RStan Codes

Cristian Villegas

29/08/19

Contents

1 Normal distribution without covariates in Stan

$Y \sim N(\mu, \sigma^2)$. Therefore, $\theta = (\mu, \sigma^2)^\top$. Here, we do not specify the prior distribution for each parameter

```
rm(list = ls())
load("normal_dist_fit1.RData")
#ajuste<- lm(stack.loss~1, data=stackloss)
```

1.1 Classic inference: MLE for μ

```
summary(ajuste)$coef[,1]#mean(stackloss$stack.loss)
```

```
## [1] 17.52381
```

1.2 Classic inference: MLE for σ

```
summary(ajuste)$sigma#sd(stackloss$stack.loss)
```

```
## [1] 10.17162
```

1.3 Bayesian Inference: Stan code (default prior)

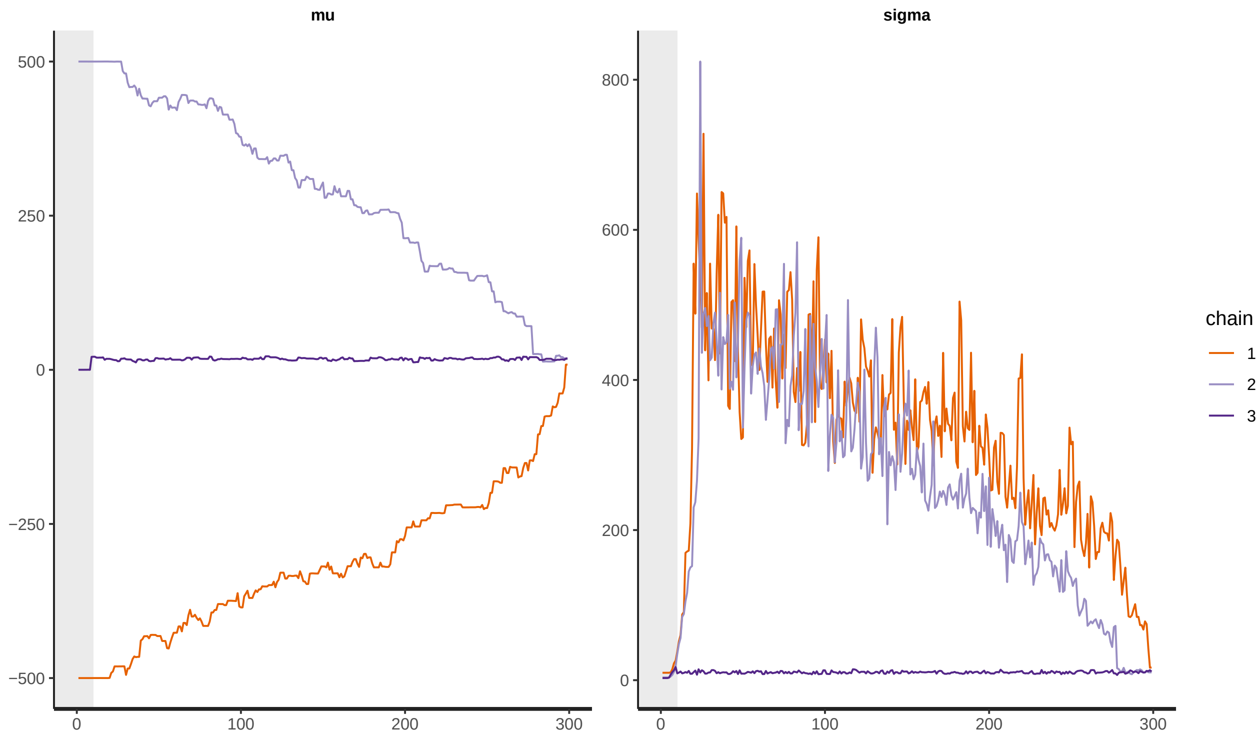
```
normal_dist_example<- '
data {
  int<lower=0> N;
  vector[N] y;
}
parameters {
//defines the sampling space
  real mu;
  real<lower=0> sigma;
}
model {
//defines the log-likelihood
  y ~ normal(mu, sigma);
}
'
normal_dist_fit <- stan(model_code = normal_dist_example,
init=list(list(mu=-500,sigma=3),
          list(mu=500,sigma=2),
          list(mu=0,sigma=1)),
data = list(N = dim(stackloss)[1],
            y = stackloss$stack.loss),
            chain = 3,
            iter = 300,#11000
            warmup = 10,#1000
            thin = 1,#10
            refresh = 0,seed=123)#refresh=-1

#save.image(file = "normal_dist_fit.RData")
```

1.4 MCMC diagnostics using the bayesplot package

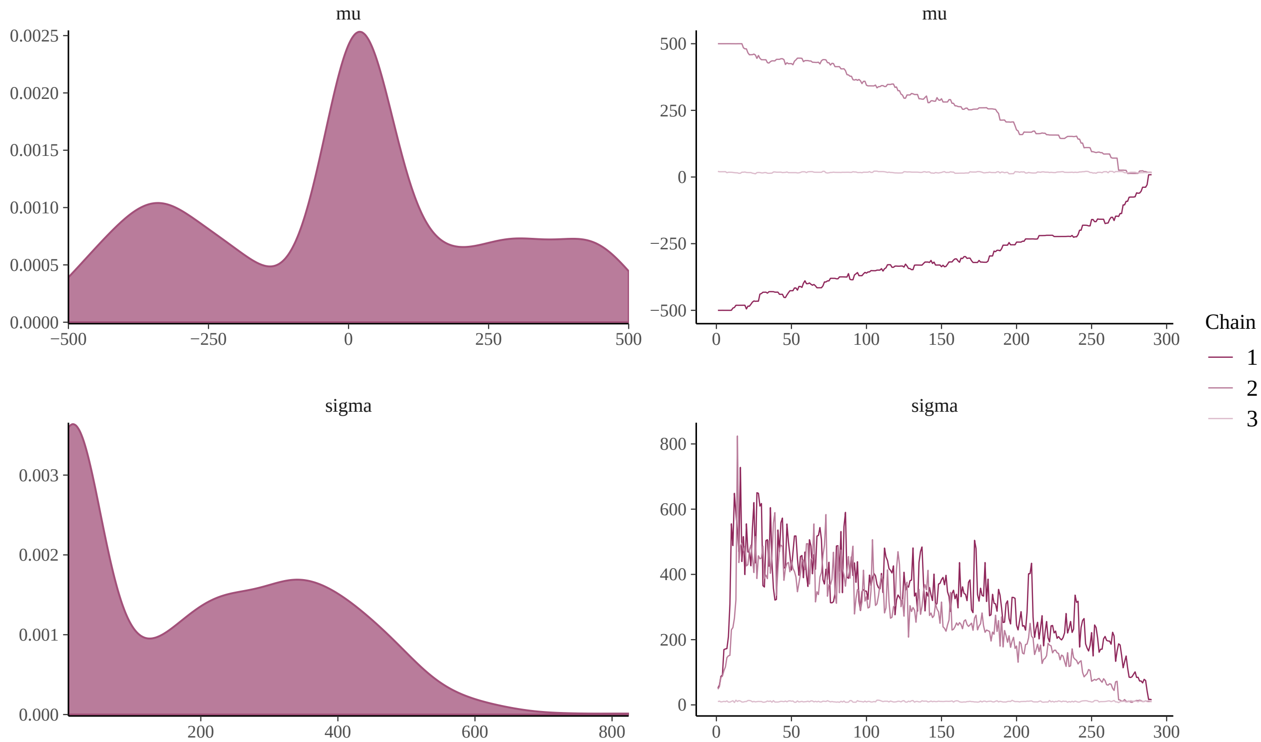
```
parametros<- c("mu", "sigma")

CI_theta <- summary(normal_dist_fit,
                    pars = parametros,
                    probs = c(0.025, 0.975))$summary
print(CI_theta)
```

```
##              mean   se_mean       sd       2.5%     97.5%    n_eff     Rhat
## mu      -3.494559 193.1230 263.4258 -478.22140 459.4751 1.860580 4.707568
## sigma 201.145878 108.8411 179.5661    8.46037 542.6115 2.721845 2.298128
```

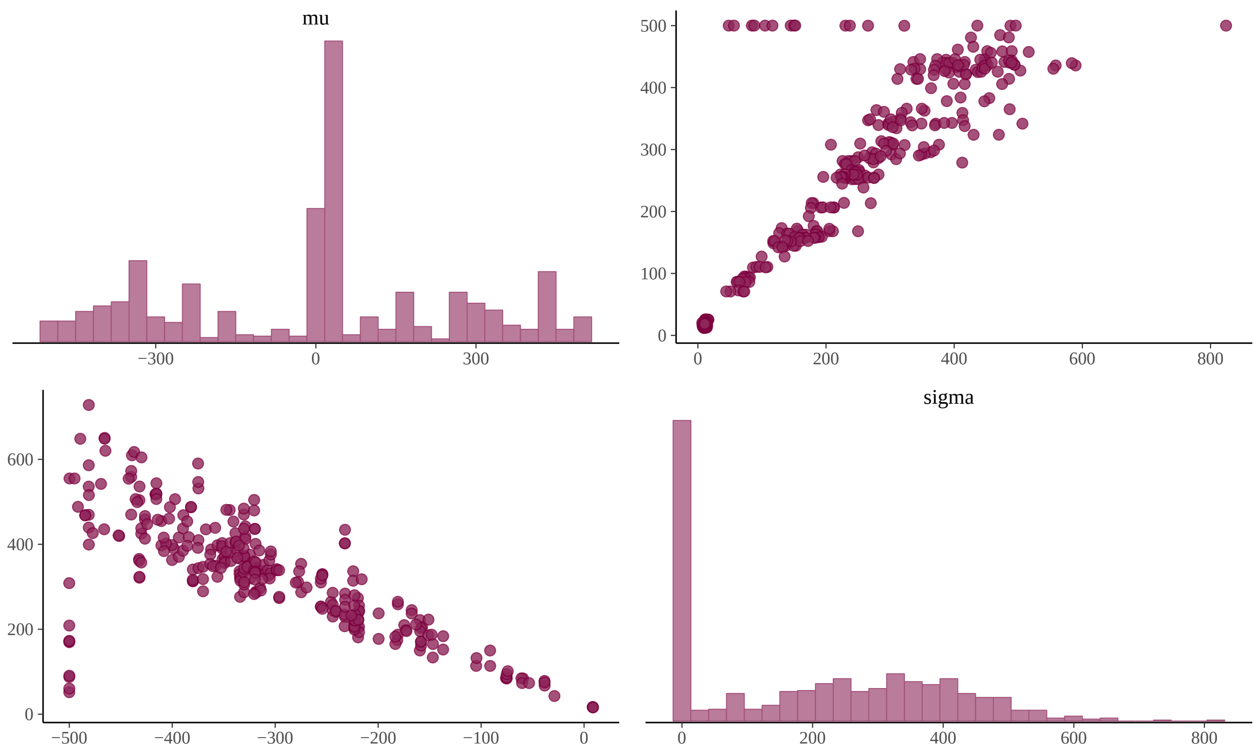
```
mcmc_cadeia <- as.array(normal_dist_fit)

traceplot(normal_dist_fit, pars = parametros, inc_warmup = TRUE)
```




```
color_scheme_set("pink")
mcmc_combo(mcmc_cadeia,pars = parametros,n_warmup=0)
```




```r
mcmc_pairs(mcmc_cadeia,pars = parametros)
```



2 Normal distribution without covariates in Stan

$Y \sim N(\mu, \sigma^2)$. Therefore, $\theta = (\mu, \sigma^2)^\top$. Here, we specify the prior distribution for each parameter.

```r
rm(list = ls())
load("normal_dist_fit2.RData")
#ajuste<- lm(stack.loss~1, data=stackloss)
```

2.1 Classic inference: MLE for μ

```r
summary(ajuste)$coef[,1]#mean(stackloss$stack.loss)
```

```
## [1] 17.52381
```

2.2 Classic inference: MLE for σ

```r
summary(ajuste)$sigma#sd(stackloss$stack.loss)
```

```
## [1] 10.17162
```

2.3 Bayesian Inference: Stan code (specific prior)

```r
normal_dist_example<- '
data {
  int<lower=0> N;
  vector[N] y;
}
parameters {
  real mu;
  real<lower=0> sigma;
}
model {
  y ~ normal(mu, sigma); //defines the log-likelihood
  mu~ normal(0,1e6);//defines Prior for mu
  sigma ~ student_t(3,0,1);//defines Prior sigma
}
'

normal_dist_fit <- stan(model_code = normal_dist_example,
data = list(N = dim(stackloss)[1],
            y = stackloss$stack.loss),
            chain = 3,
            iter = 11000,
            warmup = 1000,
            thin = 10,
            refresh = 0)#refresh=-1
#save.image(file = "normal_dist_fit2.RData")
```

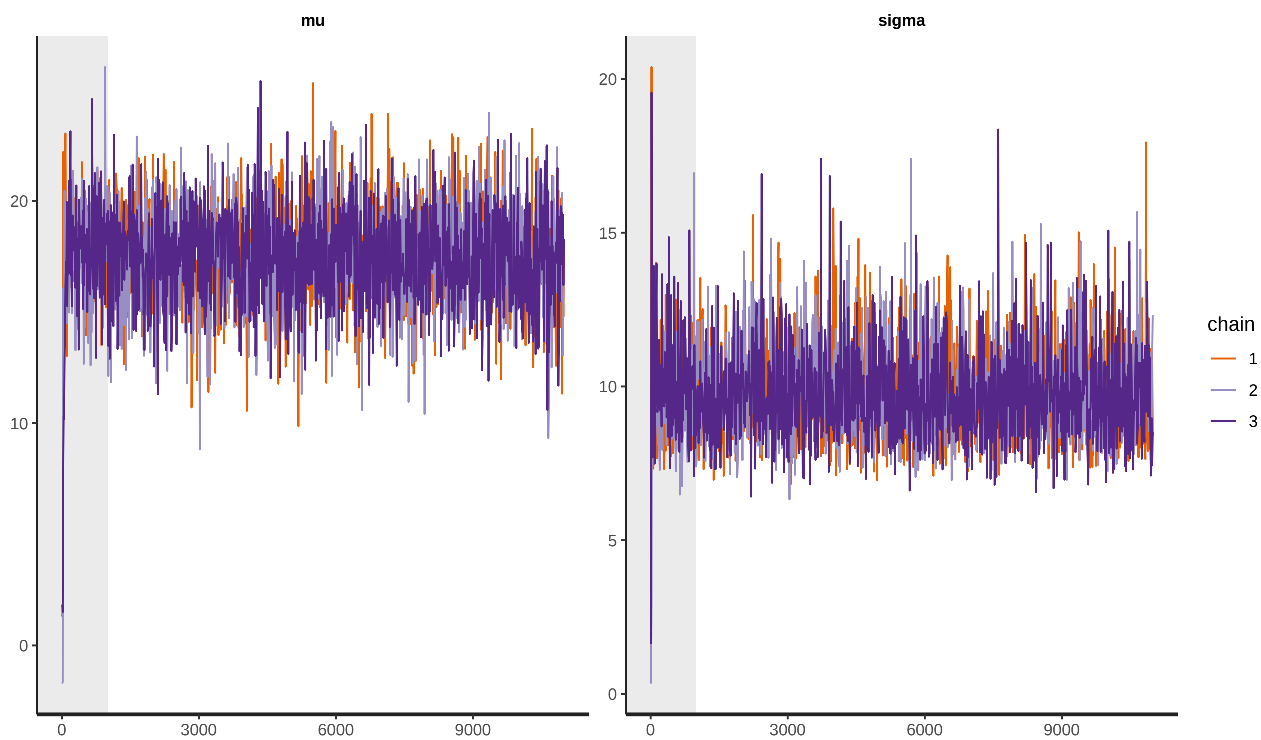
2.4 MCMC diagnostics using the bayesplot package

```r
parametros<- c("mu", "sigma")
```

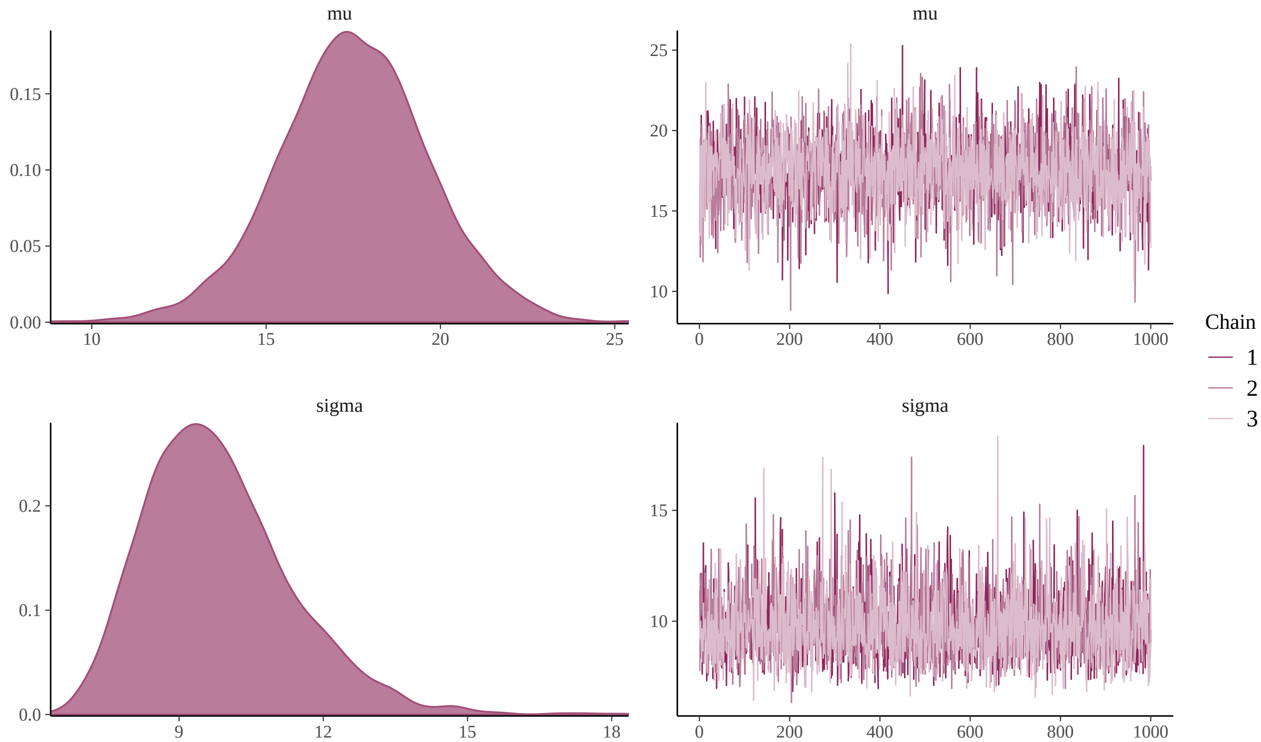
```r
CI_theta <- summary(normal_dist_fit,
                    pars = parametros,
                    probs = c(0.025, 0.975))$summary
print(CI_theta)
```

```
##            mean    se_mean       sd      2.5%    97.5%    n_eff      Rhat
## mu    17.503207 0.03946205 2.150759 13.183245 21.80289 2970.463 1.0001178
## sigma  9.821392 0.02801285 1.527582  7.404955 13.26666 2973.684 0.9997358
```

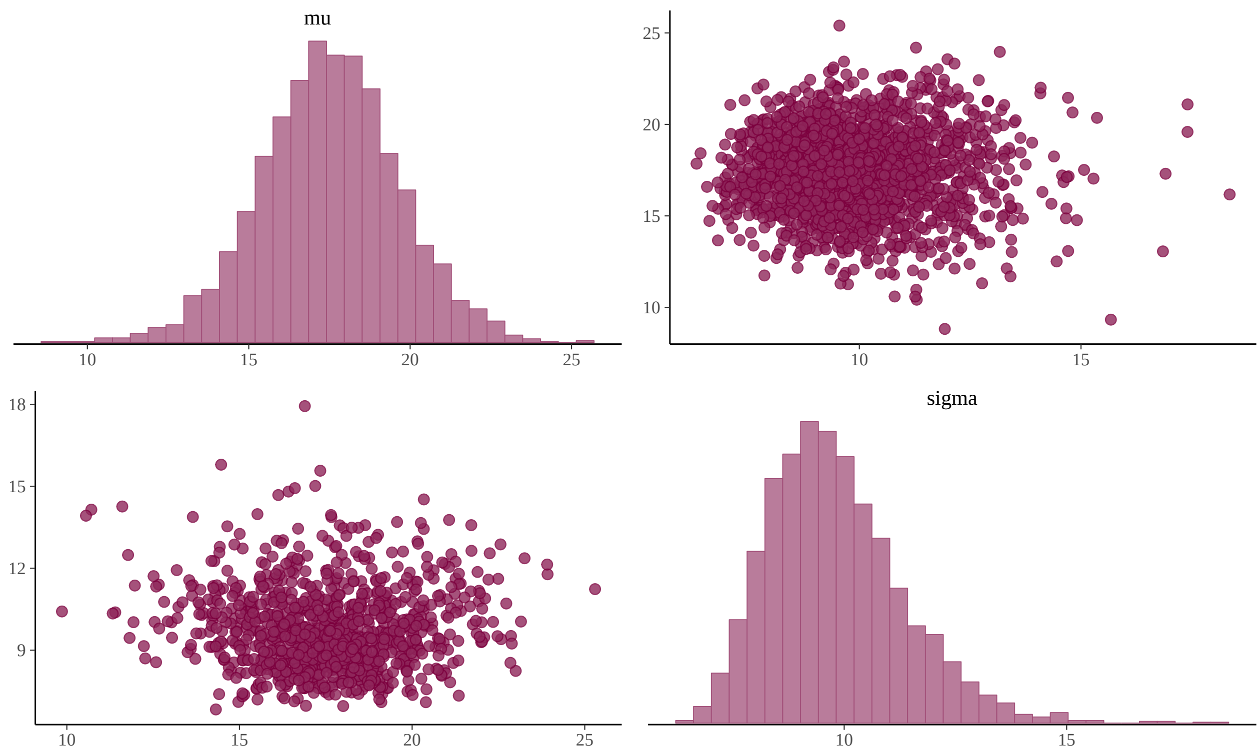
```r
mcmc_cadeia <- as.array(normal_dist_fit)

traceplot(normal_dist_fit, pars = parametros, inc_warmup = TRUE)
```




```r
color_scheme_set("pink")
mcmc_combo(mcmc_cadeia,pars = parametros,n_warmup=0)
```




```r
mcmc_pairs(mcmc_cadeia,pars = parametros)
```



3 Normal Linear Model

$$stackloss_i = \beta_1 + \beta_2 * AirFlow_i + \varepsilon_i$$

with ε_i $N(0,\sigma^2)$. Therefore $\theta = (\beta_1, \beta_2, \sigma^2)^\top$. Here, we do not specify the prior distribution for each parameter.

3.1 Classic inference: MLE for β

```
rm(list = ls())
ajuste<- lm(stack.loss~Air.Flow, data=stackloss)
load("lm_fit1.RData")
```

3.2 Classic inference: MLE for β

```
summary(ajuste)$coef[,1]
```

```
## (Intercept)    Air.Flow
##  -44.132025    1.020309
```

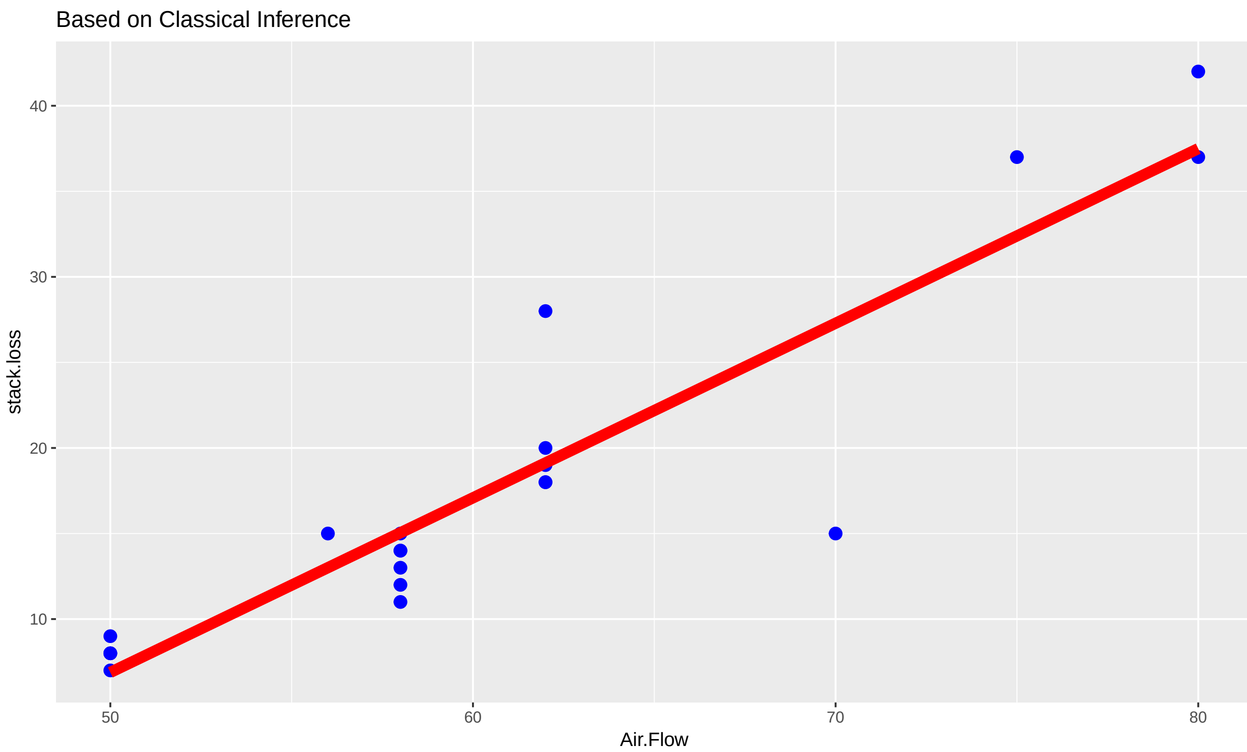
3.3 Classic inference: MLE for σ

```
summary(ajuste)$sigma#sd(stackloss$stack.loss)
```

```
## [1] 4.098242
```

3.3.1 Fitted curve

```
ggplot(stackloss, aes(Air.Flow,stack.loss))+geom_point(col="blue",size=3)+
  geom_smooth(method ="lm", col="red", se=FALSE, size=3)+
  ggtitle("Based on Classical Inference")
```



3.4 Bayesian Inference: Stan code

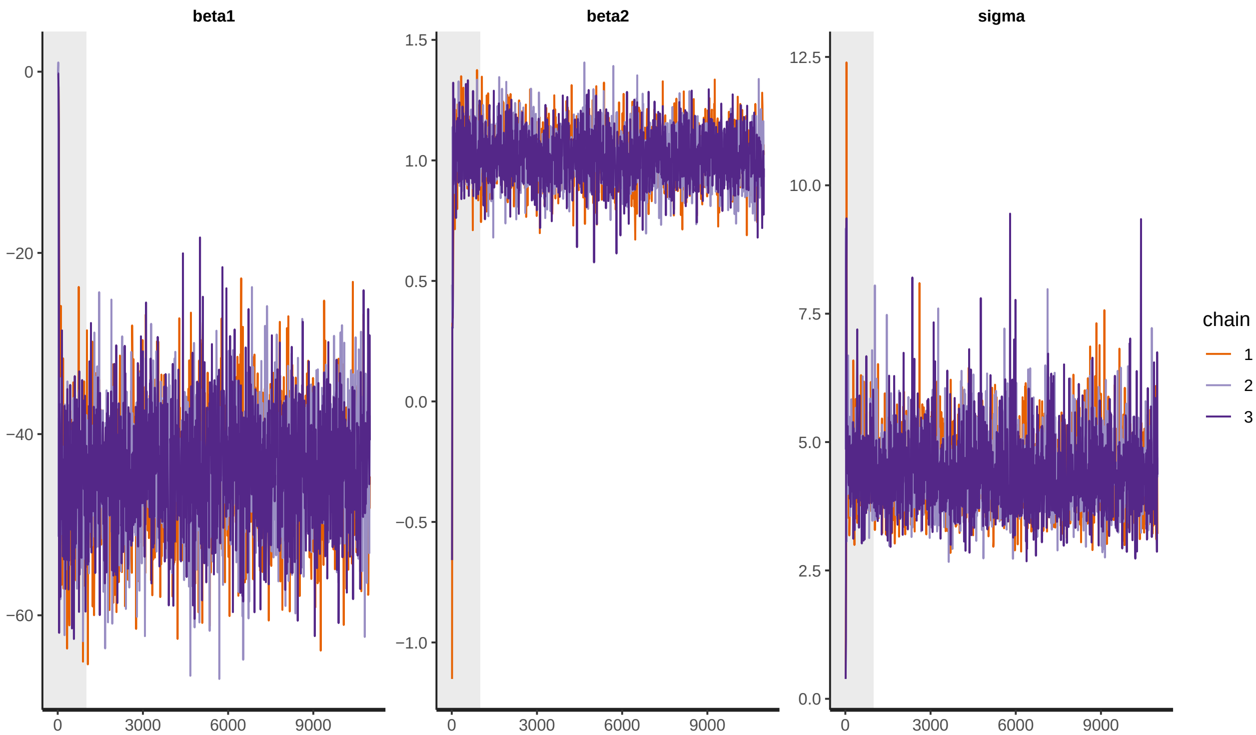
```
lm_example<- '
data {
int<lower=0> N;
vector[N] x;
vector[N] y;
}
parameters {
real beta1;
real beta2;
real<lower=0> sigma;
}
model {
y ~ normal(beta1+beta2*x, sigma);
}
'
lm_fit <- stan(model_code = lm_example,
data = list(N = dim(stackloss)[1],
            x = stackloss$Air.Flow,
y = stackloss$stack.loss),
                chain = 3,
                iter = 11000,
                warmup = 1000,
                thin = 10,
                refresh=0)
#save.image(file = "lm_fit1.RData")
```

3.5 MCMC diagnostics using the bayesplot package

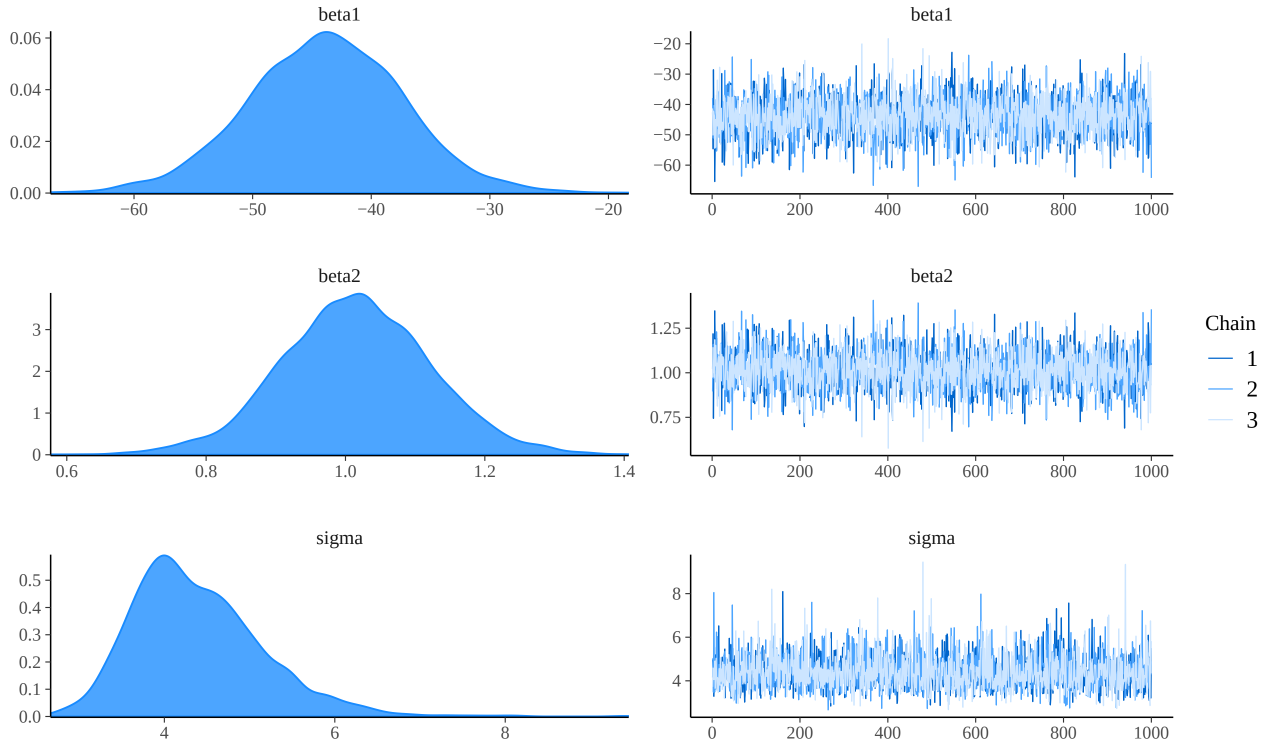
```
parametros<- c(paste('beta',1:2, sep=""), 'sigma')

CI_theta <- summary(lm_fit,
                       pars = parametros,
                       probs = c(0.025, 0.975))$summary
print(CI_theta)
```

```
##              mean     se_mean        sd        2.5%      97.5%    n_eff
## beta1 -43.847882 0.123731167 6.5662771 -56.8302195 -30.913163 2816.308
## beta2   1.015100 0.002019555 0.1074735   0.7983867   1.229148 2831.987
## sigma   4.393619 0.014107408 0.7696754   3.1979503   6.144547 2976.602
##           Rhat
## beta1 1.000541
## beta2 1.000703
## sigma 1.000270
```

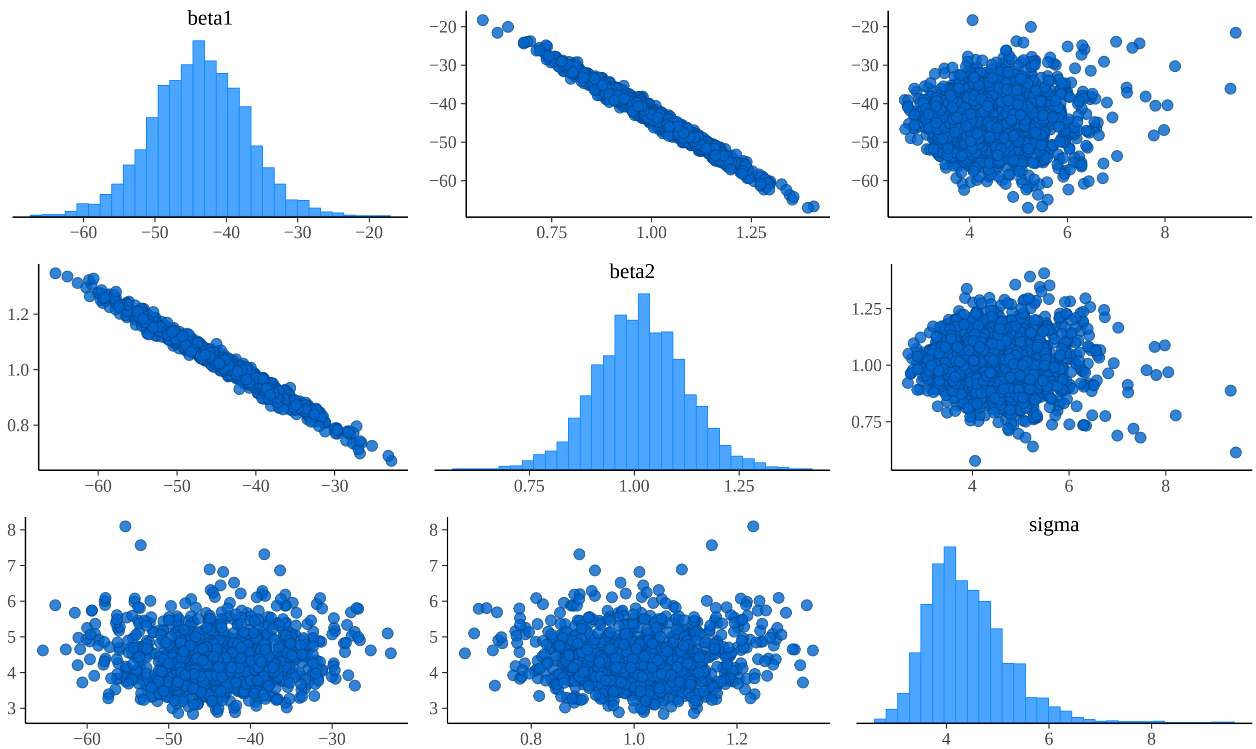
```
mcmc_cadeia <- as.array(lm_fit)

traceplot(lm_fit, pars = parametros, inc_warmup = TRUE)
```




```
color_scheme_set("brightblue")

mcmc_combo(mcmc_cadeia,pars = parametros,n_warmup=0)
```

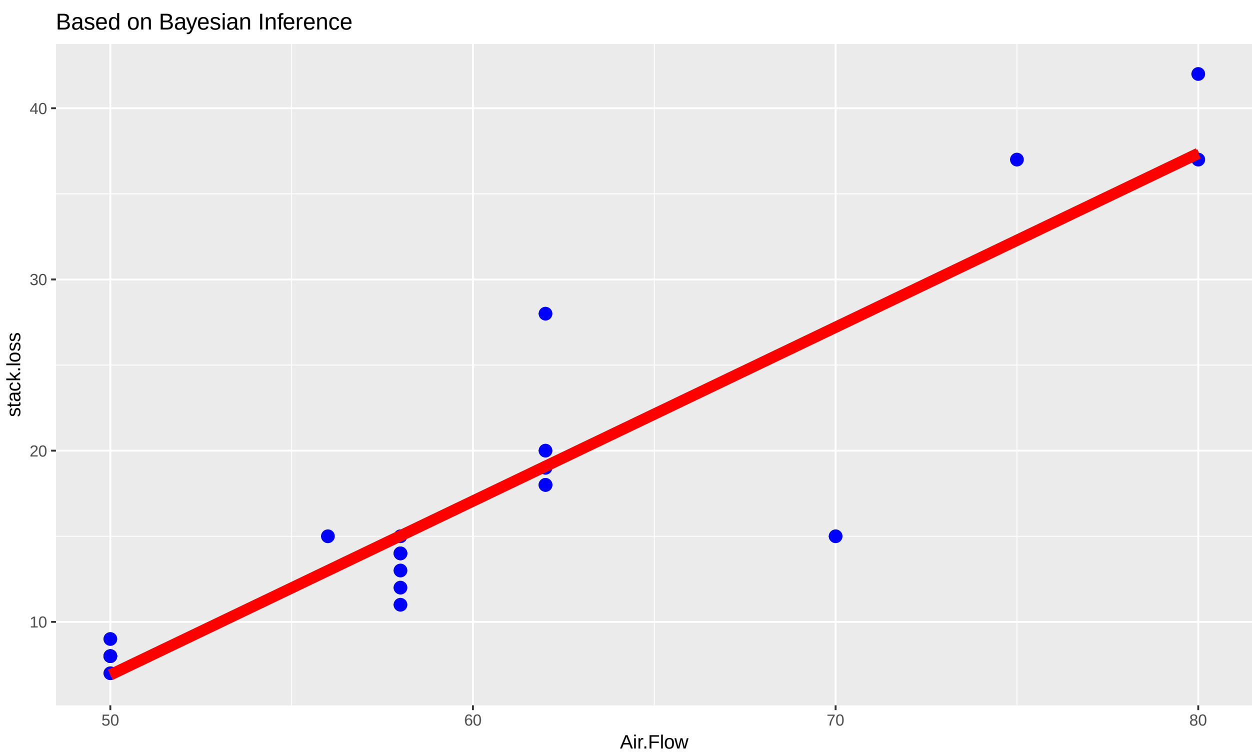



```
mcmc_pairs(mcmc_cadeia,pars = parametros)
```



3.5.1 Fitted curve

```
LM.function<- function(x,beta1, beta2){
  beta1+beta2*x
}

parametros<- c(paste('beta',1:2, sep=""), 'sigma')

CI_theta <- summary(lm_fit,
                      pars = parametros,
                      probs = c(0.025, 0.975))$summary

ggplot(stackloss, aes(Air.Flow,stack.loss))+geom_point(col="blue",size=3)+
  geom_smooth(method ="lm", col="red", se=FALSE)+
  ggtitle("Based on Bayesian Inference")+
stat_function(data=stackloss,aes(x=Air.Flow),fun = LM.function,
args = list(beta1=CI_theta[1,1],
            beta2=CI_theta[2,1]),
col="red", size=3)
```



4 Logistic Regression

Binomial response with logit link function. Here, we do not specify the prior distribution for each parameter.

```
rm(list = ls())
load("logistic_fit1.RData")
```

4.1 Classic inference for β

```
fitLogistic$coefficients
```

```
## (Intercept)       lDose
##   -60.74013    34.28593
```

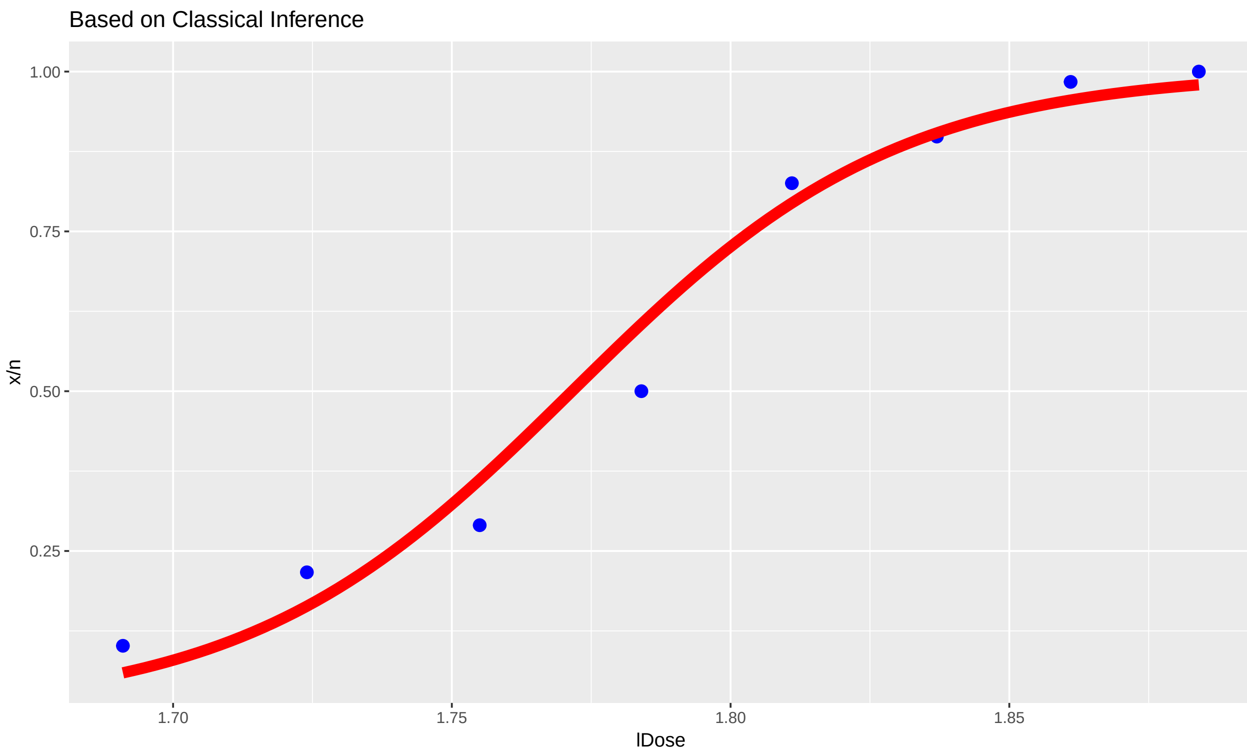
4.2 Predicted values for probabilities

```
fitLogistic$fitted.values
```

```
##          1          2          3          4          5          6
## 0.05937747 0.16366723 0.36162283 0.60490961 0.79440490 0.90405532
##          7          8
## 0.95546748 0.97925643
```

4.3 Fitted curve based on Classical Inference

```
Logit_prob<-function(x,beta1, beta2){
  eta<- beta1+beta2*x
  exp(eta)/(1+exp(eta))
}

ggplot(beetleDat, aes(lDose,x/n))+geom_point(col="blue",size=3)+
stat_function(data=beetleDat,aes(x=lDose),
fun = Logit_prob,
args = list(beta1=coef(fitLogistic)[1], beta2=coef(fitLogistic)[2]),
col="red", size=3)+
  ggtitle("Based on Classical Inference")
```



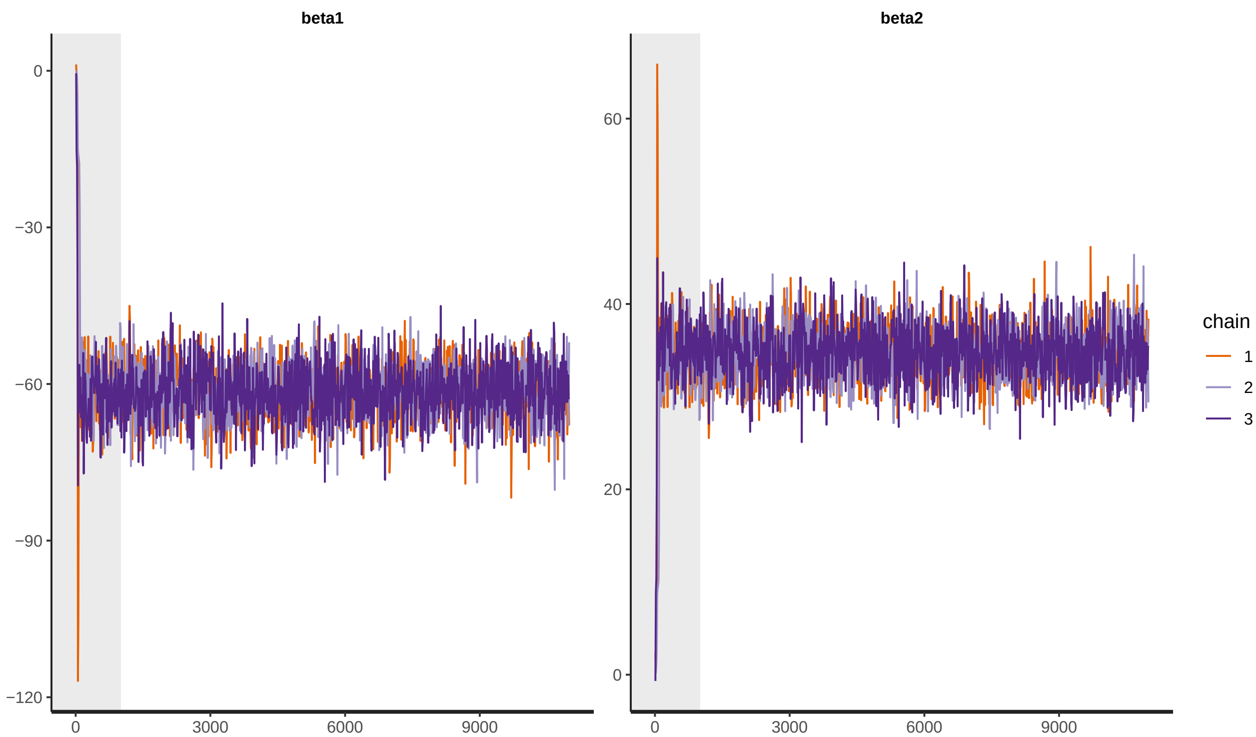
4.4 Bayesian Inference: Stan code

```
logistic_example<- '
data {
int<lower=0> N;
vector[N] x;       // lDose
int<lower=0> y[N]; // Counts
int<lower=0> n[N]; // Binomial Totals
}
parameters {
  real beta1;
  real beta2;
}
transformed parameters  {
// Probability trasformation from linear predictor
real exp_eta[N];
real<lower=0, upper=1> prob[N];
for (i in 1:N) {
exp_eta[i] = exp(beta1 + beta2*x[i]);
prob[i]= exp_eta[i]/(exp_eta[i] + 1);
}
}
model {
  y ~ binomial_logit(n, beta1 + beta2 * x);
}
'
```

```r
logistic_fit <- stan(model_code = logistic_example,
data = list(N = dim(beetleDat)[1],
            n = beetleDat$n,
            x = beetleDat$lDose,
            y = beetleDat$x),
                chain = 3,
                iter = 11000,
                warmup = 1000,
                thin = 10,
                refresh=0)

#save.image(file = "logistic_fit1.RData")
```

4.5 MCMC diagnostics using the bayesplot package

```r
parametros<- c(paste('beta',1:2, sep=""))
traceplot(logistic_fit, pars = parametros, inc_warmup = TRUE)
```

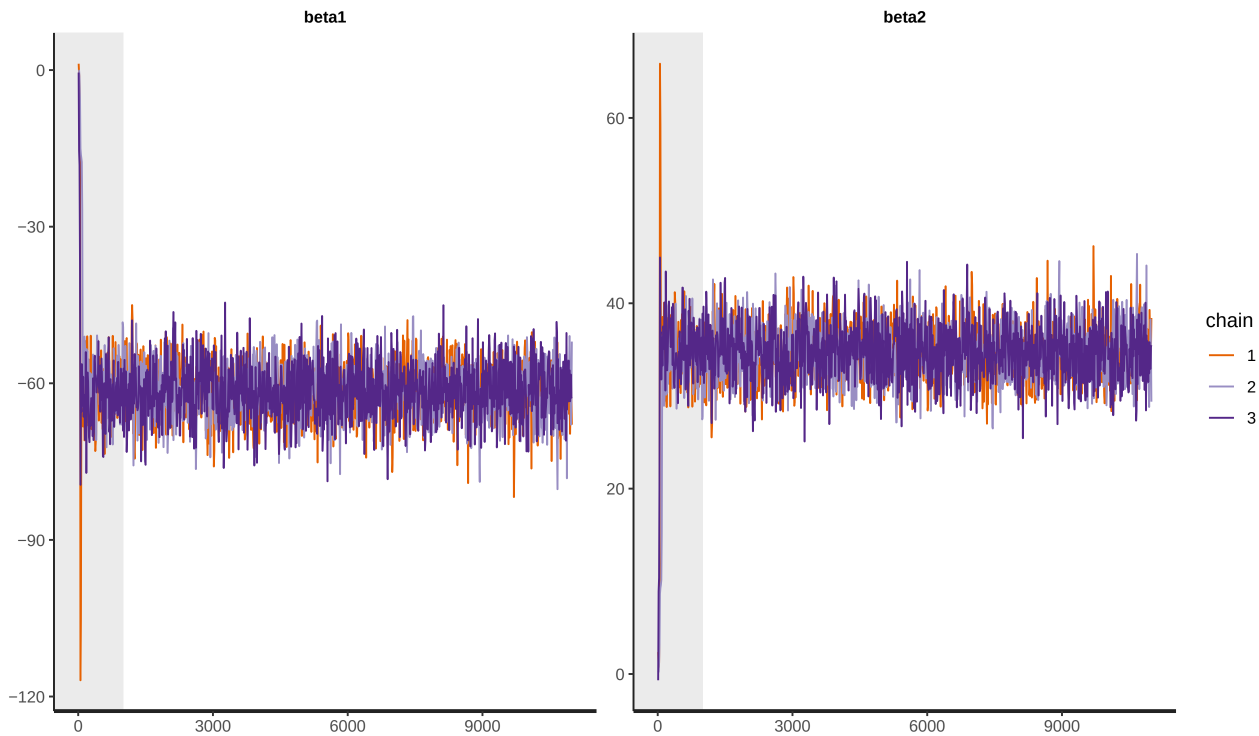



```r
CI_theta <- summary(logistic_fit,
                    pars = parametros,
                    probs = c(0.025, 0.975))$summary
print(CI_theta)
```

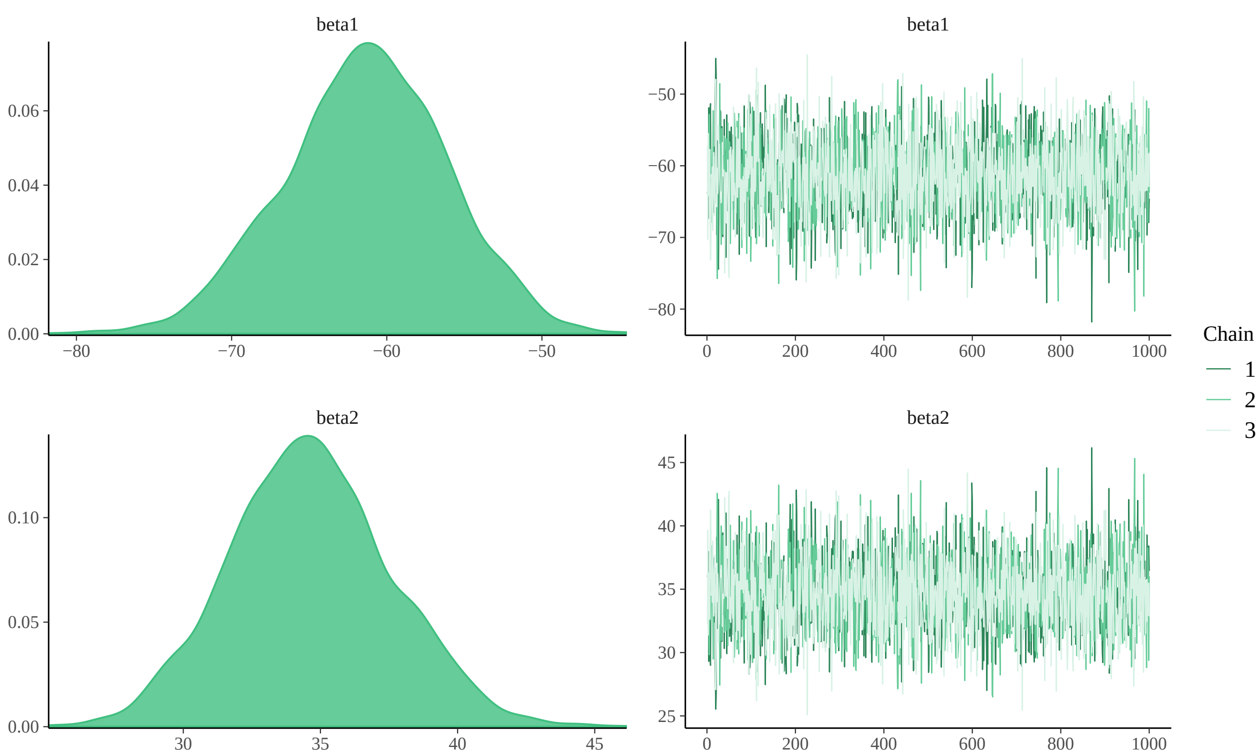
```
##            mean    se_mean       sd      2.5%     97.5%    n_eff      Rhat
## beta1 -61.37130 0.1066553 5.247504 -72.01086 -51.42912 2420.699 0.9997729
## beta2  34.64121 0.0599260 2.949725  29.07175  40.58387 2422.883 0.9997907
```

```r
mcmc_cadeia <- as.array(logistic_fit)
```

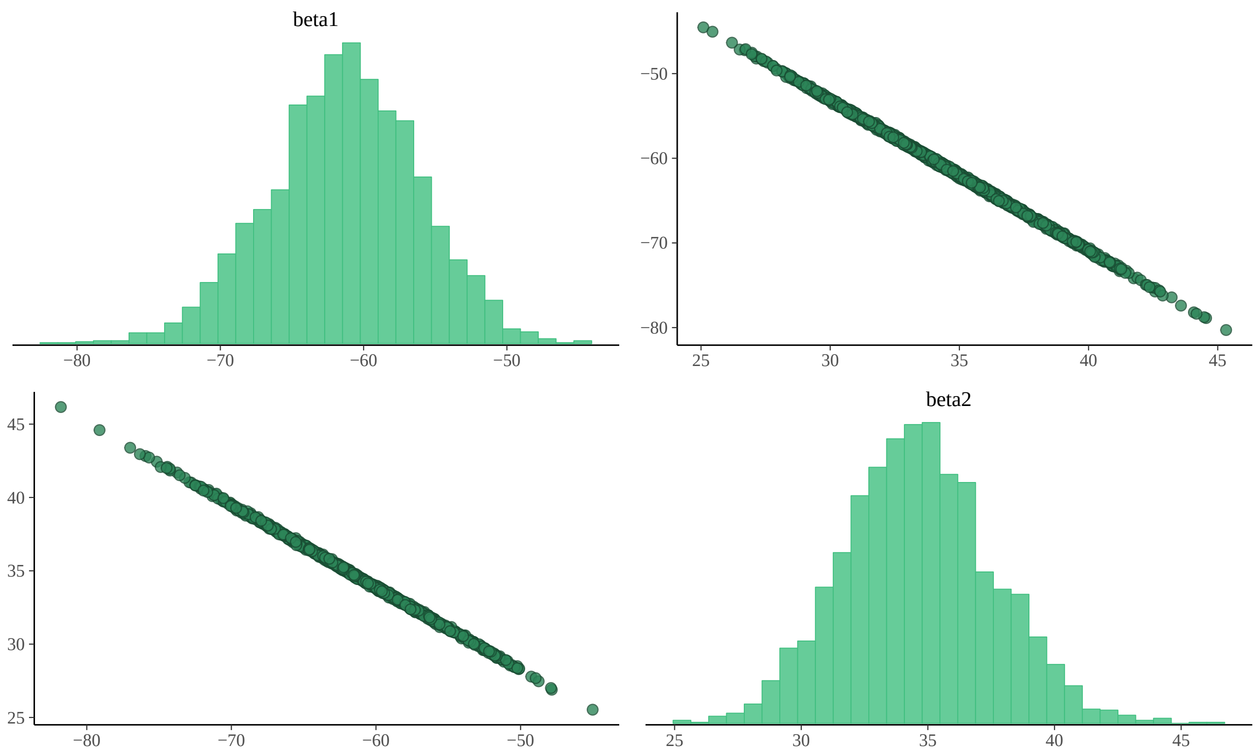
```r
traceplot(logistic_fit, pars = parametros, inc_warmup = TRUE)
```




```r
color_scheme_set("green")

mcmc_combo(mcmc_cadeia,pars = parametros,n_warmup=0)
```



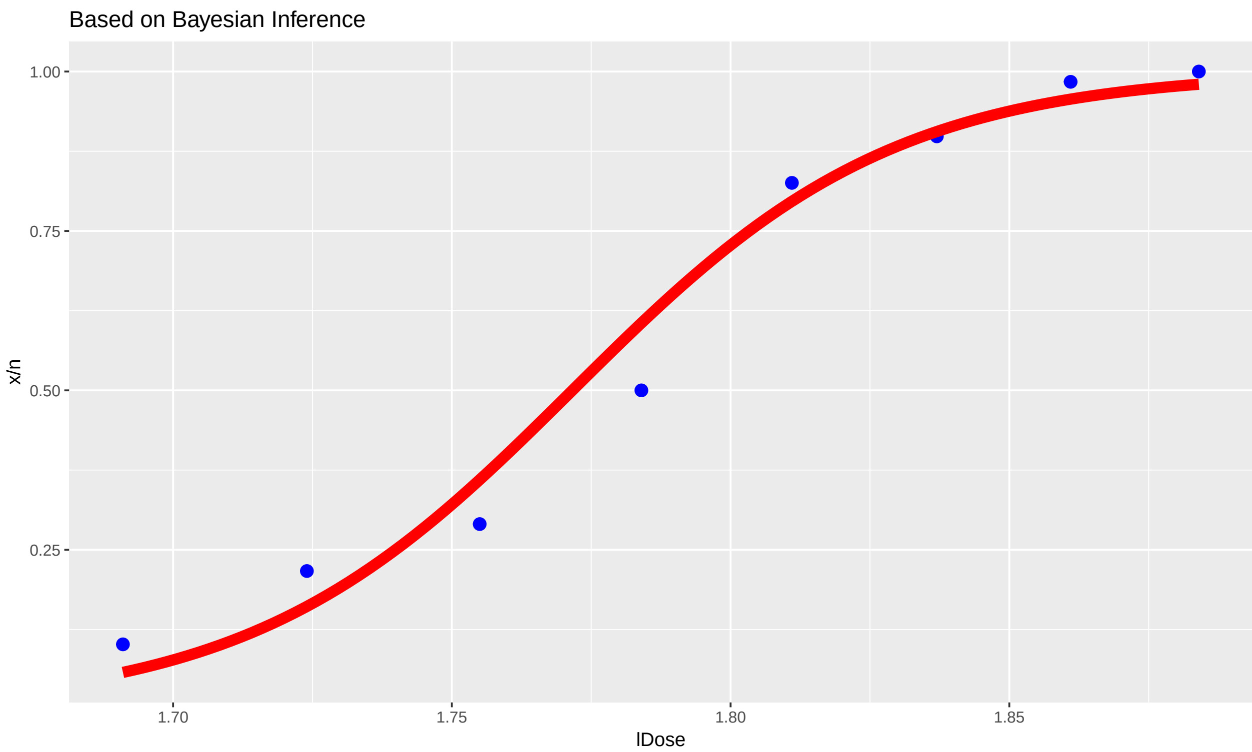
```
mcmc_pairs(mcmc_cadeia,pars = parametros)
```




```
#https://www.youtube.com/watch?v=uSjsJg8fcwY
```

4.6 Fitted curve based on Bayesian Inference

```
Logit_prob<-function(x,beta1, beta2){
  eta<- beta1+beta2*x
  exp(eta)/(1+exp(eta))
}

parametros<- paste('beta',1:2, sep="")
CI_theta <- summary(logistic_fit,
                    pars = parametros,
                    probs = c(0.025, 0.975))$summary

ggplot(beetleDat, aes(lDose,x/n))+geom_point(col="blue",size=3)+
  ggtitle("Based on Bayesian Inference")+
  stat_function(data=beetleDat,aes(x=lDose),
              fun = Logit_prob,
args = list(beta1=CI_theta[1,1], beta2=CI_theta[2,1]),
col="red", size=3)
```



5 Von Bertanlanffy

```
rm(list=ls())
load("modelo_Von_Bertanlanffy.RData")
#set.seed(123)
#dados.amostra<- dados[sample(x=dim(dados)[1], size=1000),]
```

5.1 Classic inference for β

```
LVB<-function(x,beta1, beta2, beta3){
   beta1*(1-exp(-beta2*(x-beta3)))
}

ajuste_LVB <- nls(Length ~ LVB(Age, beta1, beta2, beta3),
start = list(beta1 = max(dados.amostra$Length),
             beta2 = 0.5,
             beta3= 0),
data = dados.amostra)
coef(ajuste_LVB)
```

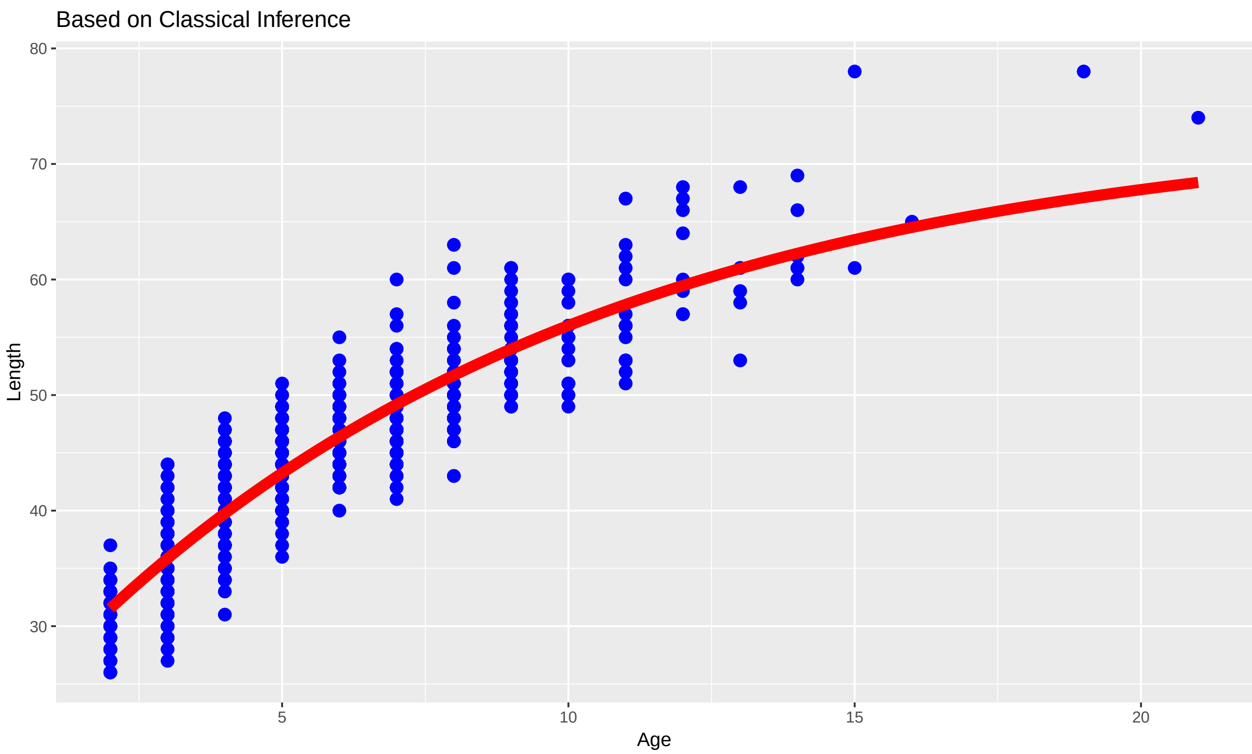
```
##       beta1       beta2       beta3
## 73.8227693  0.1080882 -3.1551711
```

```
sigma(ajuste_LVB)
```

```
## [1] 3.247594
```

5.2 Fitted curve based on Classical Inference

```
ggplot(dados.amostra, aes(Age,Length))+geom_point(col="blue",size=3)+
stat_function(data=dados.amostra,aes(x=Age),fun = LVB,
args = list(beta1=coef(ajuste_LVB)[1], beta2=coef(ajuste_LVB)[2], beta3=coef(ajuste_LVB)[3]),
col="red", size=3) +ggtitle("Based on Classical Inference")
```



5.3 Bayesian Inference

```
Von_Bertanlanffy_mcmc <- '
data {
int<lower = 0> N ;
vector[N] x ;
vector[N] y ;
}
parameters {
real<lower = .0> beta1 ;
real<lower = .0> beta2 ;
real beta3 ;
real<lower = .0> sigma ;
}

model {
y ~ normal(beta1*(1-exp(-beta2*(x-beta3))), sigma) ;
}
'

ajuste_Von_Bertanlanffy <- stan(model_code = Von_Bertanlanffy_mcmc,
```

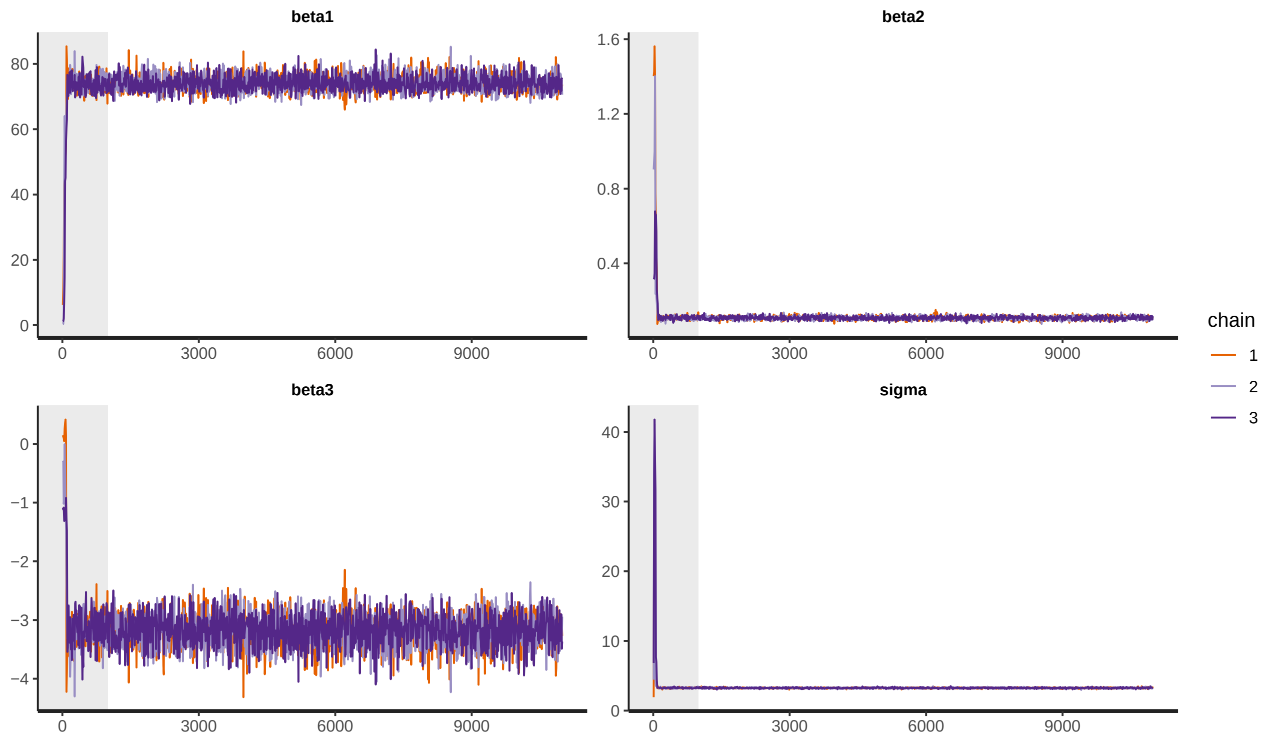
```r
data = list(N = nrow(dados.amostra),
x = dados.amostra$Age,y = dados.amostra$Length),
chain = 3,
iter = 11000,
warmup = 1000,
thin = 10,
refresh=0)

#save.image("modelo_Von_Bertanlanffy_sem_convergencia.RData")
#save.image("modelo_Von_Bertanlanffy.RData")
```

5.4 MCMC diagnostics using the `bayesplot` package

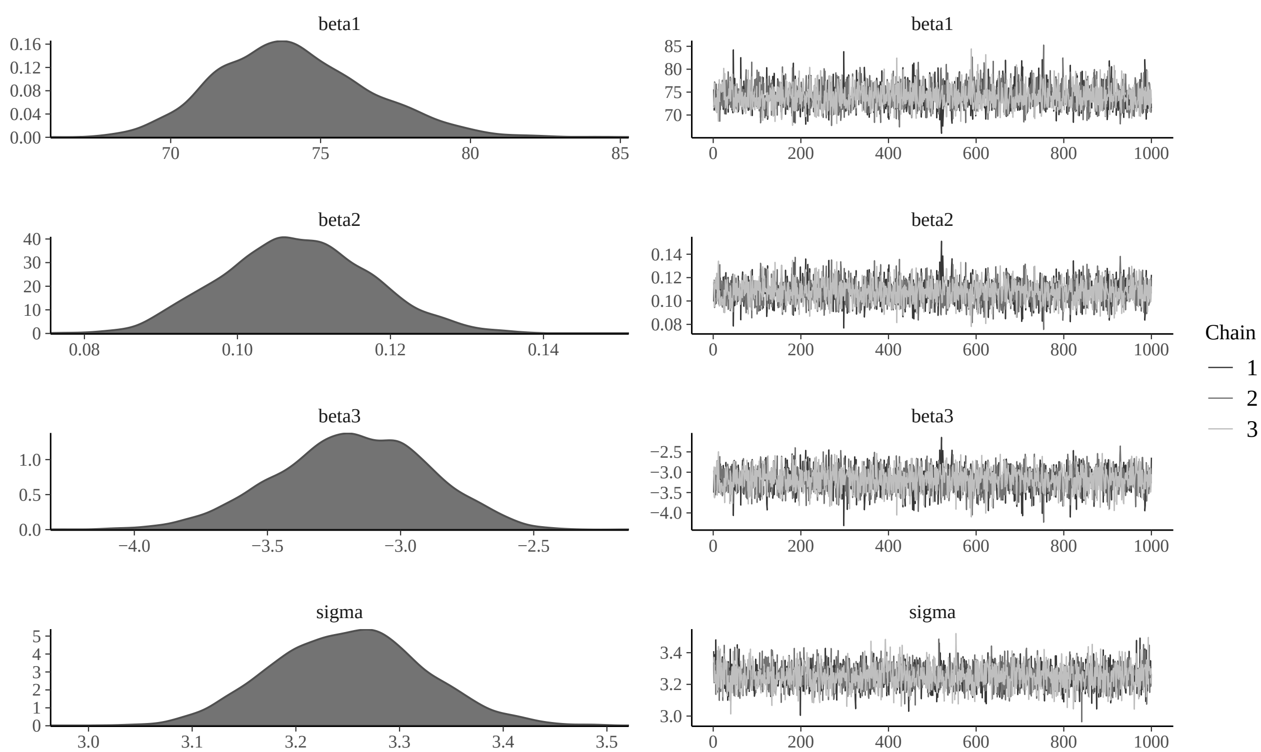
```r
parametros<- c(paste('beta',1:3, sep=""), 'sigma')

CI_theta <- summary(ajuste_Von_Bertanlanffy,
                    pars = parametros,
                    probs = c(0.025, 0.975))$summary
print(CI_theta)
```

```
##             mean      se_mean          sd        2.5%     97.5%    n_eff
## beta1 74.0883836 0.0475594521 2.538043586 69.63012718 79.494074 2847.898
## beta2  0.1077937 0.0001796031 0.009648242  0.08971907  0.127173 2885.815
## beta3 -3.1803626 0.0052332900 0.283401344 -3.75675263 -2.659727 2932.609
## sigma  3.2509435 0.0013174606 0.072781075  3.11379297  3.397478 3051.839
##            Rhat
## beta1 1.0008503
## beta2 1.0006212
## beta3 1.0005698
## sigma 0.9994522
```

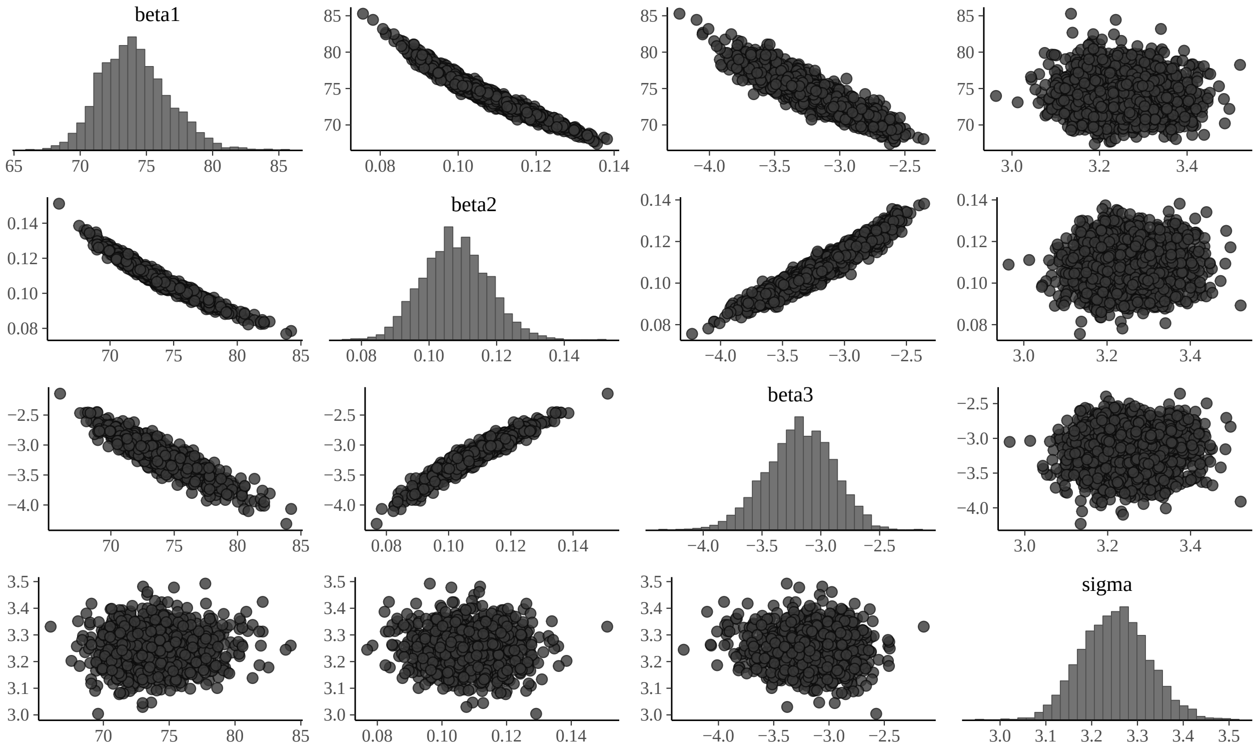
```r
mcmc_cadeia <- as.array(ajuste_Von_Bertanlanffy)

traceplot(ajuste_Von_Bertanlanffy, pars = parametros, inc_warmup = TRUE)
```




```r
color_scheme_set("darkgray")
mcmc_combo(mcmc_cadeia,pars = parametros,n_warmup=0)
```

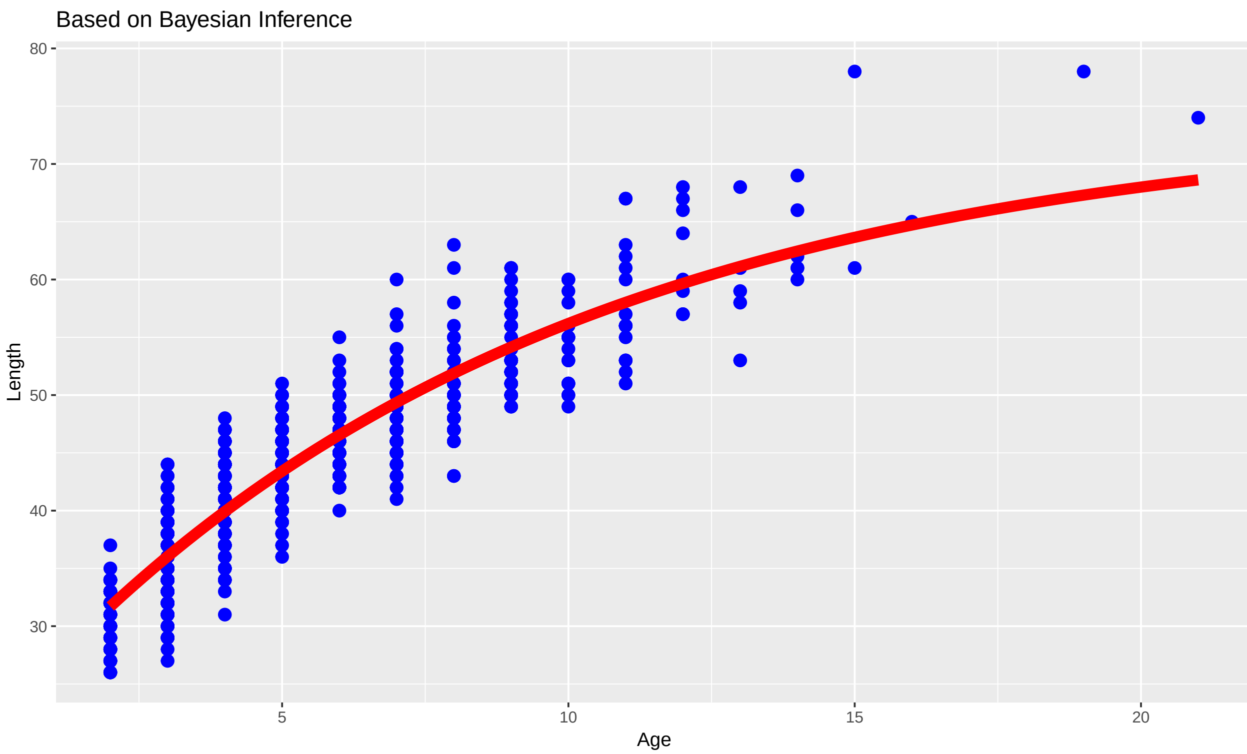



```r
mcmc_pairs(mcmc_cadeia,pars = parametros)
```



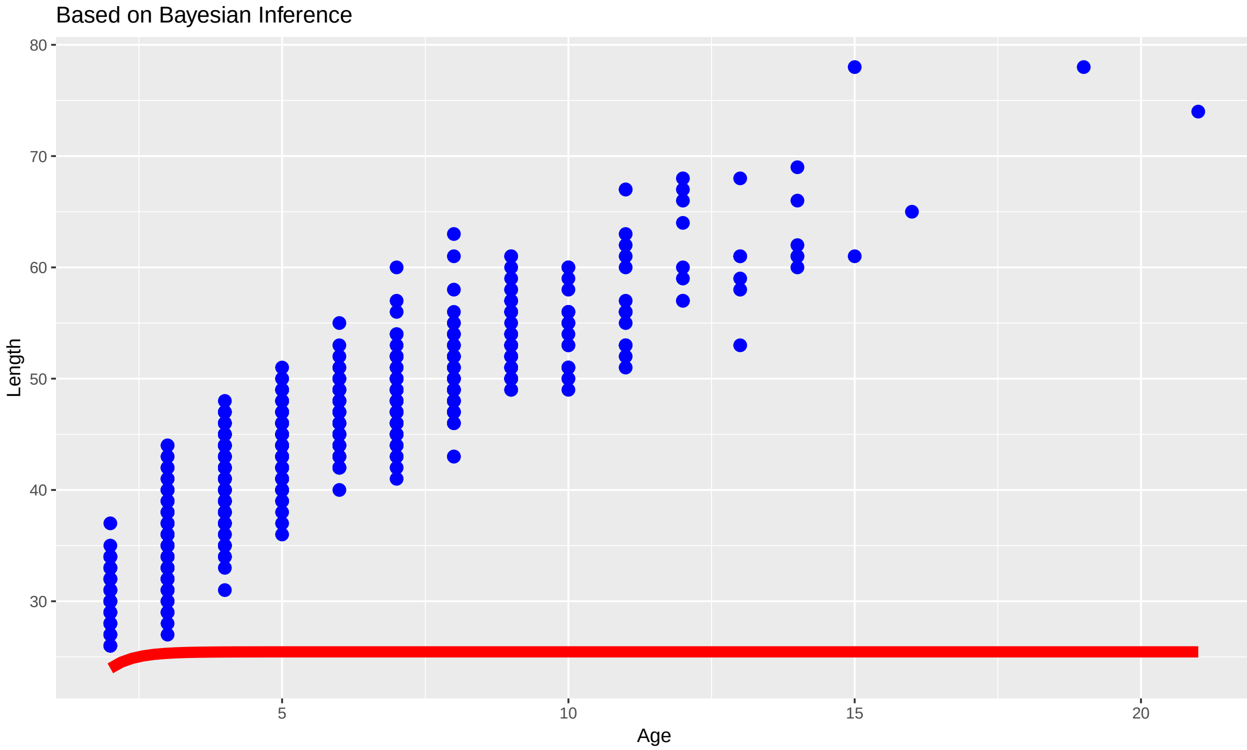
5.5 Fitted curve based on Bayesian Inference

```
ggplot(dados.amostra, aes(Age,Length))+geom_point(col="blue",size=3)+
stat_function(data=dados.amostra,aes(x=Age),fun = LVB,
args = list(beta1=CI_theta[1,1],
            beta2=CI_theta[2,1],
            beta3=CI_theta[3,1]),
col="red", size=3) +ggtitle("Based on Bayesian Inference")
```



5.6 Fitted curve based on Bayesian Inference (without convergence)

```
rm(list=ls())
load("modelo_Von_Bertanlanffy_sem_convergencia.RData")

LVB<-function(x,beta1, beta2, beta3){
   beta1*(1-exp(-beta2*(x-beta3)))
}

parametros<- c(paste('beta',1:3, sep=""), 'sigma')
CI_theta <- summary(ajuste_Von_Bertanlanffy,
                    pars = parametros,
                    probs = c(0.025, 0.975))$summary

ggplot(dados.amostra, aes(Age,Length))+geom_point(col="blue",size=3)+
stat_function(data=dados.amostra,aes(x=Age),fun = LVB,
args = list(beta1=CI_theta[1,1],
            beta2=CI_theta[2,1],
            beta3=CI_theta[3,1]),
col="red", size=3) +ggtitle("Based on Bayesian Inference")
```

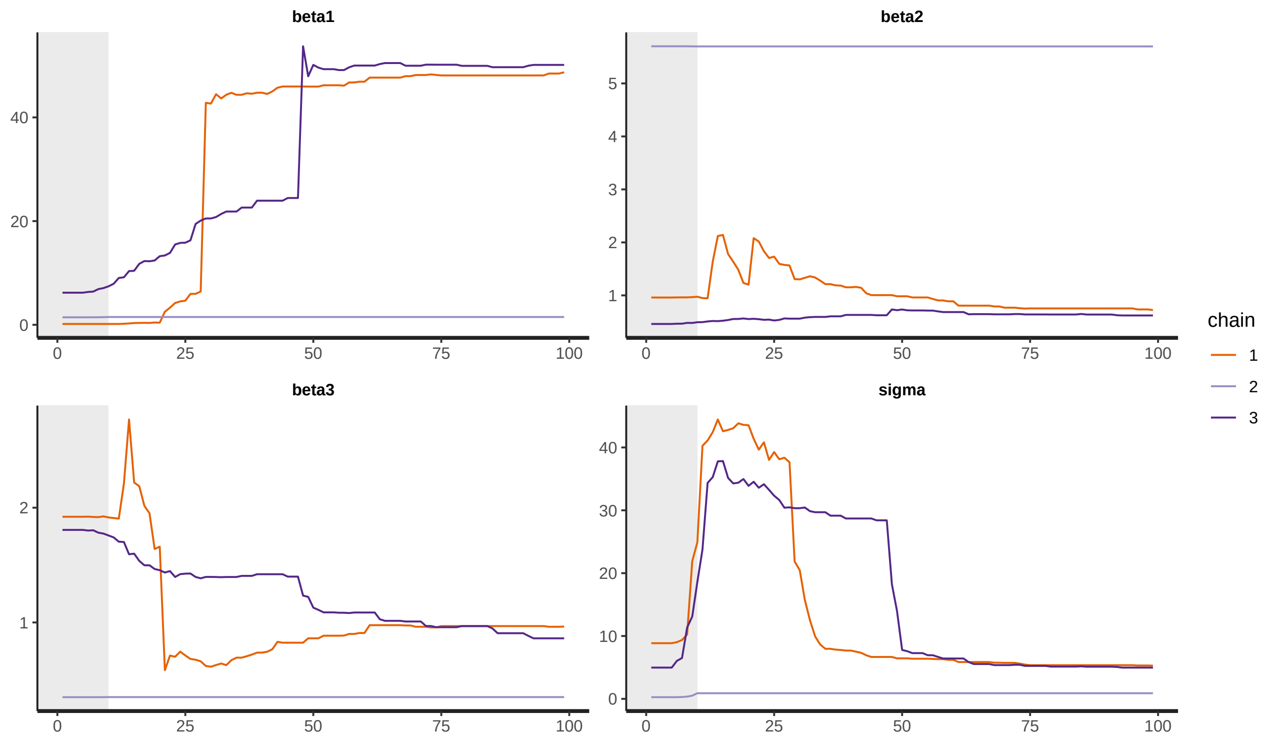


5.7 MCMC diagnostics using the bayesplot package

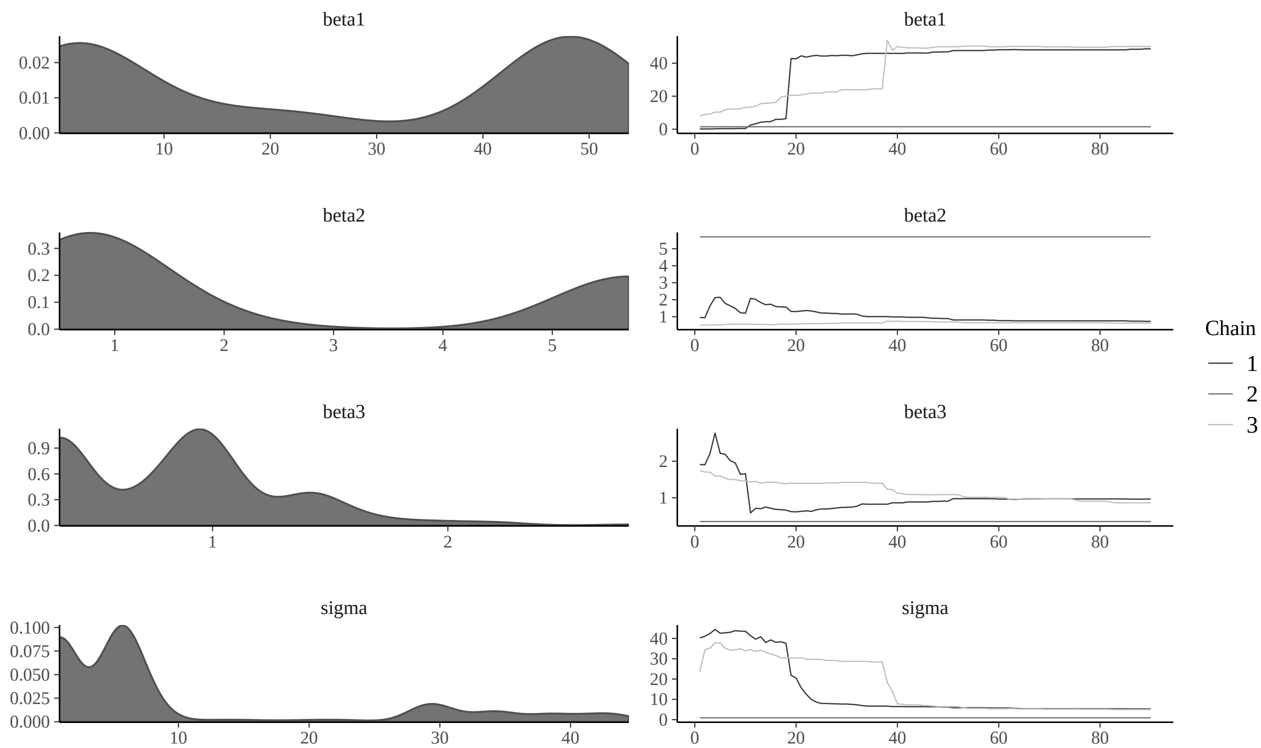
```
parametros<- c(paste('beta',1:3, sep=""), 'sigma')

CI_theta <- summary(ajuste_Von_Bertanlanffy,
                    pars = parametros,
                    probs = c(0.025, 0.975))$summary
print(CI_theta)
```

```
##             mean    se_mean         sd      2.5%     97.5%    n_eff
## beta1 25.4525584 13.4586171 21.9457933 0.3701932 50.176142 2.658898
## beta2  2.4567863  1.8754883  2.3129298 0.5367772  5.698166 1.520884
## beta3  0.8429762  0.2975077  0.4496281 0.3495844  1.905890 2.284075
## sigma 10.3197269  5.1653622 12.8660239 0.8990780 42.475302 6.204220
##            Rhat
## beta1  2.302762
## beta2 16.999568
## beta3  1.968142
## sigma  1.783695
```

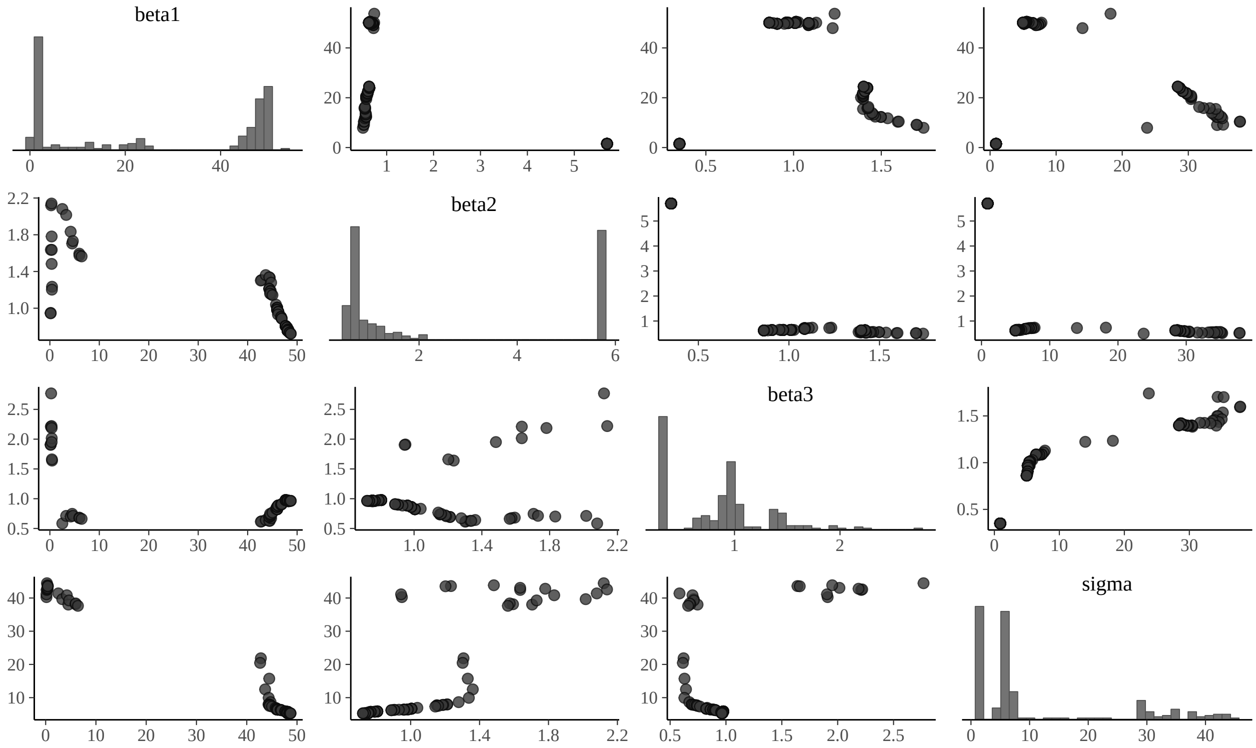
```
mcmc_cadeia <- as.array(ajuste_Von_Bertanlanffy)

traceplot(ajuste_Von_Bertanlanffy, pars = parametros, inc_warmup = TRUE)
```




```
color_scheme_set("darkgray")
mcmc_combo(mcmc_cadeia,pars = parametros,n_warmup=0)
```




```
mcmc_pairs(mcmc_cadeia,pars = parametros)
```



5.8 Normal distribution without covariates in Stan

$Y \sim N(\mu, \sigma^2)$. Therefore, $\theta = (\mu, \sigma^2)^\top$. Here, we specify the prior distribution for each parameter.

5.9 Classic inference: MLE for μ and σ

```
c(mu=coef(ajuste), sigma=sigma(ajuste))
```

```
## mu.(Intercept)          sigma
##       17.52381       10.17162
```

5.10 Bayesian Inference: Stan code (specific prior)

```
normal_dist_example<- '
data {
  int<lower=0> N;
  vector[N] y;
}
parameters {
  real mu;
  real<lower=0> sigma;
}
model {
  y ~ normal(mu, sigma);
  mu~ normal(0,1e6);
  sigma ~ student_t(3,0,1);
```

```
}
'
```

```
normal_dist_fit <- stan(model_code = normal_dist_example,
data = list(N = dim(stackloss)[1],
            y = stackloss$stack.loss),
            chain = 3,
            iter = 11000,
            warmup = 1000,
            thin = 10,
            refresh = 0)
```

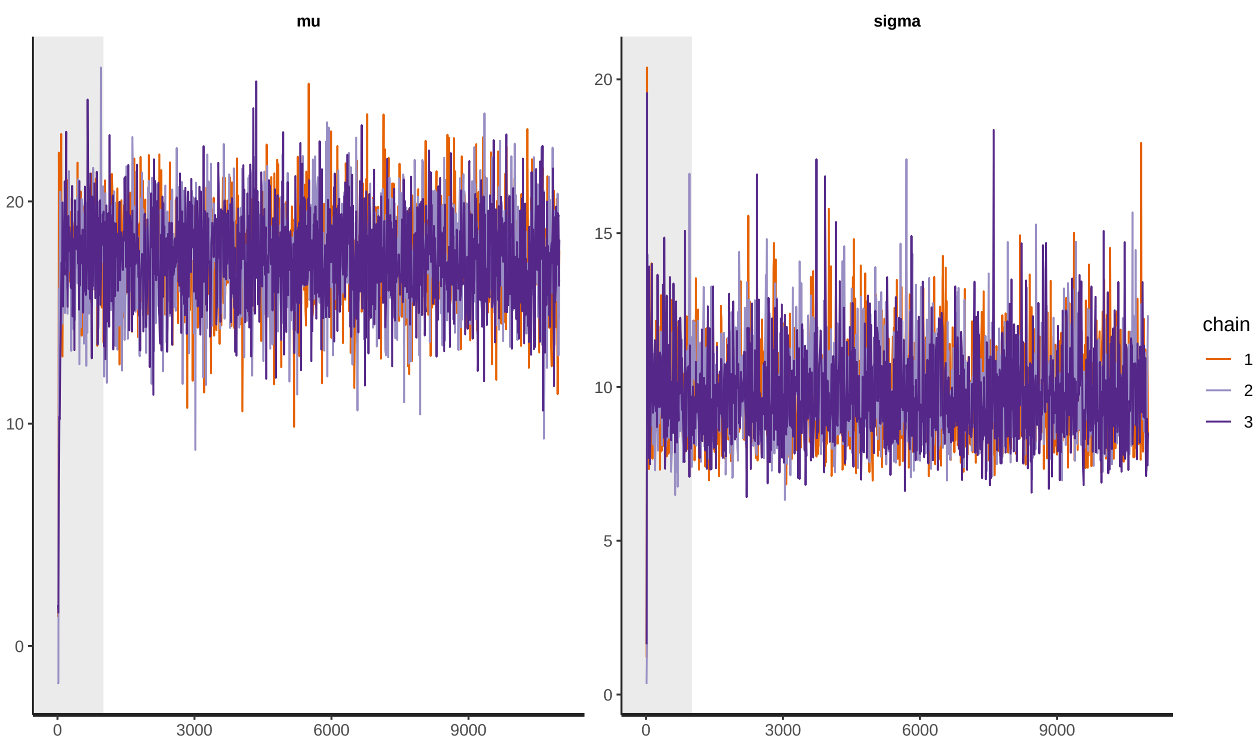
5.11 MCMC diagnostics using the `bayesplot` package

```
parametros<- c("mu", "sigma")

CI_theta <- summary(normal_dist_fit,
                    pars = parametros,
                    probs = c(0.025, 0.975))$summary
print(round(CI_theta),10)
```

```
##        mean se_mean sd 2.5% 97.5% n_eff Rhat
## mu       18       0  2   13    22  2970    1
## sigma    10       0  2    7    13  2974    1
```

5.12 MCMC diagnostics using the `bayesplot` package

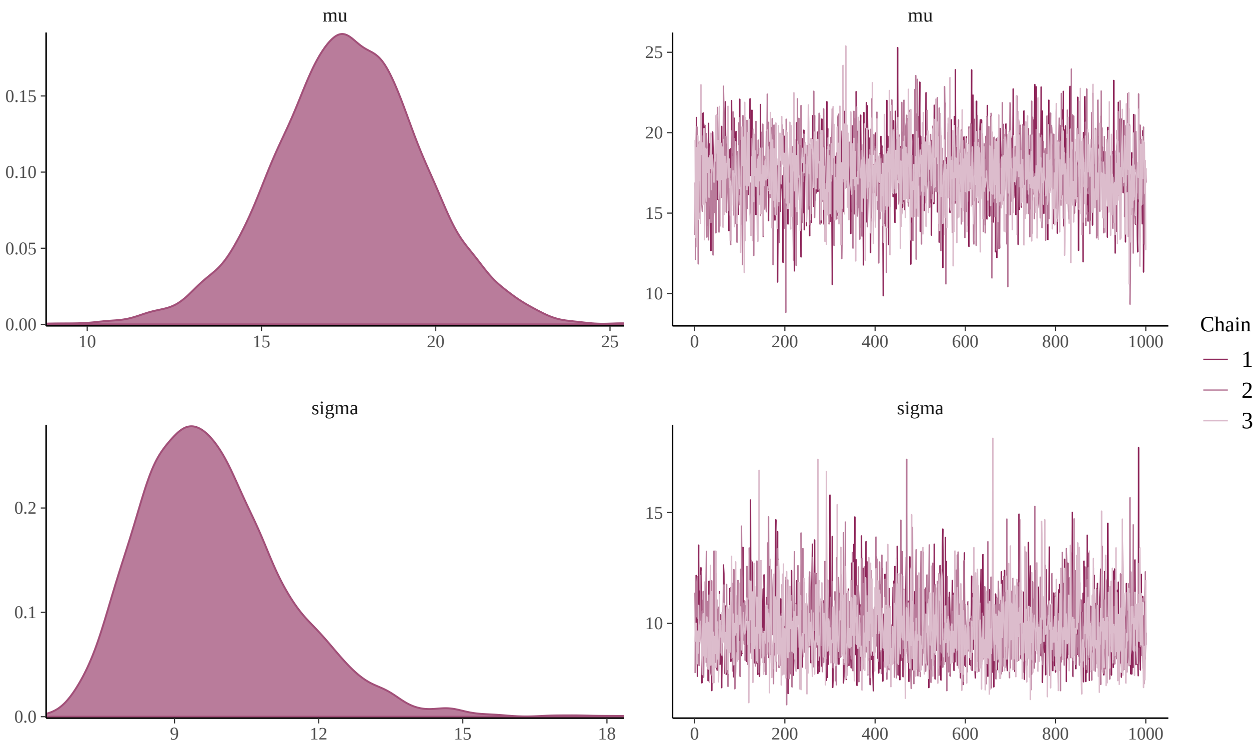
```
traceplot(normal_dist_fit, pars = parametros, inc_warmup = TRUE)
```



```r
color_scheme_set("pink")
```

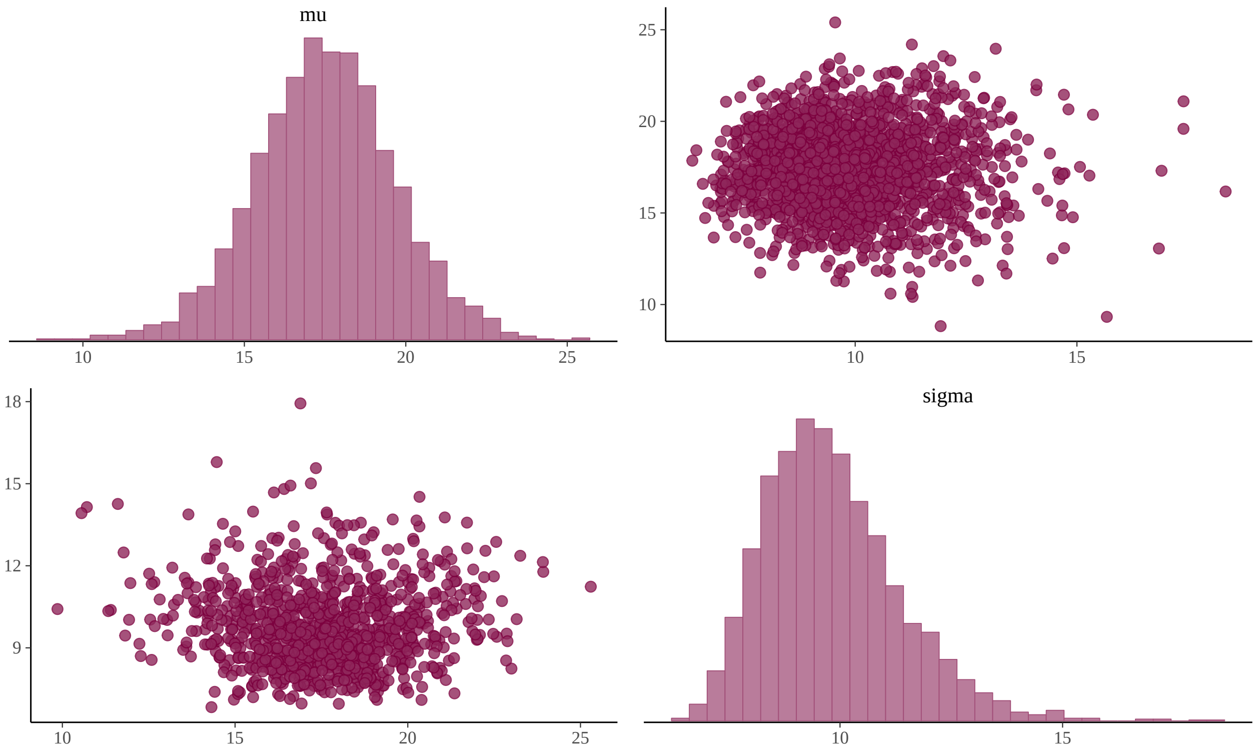
5.13 MCMC diagnostics using the `bayesplot` package

```r
mcmc_combo(mcmc_cadeia,pars = parametros,n_warmup=0)
```



5.14 MCMC diagnostics using the `bayesplot` package

```r
mcmc_pairs(mcmc_cadeia,pars = parametros)
```


mu
sigma
10
15
20
25
9
12
15
18