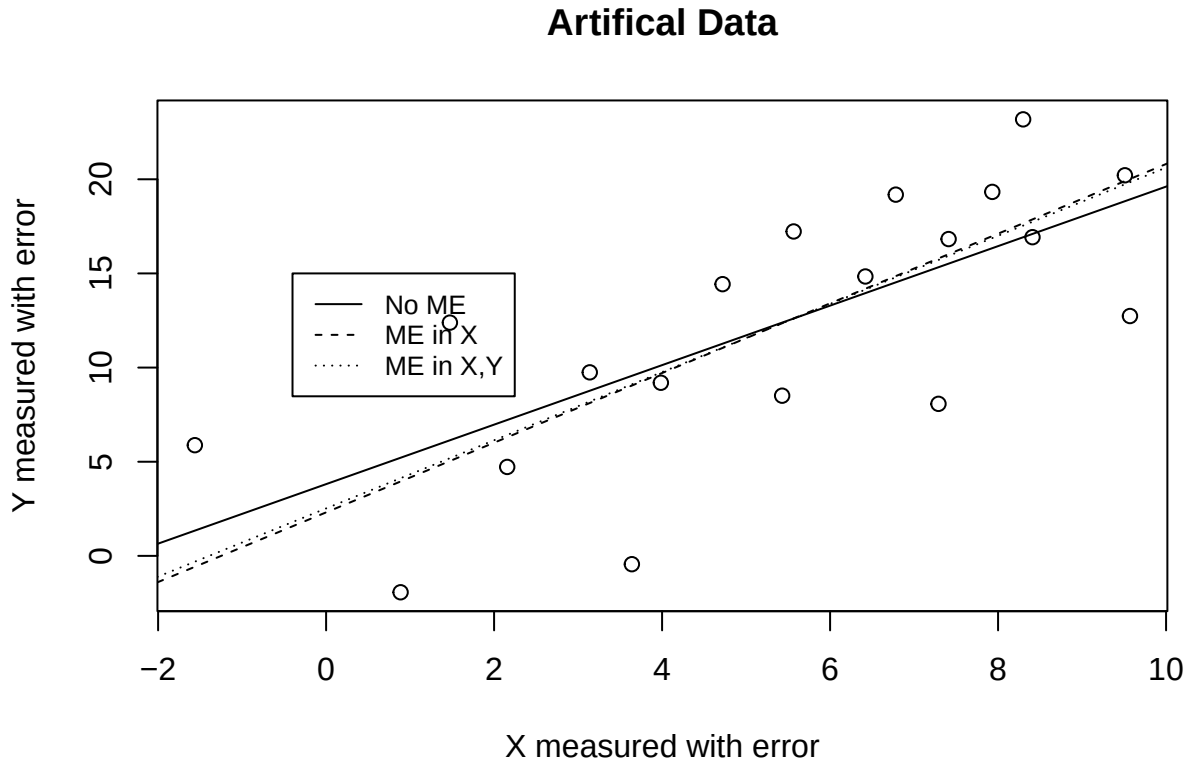









```
plot(xerr, yerr1, xlab="X measured with error", ylab="Y measured with error", main="Artifical Data")
abline(3.81, 1.58, lty=1)
abline(2.31, 1.85, lty=2)
abline(2.52, 1.81, lty=3)
legend(-0.4, 15, legend=c("No ME", "ME in X", "ME in X,Y"), lty=1:3, cex=0.8)
```



Hierarchical Models

The linear model with measurement error is an example of a hierarchical model. To this point we have required that there be a sampling pf $p(x|\theta)$ and a prior pf $\pi(\theta)$. Formally, this is a (one-stage) hierarchical model. This idea can be extended in a way that will be helpful when we encounter more complicated settings. Notice that π is also typically from a parametric family so itwould be better to write $\pi(\theta|\theta_0)$ where θ_0, to this point, has been treated as a hyperparameter that is specified by the analyst. If the analyst is uncertain about the value of θ_0, then it is natural to quantify this through a prior, which leads to a generic two-stage hierarchical model

$$X|\theta_1 \sim p(x|\theta_1), \qquad \theta_1|\theta_0 \sim \pi_1(\theta_1|\theta_0), \qquad \theta_0 \sim \pi_0(\theta_0). \tag{1}$$

The resulting posterior (if it exists) is then characterized by

$$q(\theta_0, \theta_1|x) \propto p(x|\theta_1)\pi_1(\theta_1|\theta_0)\pi_0(\theta_0).$$

Now this is a joint pf and inference will often be based on the univariate marginal pfs which can be obtained by integration:

$$q(\theta_0|x) \propto \int q(\theta_0, \theta_1|x)d\theta_1$$

and

$$q(\theta_1|x) \propto \int q(\theta_0, \theta_1|x)d\theta_0.$$

There are two ways to view the hierarchical model. Notice that

$$p(x|\theta_0) = \int p(x|\theta_1)\pi_1(\theta_1|\theta_0)d\theta_1$$

defines a legitimate sampling density for $X|\theta_0$. Thus the model defined in above is equivalent to using the model

$$p(x|\theta_0) \qquad \theta_0 \sim \pi_0(\theta_0).$$

On the other hand,

$$\pi(\theta_1) = \int \pi_1(\theta_1|\theta_0)\pi_0(\theta_0)d\theta_0$$

defines a legitimate prior density and hence the model defined above is equivalent to using the model

$$p(x|\theta_1) \qquad \theta \sim \pi_1(\theta_1).$$

Thus there is some ambiguity in what exactly is serving as *the* prior.

Exercise The following model is proposed by Hilbe, et al (2017). If

$$\mu_i = \beta_1 + \beta_2 X_i,$$

then, for $i = 1, \ldots, n$

$$Y_i|\beta_1, \beta_2, X_i \overset{ind}{\sim} \text{Poisson}(\mu_i)$$

and independently

$$\beta_1 \sim \text{N}(0, 10^3) \qquad \beta_2 \sim \text{N}(0, 10^3).$$

There is an inconsistency between the priors and the likelihood. What is it? [Hint: Think about what values of μ_i are permitted by the priors and what values are permitted by the sampling model.]

M-sigma relationship: data

This data consists of 46 observations of the mass of supermassive black holes at the center of a galaxy and the velocity dispersion of the stars in its bulge. The logarithm of the black hole mass is taken to be $\log(M_\bullet/M_\odot)$ and the logarithm of the velocity dispersion of the stars in its bulge is $\log(\sigma_e/\sigma_0)$ with σ_0 a known reference value. The hypothesized relationship is

$$\log(M_\bullet/M_\odot) = \beta_1 + \beta_2 \log(\sigma_e/\sigma_0)$$

where β_1 and β_2 are unknown. To make the notation a little bit lighter set $Y = \log(M_\bullet/M_\odot)$, $X = (1, \log(\sigma_e/\sigma_0))$, and $\beta = (\beta_1, \beta_2)^T$ yielding

$$Y = \beta_1 + \beta_2 x_2 = x\beta.$$

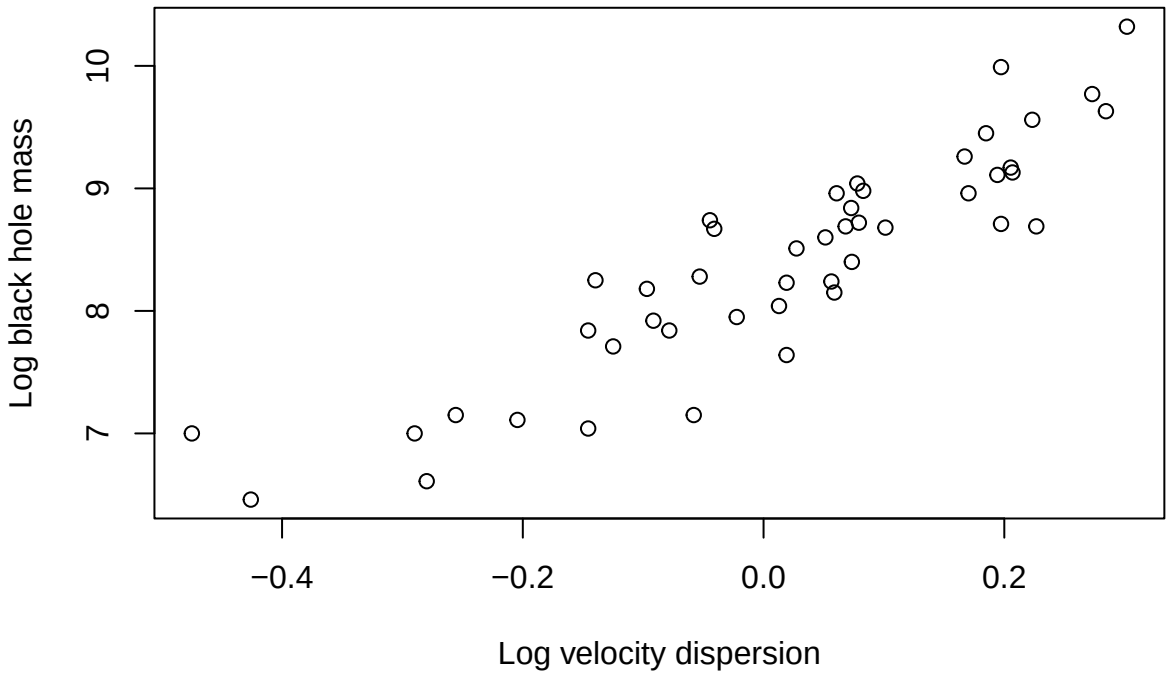
```
library(curl)
```

```
## Using libcurl 7.64.1 with LibreSSL/2.8.3
```

```
bh_dat<-read.csv(curl("https://raw.githubusercontent.com/astrobayes/BMAD/master/data/Section_10p1/M_sig
summary(bh_dat)
```

```
##       obsx               errx              obsy             erry
##  Min.   :-0.47496   Min.   :0.01450   Min.   : 6.460   Min.   :0.0300
##  1st Qu.:-0.08821   1st Qu.:0.03500   1st Qu.: 7.860   1st Qu.:0.0800
##  Median : 0.05384   Median :0.04000   Median : 8.555   Median :0.1250
##  Mean   : 0.01567   Mean   :0.04663   Mean   : 8.399   Mean   :0.1465
##  3rd Qu.: 0.16945   3rd Qu.:0.05875   3rd Qu.: 8.975   3rd Qu.:0.1938
##  Max.   : 0.30190   Max.   :0.10000   Max.   :10.320   Max.   :0.4350
##      Type
##  Length:46
##  Class :character
##  Mode  :character
##
##
##
```

```
lbhm<-bh_dat$obsy
lvd<-bh_dat$obsx

plot(lvd, lbhm, xlab="Log velocity dispersion", ylab="Log black hole mass", main="")
```



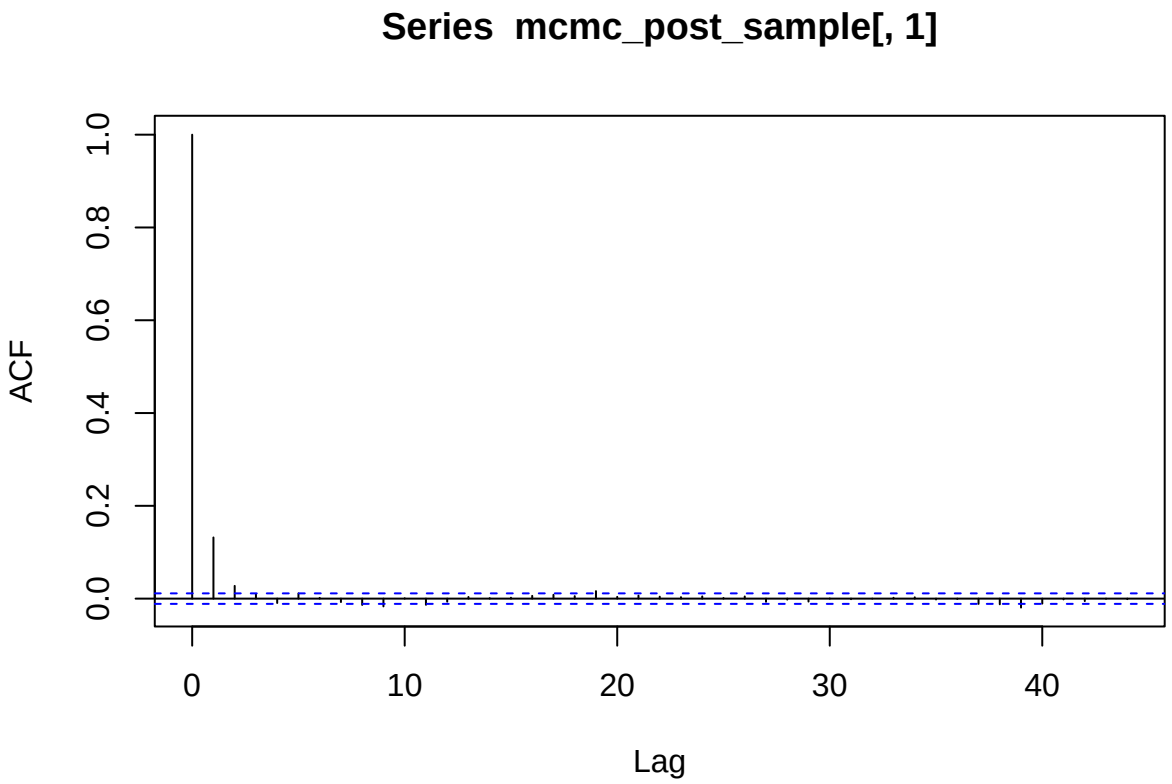
M-sigma relationship with measuremment error: reference priors

```
library(R2jags)
library(mcmcplots)
library(mcmcse)

lbhm_mod<-lm(lbhm ~ lvd)

# Identify variables
N <- nrow(bh_dat) # number of data points
obsx <- bh_dat$obsx # log velocity dispersion
errx <- bh_dat$errx # error on log velocity dispersion
obsy <- bh_dat$obsy # log black hole mass
erry <- bh_dat$erry # error on log black hole mass

# Prepare data to JAGS
bh_data <- list(
  obsx = obsx,
  obsy = obsy,
  errx = errx,
  erry = erry,
  N = N
)

# Fit
NORM_errors <-"model{
# Normal priors for predictors
beta_int ~ dnorm(0,1e-5)
beta_slope ~ dnorm(0,1e-5)
```

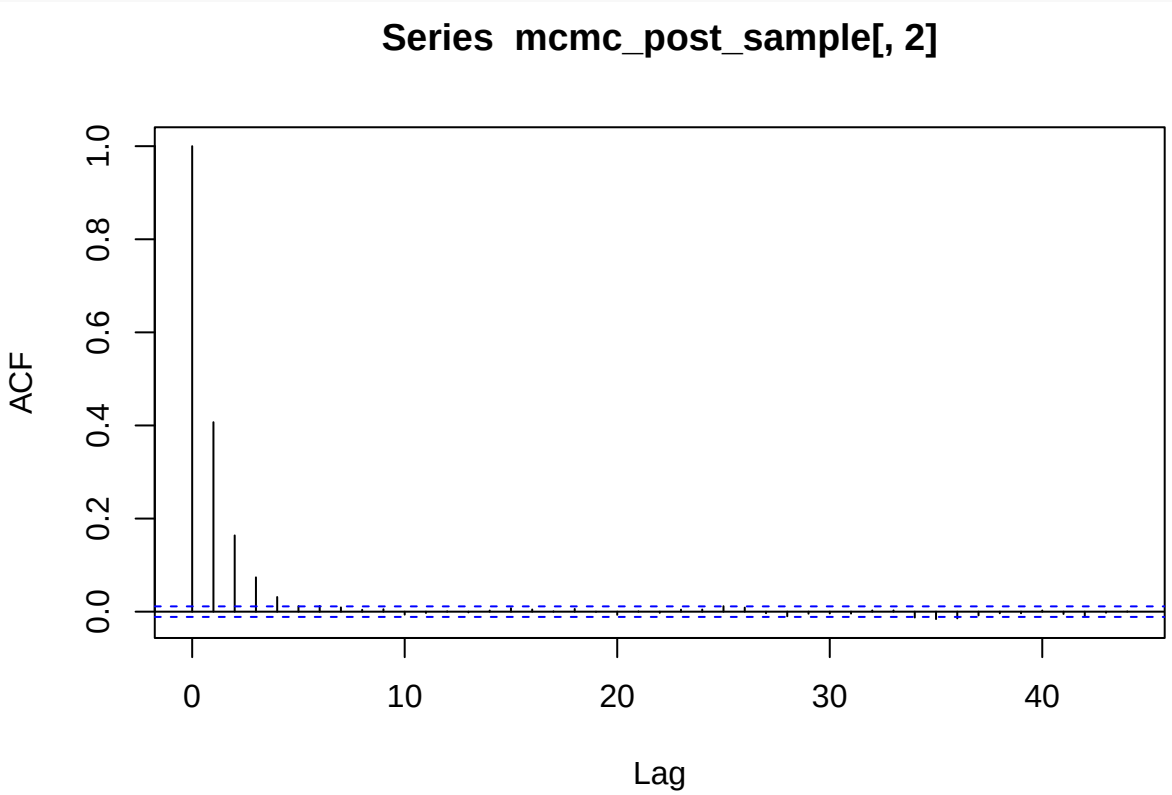
```
# Gamma prior for scatter
tau ~ dgamma(0.5,0.5) # precision
epsilon <- 1/sqrt(tau) # intrinsic scatter

# Normal priors for true x
for (i in 1:N){ x[i] ~ dnorm(0,1e-5) }

for (i in 1:N){
obsx[i] ~ dnorm(x[i], 1/errx[i]^2)
obsy[i] ~ dnorm(y[i], 1/erry[i]^2) # likelihood function
y[i] ~ dnorm(mu[i], tau)
mu[i] <- beta_int + beta_slope*x[i] # linear predictor
}
}"

# Define initial values
inits <- function () {
  list(beta_int=lbhm_mod$coefficients[1],
       beta_slope=lbhm_mod$coefficients[2])
}

# Identify parameters
params0 <- c("beta_int","beta_slope", "tau")

# Fit
NORM_fit <- jags(data = bh_data,
                 inits = inits,
                 parameters = params0,
                 model = textConnection(NORM_errors),
                 n.chains = 1,
                 n.iter = 3e4,
                 n.thin = 1,
                 n.burnin = 0,
                 jags.seed = 407
)
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 92
##    Unobserved stochastic nodes: 95
##    Total graph size: 469
##
## Initializing model
```

```
mcmc_post_sample <- as.mcmc(NORM_fit)

acf(mcmc_post_sample[,1])
```




```
acf(mcmc_post_sample[,2])
```




```
apply(mcmc_post_sample[,1:2], 2, ess)
```

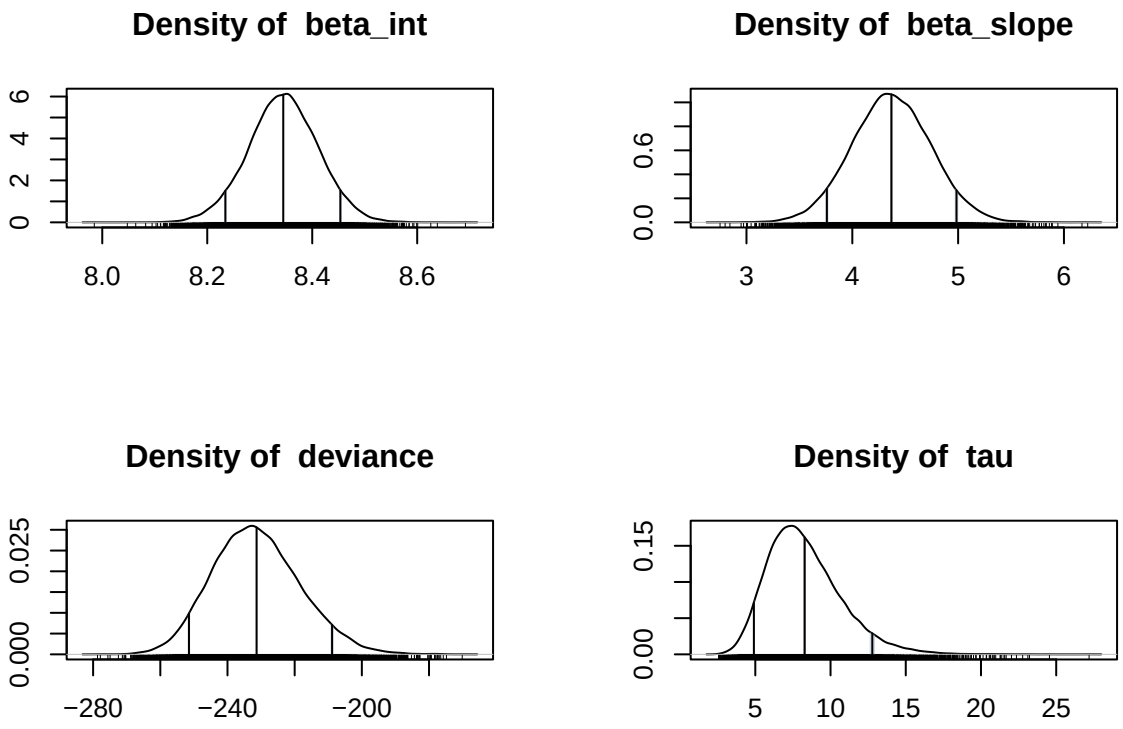
```
##   beta_int beta_slope
```

```
##   21343.39   11331.94
```

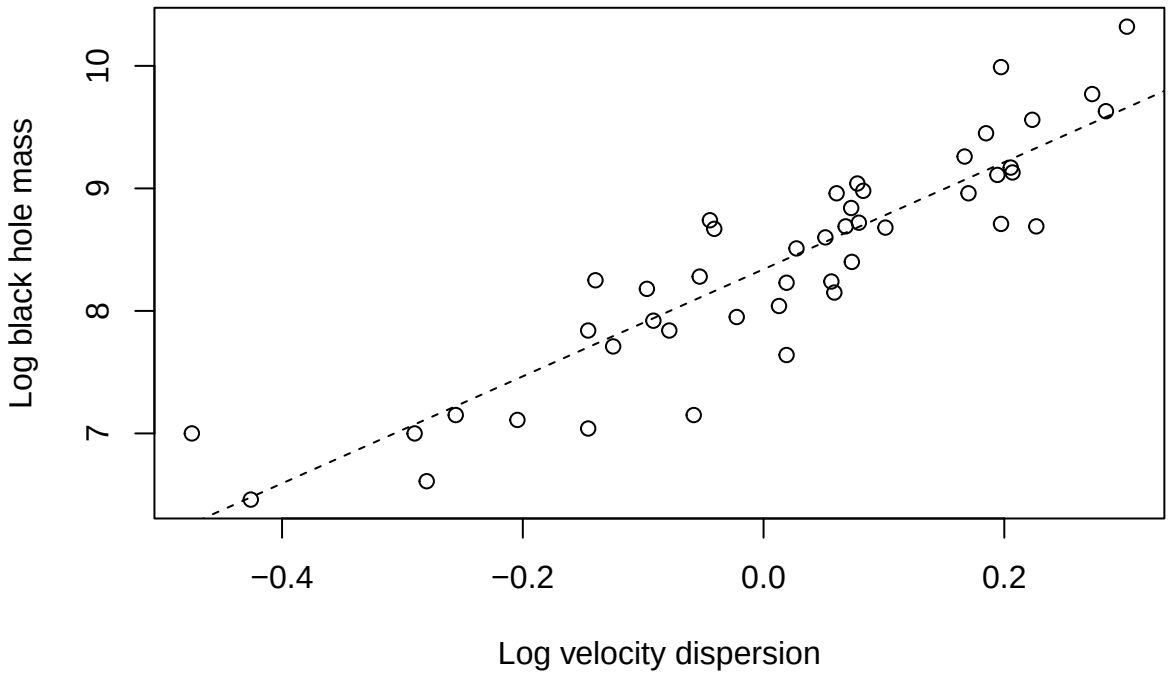
```
# Output
print(NORM_fit,intervals=c(0.05, 0.95), justify = "left", digits=2)
```

```
## Inference for Bugs model at "6", fit using jags,
##  1 chains, each with 30000 iterations (first 0 discarded)
##  n.sims = 30000 iterations saved
##            mu.vect sd.vect      5%     95%
## beta_int      8.34    0.07    8.23    8.45
## beta_slope    4.37    0.37    3.76    4.98
## tau           8.28    2.46    4.91   12.78
## deviance   -231.33   13.02 -251.47 -208.86
##
## DIC info (using the rule, pD = var(deviance)/2)
## pD = 84.8 and DIC = -146.5
## DIC is an estimate of expected predictive error (lower deviance is better).
```

```
S_post<-Smcmc(mcmc_post_sample)
plot(S_post, Q=c(0.05, 0.95),  xlab="", ylab="")
```




```
plot(lvd, lbhm, xlab="Log velocity dispersion", ylab="Log black hole mass", main="")
abline(8.34, 4.37, lty=2)
```



M-sigma relationship with measuremment error: informed priors

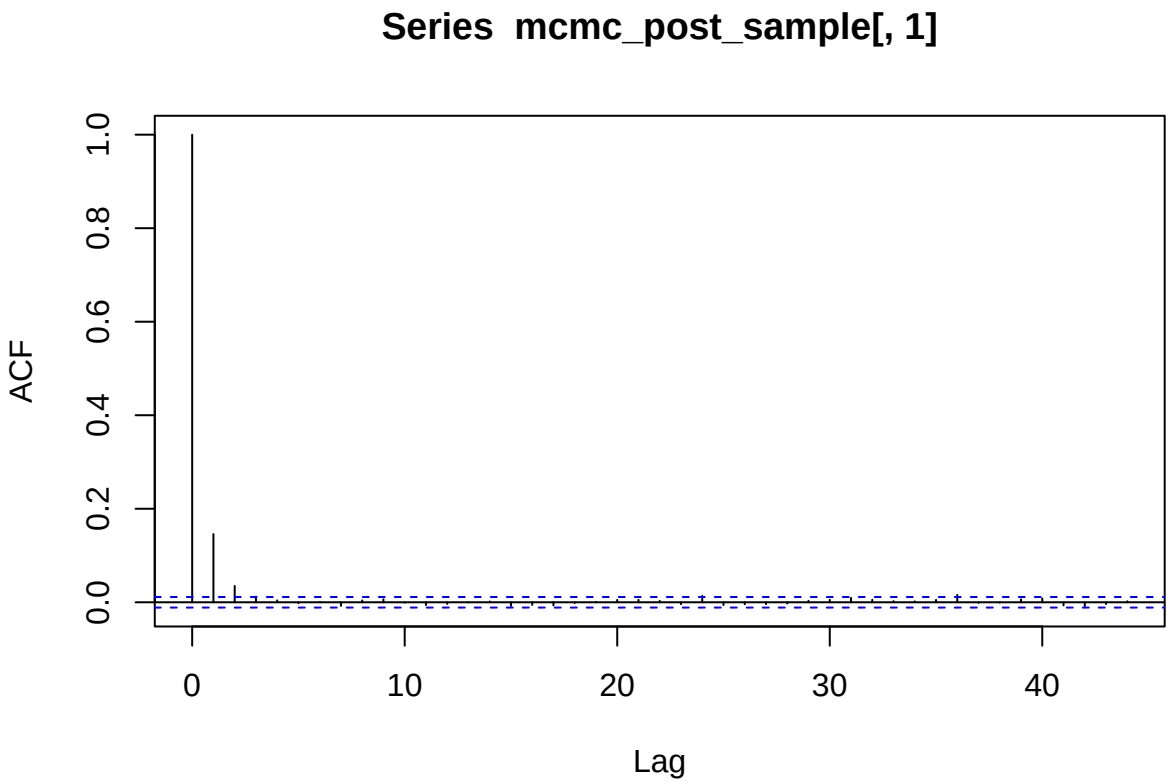
We center the prior of the regression coefficients at the values found by Merritt and Ferrarese (2018) [arXiv:astro-ph/0008310].

```
library(R2jags)
library(mcmcplots)
library(mcmcse)

lbhm_mod<-lm(lbhm ~ lvd)

# Identify variables
N <- nrow(bh_dat) # number of data points
obsx <- bh_dat$obsx # log velocity dispersion
errx <- bh_dat$errx # error on log velocity dispersion
obsy <- bh_dat$obsy # log black hole mass
erry <- bh_dat$erry # error on log black hole mass

# Prepare data to JAGS
bh_data <- list(
  obsx = obsx,
  obsy = obsy,
  errx = errx,
  erry = erry,
  N = N
)

# Fit
NORM_errors <-"model{
# Normal priors for predictors
beta_int ~ dnorm(-2.95, 0.72) #Taken from Table 3 of Merritt and Ferrarese
```

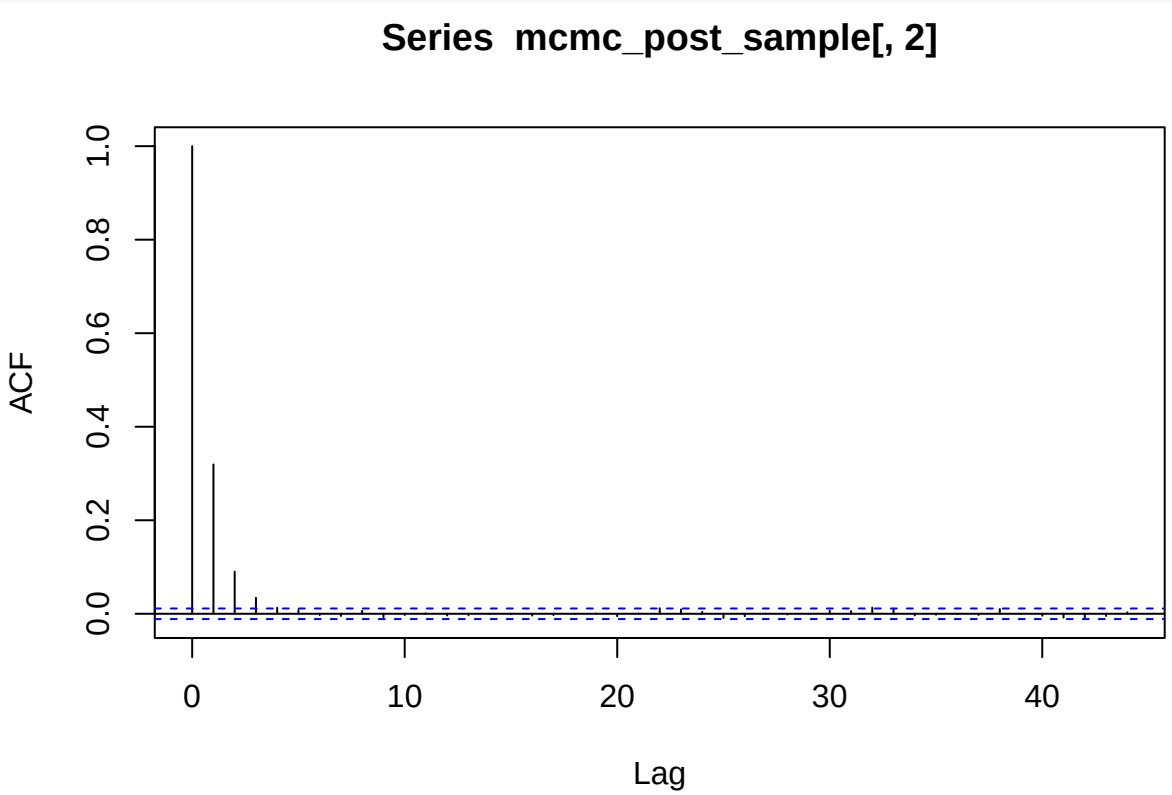
```
beta_slope ~ dnorm(4.81, 4)

# Gamma prior for scatter
tau ~ dgamma(0.5,0.5) # precision
epsilon <- 1/sqrt(tau) # intrinsic scatter

# Normal priors for true x
for (i in 1:N){ x[i] ~ dnorm(0,1e-5) }

for (i in 1:N){
obsx[i] ~ dnorm(x[i], 1/errx[i]^2)
obsy[i] ~ dnorm(y[i], 1/erry[i]^2) # likelihood function
y[i] ~ dnorm(mu[i], tau)
mu[i] <- beta_int + beta_slope*x[i] # linear predictor
}
}"

# Define initial values
inits <- function () {
  list(beta_int=lbhm_mod$coefficients[1],
       beta_slope=lbhm_mod$coefficients[2])
}

# Identify parameters
params0 <- c("beta_int","beta_slope", "tau")

# Fit
NORM_fit <- jags(data = bh_data,
                 inits = inits,
                 parameters = params0,
                 model = textConnection(NORM_errors),
                 n.chains = 1,
                 n.iter = 3e4,
                 n.thin = 1,
                 n.burnin = 0,
                 jags.seed = 1228
)
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 92
##    Unobserved stochastic nodes: 95
##    Total graph size: 474
##
## Initializing model
```

```
mcmc_post_sample <- as.mcmc(NORM_fit)

acf(mcmc_post_sample[,1])
```




```
acf(mcmc_post_sample[,2])
```




```
apply(mcmc_post_sample[,1:2], 2, ess)
```

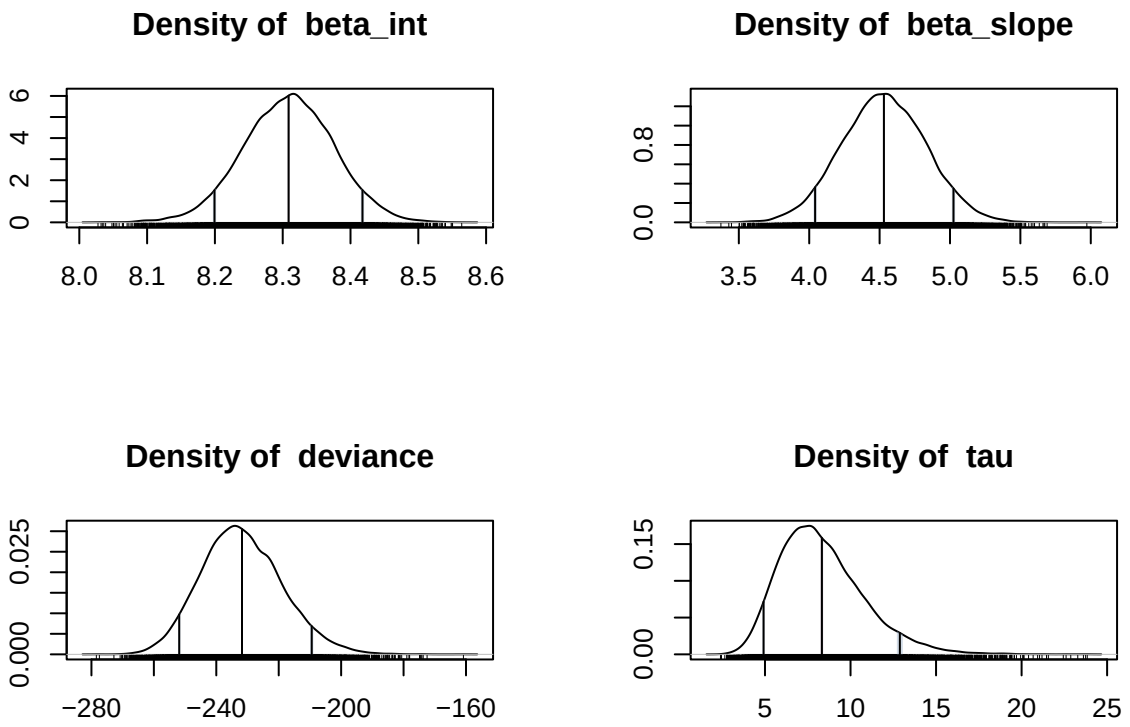
```
##   beta_int beta_slope
```

```
##   20557.63   15281.14
```

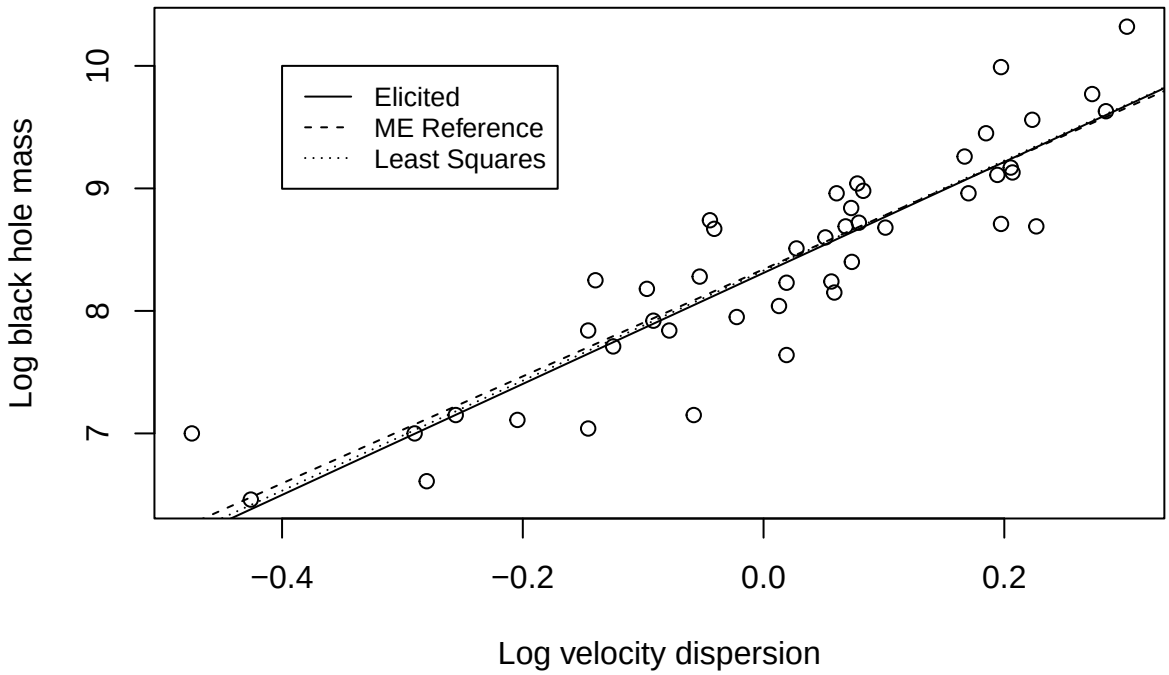
```
# Output
print(NORM_fit,intervals=c(0.05, 0.95), justify = "left", digits=2)
```

```
## Inference for Bugs model at "7", fit using jags,
##  1 chains, each with 30000 iterations (first 0 discarded)
##  n.sims = 30000 iterations saved
##            mu.vect sd.vect      5%     95%
## beta_int      8.31    0.07    8.20    8.42
## beta_slope    4.53    0.30    4.04    5.02
## tau           8.33    2.47    4.92   12.88
## deviance   -231.77   12.94 -251.87 -209.43
##
## DIC info (using the rule, pD = var(deviance)/2)
## pD = 83.7 and DIC = -148.1
## DIC is an estimate of expected predictive error (lower deviance is better).
```

```
S_post<-Smcmc(mcmc_post_sample)
plot(S_post, Q=c(0.05, 0.95),  xlab="", ylab="")
```




```
plot(lvd, lbhm, xlab="Log velocity dispersion", ylab="Log black hole mass", main="")
abline(8.31, 4.53, lty=1)
abline(8.34, 4.37, lty=2)
abline(lm(lbhm ~ lvd), lty=3)
legend(-0.4, 10, legend=c("Elicited", "ME Reference", "Least Squares"), lty=1:3, cex=0.8)
```



Exercise Compare the reference prior and measurement error models in terms of their estimated fits and using their posterior predictive distributions.

Exercise For the M-sigma data fit the measurement error model assuming no error in the response. Summarize the impact on posterior inference and posterior predictive inference.