GLM(M)s for binary data

Bayesian statistics 7 – generalized linear models for binary data

Frédéric Barraquand (CNRS, IMB)

12/12/2023

Some things that we learned the last time

- Poisson models good for rare events generating small counts
- Classical *link function* is the log-link so that $Y_i|\epsilon_i \sim \mathcal{P}(\exp(a + bx_i + [\text{stuff}] + \epsilon_i))$ (with extra dispersion)
- Posterior predictive checks

The Binomial distribution: reminders and GLM

Let $U_i \sim \text{Bernoulli}(p)$ a coin toss with probability p.

- Then $Y = \sum_{i=1}^{n} U_i \sim \mathcal{B}(n,p)$
- Converges to normal distribution for large np (≥ 10) as n grows
- $\mathbb{E}(Y) = np$ and $\mathbb{V}(Y) = np(1-p)$
- Conjugate prior for p = Beta distribution.

Two ways to specify a GLM:

- $Y_i \sim \mathcal{B}(n,p)$ with $\text{logit}(p) = a + bx_i + [\text{stuff}]$ in which case the data resembles `c(31,14,5,0,19)`
- $U_i \sim \mathcal{B}(p)$ with $\text{logit}(p) = a + bx_i + [\text{stuff}]$ in which case the data resembles `c(0,0,1,0,1,1,0)`

Environnementally-driven turtle sex determination

Figure 1: Green turtle, Malaysia. Bernard Dupont. Licence: CC BY SA 2.0

First described in 1966 by Madeleine Charnier in a lizard, subsequent work on turtles (Chelonia) and Crocodylia, see Janzen & Paukstis QRB 1991.

$\mathbb{P}$(female hatchling) = f(temperature)

Figure 2: Stephan Hunt. Hatching green turtle, Ascension Island. CC BY 3.0

Empirical target

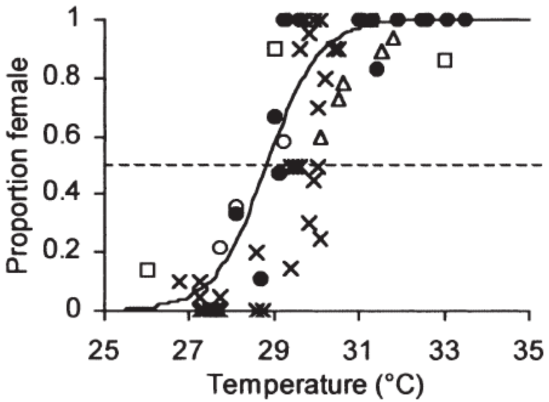


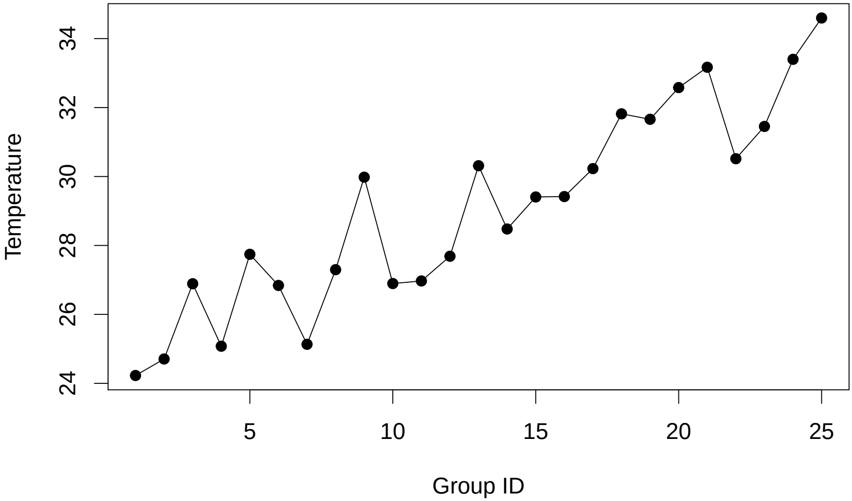
Figure 3: Temperature-dependent sex determination of Ascension Island green turtles, by Goldley et al. Marine Ecology Progress Series Vol. 226: 115–124, 2002.

Simulating data I

We have got 25 groups *i* of size 50 or less (i.e., how many baby turtles are hatched at one beach location z_i), with a different temperature in each group.

```r
sample_size_per_group = round(30*runif(25))+20
n_groups = length(sample_size_per_group)
temperature = 25 + (1:n_groups)*8/25 + rnorm(n_groups,0,2)
par(cex=1.5,pch=19)
plot(1:n_groups,temperature,type="o",
     xlab="Group ID",ylab="Temperature")
```

Simulating data II

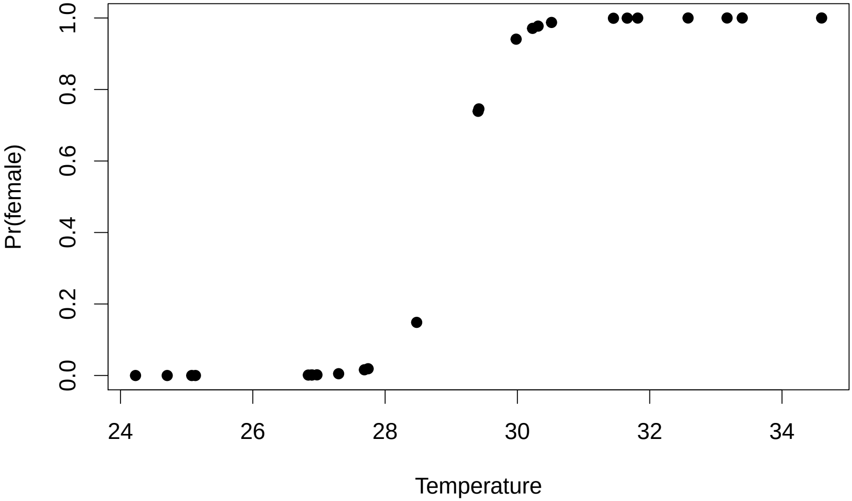


Binomial sampling I

Females are born with probability p_i in group i, which itself depends on temperature.

```r
Y = p = temperature
for (i in 1:n_groups){
  p[i] = 1/(1+exp(-3*(temperature[i]-mean(temperature)) ) )
  Y[i] = rbinom(1,sample_size_per_group[i],p[i])
}
par(cex=1.5,pch=19)
plot(temperature,p,type="p",
     xlab="Temperature",ylab="Pr(female)")
```

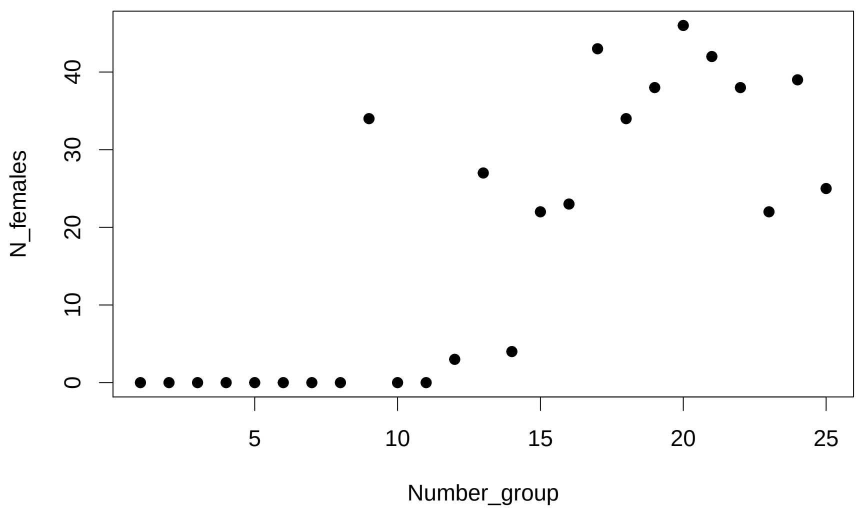
Binomial sampling II



The data I

```r
par(cex=1.5,pch=19)
plot(1:n_groups,Y,type="p",
     xlab="Number_group",ylab="N_females")
```

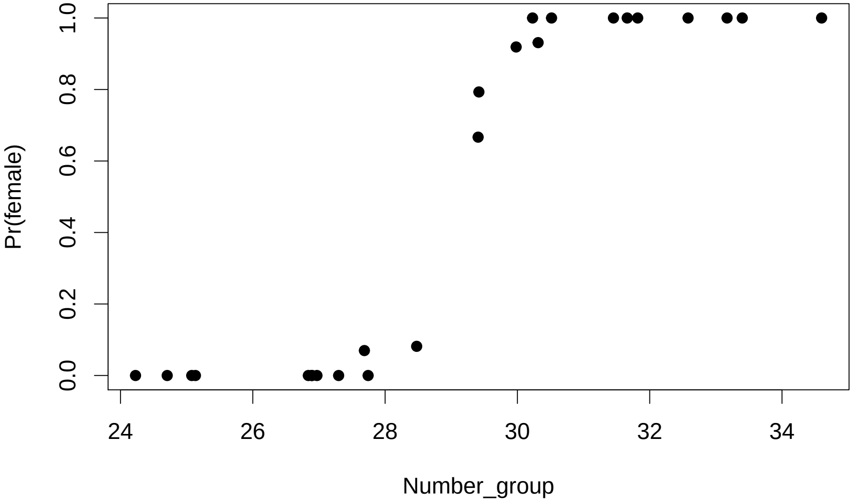
The data II



Empirical proportion estimates I

```
par(cex=1.5,pch=19)
plot(temperature,Y/sample_size_per_group,
     type="p",xlab="Number_group",
     ylab="Pr(female)")
```

Empirical proportion estimates II



Prior predictive checks

We have seen posterior predictive distribution = the distribution of imaginary data under the fitted model (given a posterior distribution).

Prior predictive distribution = the distribution of imaginary data under the priors.

Let's say we have proportion $\theta \sim \text{Beta}(\alpha, \beta)$ which is the prior for a very simple $Y_i \sim \mathcal{B}(n, \theta)$ model where n is known. Then simulating data under the prior predictive distribution can be done as

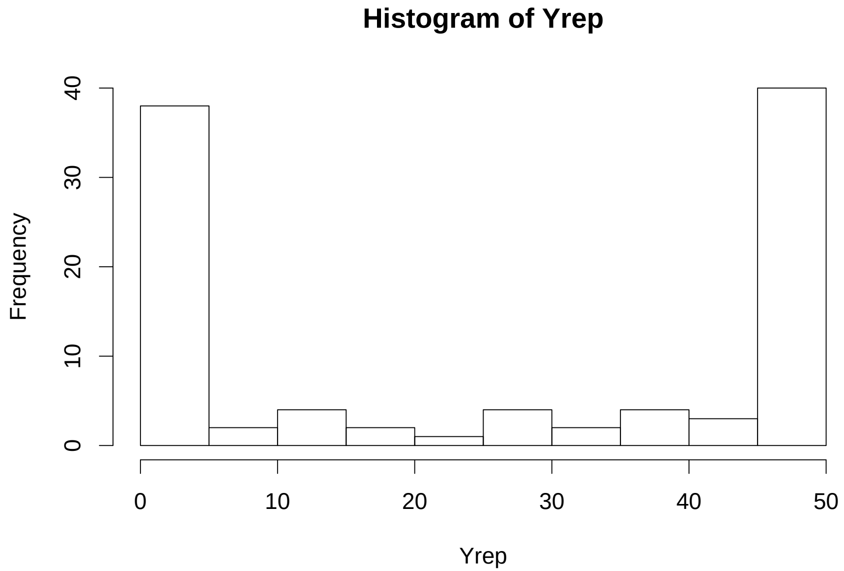
```r
for (rep in 1:nrep){
  theta[rep] = rbeta(1,alpha,beta)
  Yrep[rep] = rbinom(1,size=n,prob=theta[rep])
}
```

(it's a Beta-Binomial distribution)

Prior predictive checks (practice) I

```r
nrep = 100
n = 50
alpha=0.1
beta=0.1
Yrep = theta = rep(0,nrep)
for (rep in 1:nrep){
  theta[rep] = rbeta(1,alpha,beta)
  Yrep[rep] = rbinom(1,size=n,prob=theta[rep])
}
par(cex=1.5,pch=19)
hist(Yrep)
```

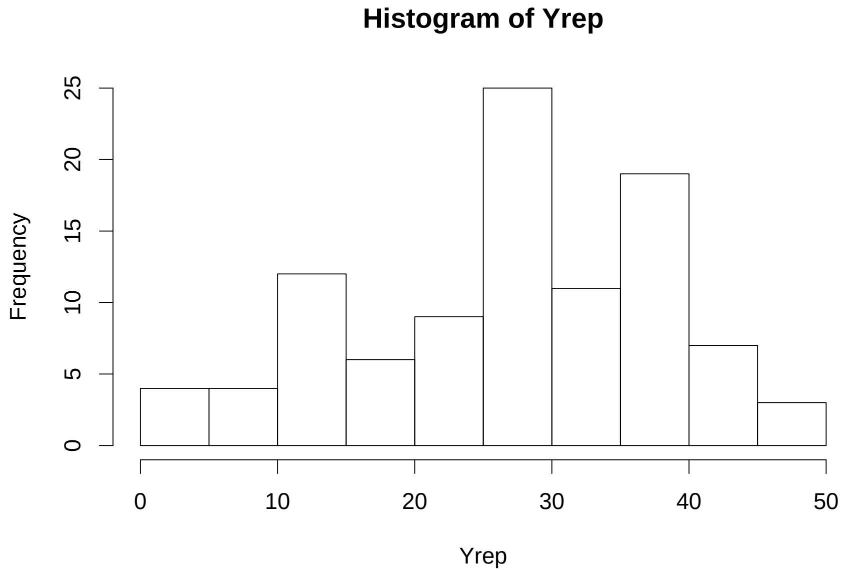
Prior predictive checks (practice) II



Prior predictive checks – better prior I

```
nrep = 100
n = 50
alpha=2
beta=2
Yrep = theta = rep(0,nrep)
for (rep in 1:nrep){
  theta[rep] = rbeta(1,alpha,beta)
  Yrep[rep] = rbinom(1,size=n,prob=theta[rep])
}
par(cex=1.5,pch=19)
hist(Yrep)
```

Prior predictive checks – better prior II



Fitting the binomial model I

Now we fit that model which writes mathematically like

$$y_i \sim \mathcal{B}(z_i, p(\mathrm{temp}_i))$$

Fitting the binomial model II

```
m11.data <- list(N = n_groups, y = Y, temp = temperature,
                 z = sample_size_per_group)

cat(file="logistic.regression.txt","
model {
  mu_temp ~ dnorm(2, 0.1) ## prior of the mean temp
  gamma ~ dnorm(1, 0.1)  ## prior of the slope

  for (k in 1:N){
  y[k] ~ dbin(p[k],z[k])     ## likelihood
  logit(p[k])<-gamma*(temp[k]-mu_temp)
  }

}
")
```

Running the model I

```r
# Inits function
inits <- function(){list(gamma = rnorm(1, 0, 1),
                         mu_temp = rnorm(1,0,1))}

# Parameters to estimate
params <- c("gamma","mu_temp")

# MCMC settings
nc <- 3  ;  ni <- 2000  ;  nb <- 1000  ;  nt <- 2

# Call JAGS, check convergence and summarize posteriors
out <- jags(m11.data, inits, params, "logistic.regression.txt", n.thin = nt,
           n.chains = nc, n.burnin = nb, n.iter = ni)
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 25
##    Unobserved stochastic nodes: 2
##    Total graph size: 156
##
## Initializing model
```

Running the model II

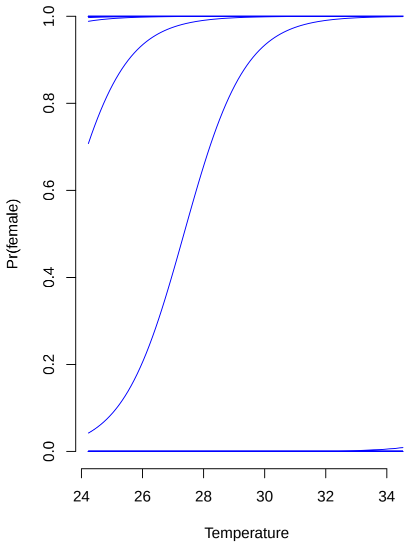
```
print(out, dig = 3)      # Bayesian analysis
```

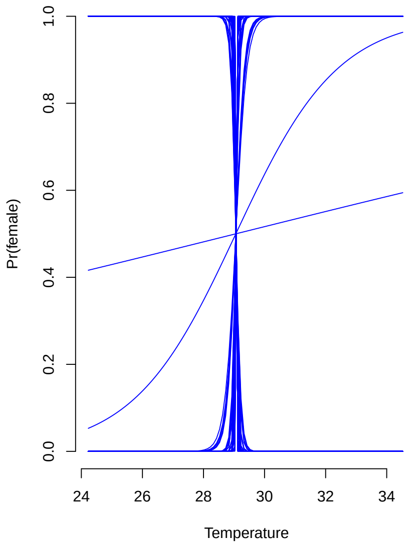
```
## Inference for Bugs model at "logistic.regression.txt", fit using jags,
##  3 chains, each with 2000 iterations (first 1000 discarded), n.thin = 2
##  n.sims = 1500 iterations saved
##          mu.vect sd.vect   2.5%    25%    50%    75%  97.5%  Rhat n.eff
## gamma      3.077   0.284  2.569  2.877  3.068  3.255  3.681 1.002  1200
## mu_temp   29.101   0.068 28.963 29.057 29.103 29.147 29.228 1.001  1500
## deviance  36.130   2.096 34.143 34.666 35.471 36.896 41.584 1.012   430
##
## For each parameter, n.eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor (at convergence, Rhat=1).
##
## DIC info (using the rule, pD = var(deviance)/2)
## pD = 2.2 and DIC = 38.3
## DIC is an estimate of expected predictive error (lower deviance is better).
```

Priors on γ and μ_{temp} I

```r
mu_temp = rnorm(100,2, 100) # prior of the mean temp
gamma = rnorm(100,1, 100)   # prior on the slope
x=seq(min(temperature),max(temperature),by=0.1)
par(mfrow=c(1,2))
plot(0, bty = 'n', pch = '', ylab = "Pr(female)",
     xlab = "Temperature",ylim=c(0,1),
     xlim=c(min(temperature),max(temperature)))
for (kprior in 1:100) {
  prob = 1/(1+exp(-1*(x-mu_temp[kprior])) )
  lines(x,prob,type="l",col="blue")}
plot(0, bty = 'n', pch = '', ylab = "Pr(female)",
     xlab = "Temperature",ylim=c(0,1),
     xlim=c(min(temperature),max(temperature)))
for (kprior in 1:100) {
  prob = 1/(1+exp(-gamma[kprior]*(x-mean(temperature))) )
  lines(x,prob,type="l",col="blue")}
```

Priors on γ and μ_{temp} II

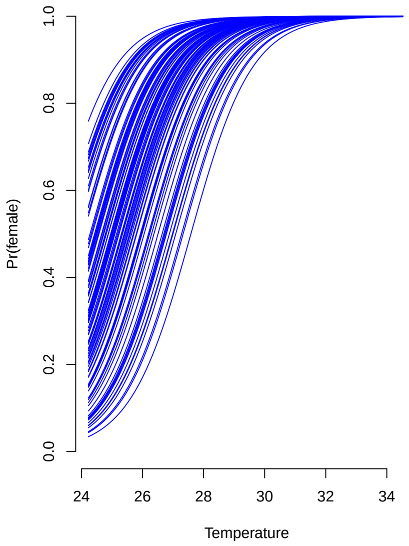


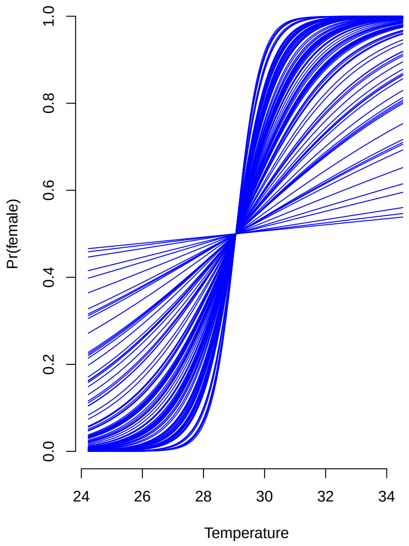


Better priors I

```r
### Better priors
mu_temp = rnorm(100,25, 1) # prior of the mean temp
gamma = rnorm(100,1, 1)   # prior on the slope
par(mfrow=c(1,2))
plot(0, bty = 'n', pch = '', ylab = "Pr(female)",
     xlab = "Temperature",ylim=c(0,1),
     xlim=c(min(temperature),max(temperature)))
for (kprior in 1:100) {
  prob = 1/(1+exp(-1*(x-mu_temp[kprior])) )
  lines(x,prob,type="l",col="blue")}
plot(0, bty = 'n', pch = '', ylab = "Pr(female)",
     xlab = "Temperature",ylim=c(0,1),
     xlim=c(min(temperature),max(temperature)))
for (kprior in 1:100) {
  prob = 1/(1+exp(-abs(gamma[kprior])*(x-mean(temperature))) )
  lines(x,prob,type="l",col="blue")}
```

Better priors II





Weakly informative priors I

What are they?

- Priors that provide *regularization* or *shrinkage*
- In practice, often $\mathcal{N}(0, [\text{small}])$ instead of $\mathcal{N}(0, [\text{huge}])$, e.g. `dnorm(0,1)` or `dnorm(0,0.1)` instead of `dnorm(0,0.0001)` in JAGS.

A more detailed explanation

A detour on identifiability, convergence, and priors

Borrowed from Mc Elreath's Statistical rethinking

We consider the (obviously wrong) model

$$Y_i \sim \mathcal{N}(\mu, \sigma^2)$$

$$\mu = \alpha_1 + \alpha_2$$

$$\alpha_j \sim \text{Unif(-10000,10000) i.i.d}$$

$$\sigma \sim \text{Exp}(1/10)$$

for 100 data points simulated as $\mathcal{N}(0,1)$.

Coding the stupid model

```
gaussian.data = list(y=rnorm(100,0,1),N=100)

cat(file="stupid.model.txt","
model {
    # Priors
    alpha[1] ~ dunif(-10000,10000) #dnorm(0,0.00001)
    alpha[2] ~ dunif(-10000,10000) #dnorm(0,0.00001)
    sigma ~ dunif(0,1000) #dexp(0.01) #dexp(1)
    tau<-pow(sigma,-2)

    # Likelihood
    mu<-alpha[1]+alpha[2]
    for (i in 1:N){
    y[i] ~ dnorm(mu,tau)
    }

}
")
```

Fitting the stupid model I

```r
# Initial values
inits <- function(){list(alpha=rnorm(2,0,1000))}
# Parameters to estimate
params <- c("alpha","sigma","mu")
# MCMC settings
nc <- 3  ;  ni <- 2000  ;  nb <- 1000  ;  nt <- 2
# Call JAGS, check convergence and summarize posteriors
out <- jags(gaussian.data, inits, params, "stupid.model.txt", n.thin = nt,
            n.chains = nc, n.burnin = nb, n.iter = ni)
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 100
##    Unobserved stochastic nodes: 3
##    Total graph size: 112
##
## Initializing model
```

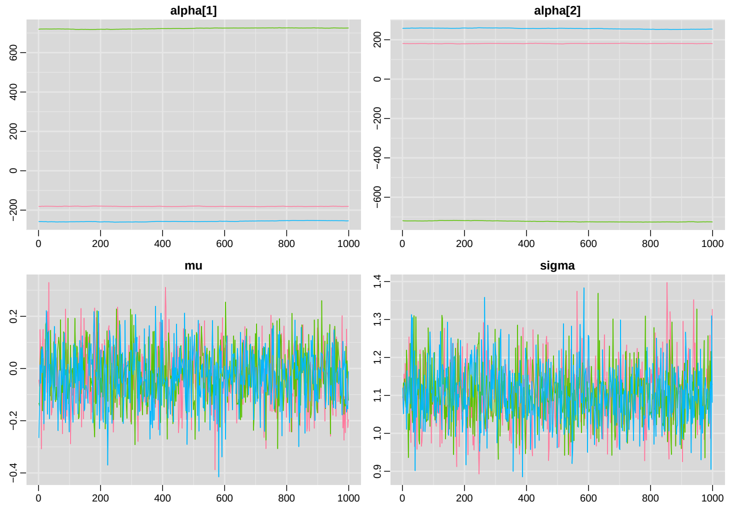
```r
print(out, dig = 3)     # Bayesian analysis
```

Fitting the stupid model II

```
## Inference for Bugs model at "stupid.model.txt", fit using jags,
##  3 chains, each with 2000 iterations (first 1000 discarded), n.thin = 2
##  n.sims = 1500 iterations saved
##           mu.vect sd.vect     2.5%      25%      50%     75%   97.5%    Rhat
## alpha[1]   95.009 445.065 -259.966 -254.465 -180.816 720.026 725.701 372.057
## alpha[2]  -95.034 445.061 -725.773 -720.024  180.816 254.485 259.924 372.253
## mu         -0.025   0.109   -0.240   -0.102   -0.022   0.050   0.184   1.002
## sigma       1.103   0.078    0.960    1.050    1.100   1.149   1.267   1.000
## deviance  302.217   1.900  300.302  300.840  301.629 303.049 306.979   1.007
##           n.eff
## alpha[1]      3
## alpha[2]      3
## mu         1200
## sigma      1500
## deviance    760
##
## For each parameter, n.eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor (at convergence, Rhat=1).
##
## DIC info (using the rule, pD = var(deviance)/2)
## pD = 1.8 and DIC = 304.0
## DIC is an estimate of expected predictive error (lower deviance is better).
```

Traceplots

```
library(mcmcplots)
traplot(as.mcmc(out),parms=c("alpha","mu","sigma"))
```



Less stupid model

```
cat(file="less.stupid.model.txt","
model {
    # Priors
    alpha[1] ~ dnorm(0,0.1)
    alpha[2] ~ dnorm(0,0.1)
    sigma ~ dexp(1)
    tau<-pow(sigma,-2)

    # Likelihood
    mu<-alpha[1]+alpha[2]
    for (i in 1:N){
    y[i] ~ dnorm(mu,tau)
    }

}
")
```

Actually even a much smaller precision on α_j's prior would work

Fitting the less stupid model I

```r
# Initial values
inits <- function(){list(alpha=rnorm(2,0,1))}
# Parameters to estimate
params <- c("alpha","sigma","mu")
# MCMC settings
nc <- 3  ;  ni <- 2000  ;  nb <- 1000  ;  nt <- 2
# Call JAGS, check convergence and summarize posteriors
out <- jags(gaussian.data, inits, params, "less.stupid.model.txt",
            n.thin = nt, n.chains = nc, n.burnin = nb, n.iter = ni)
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 100
##    Unobserved stochastic nodes: 3
##    Total graph size: 111
##
## Initializing model
```

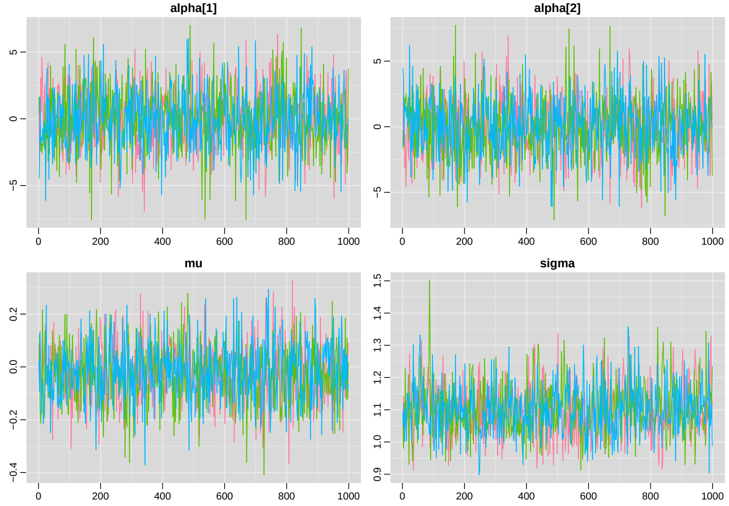
```r
print(out, dig = 3)     # Bayesian analysis
```

Fitting the less stupid model II

```
## Inference for Bugs model at "less.stupid.model.txt", fit using jags,
##  3 chains, each with 2000 iterations (first 1000 discarded), n.thin = 2
##  n.sims = 1500 iterations saved
##          mu.vect sd.vect    2.5%     25%     50%     75%   97.5%  Rhat n.eff
## alpha[1]   0.019   2.265  -4.521  -1.507  -0.004   1.590   4.362 1.002  1000
## alpha[2]  -0.043   2.267  -4.403  -1.632  -0.002   1.476   4.556 1.002   900
## mu        -0.024   0.110  -0.233  -0.096  -0.022   0.048   0.200 1.004   460
## sigma      1.102   0.079   0.958   1.049   1.098   1.148   1.272 1.001  1500
## deviance 302.257   2.023 300.294 300.806 301.597 303.074 307.792 1.001  1500
##
## For each parameter, n.eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor (at convergence, Rhat=1).
##
## DIC info (using the rule, pD = var(deviance)/2)
## pD = 2.0 and DIC = 304.3
## DIC is an estimate of expected predictive error (lower deviance is better).
```

Traceplots again

```
library(mcmcplots)
traplot(as.mcmc(out),parms=c("alpha","mu","sigma"))
```



Now a real binary data example: bald eagles I

Figure 4: Bald eagle with salmon, Alaska. KJ Gill aka Gillfoto. CC BY 2.0

Now a real binary data example: bald eagles II

Records of (160!) salmon-pirating attempts by one Bald eagle on another Bald eagle (not always the same!). Also borrowed from McElreath's Statistical Rethinking p. 330.

```
library(MASS)
data(eagles)
head(eagles)
```

```
##    y  n P A V
## 1 17 24 L A L
## 2 29 29 L A S
## 3 17 27 L I L
## 4 20 20 L I S
## 5  1 12 S A L
## 6 15 16 S A S
```

Now a real binary data example: bald eagles III

```
#P // Size of pirating eagle (L = large, S = small).
#A // Age of pirating eagle (I = immature, A = adult).
#V // Size of victim eagle (L = large, S = small).

m2.data = list(N=nrow(eagles),y=eagles$y,z=eagles$n,
               P=as.numeric(eagles$P)-1,A=as.numeric(eagles$A)-1,
               V=as.numeric(eagles$V)-1)
# m2.data = list(N=nrow(eagles),y=eagles$y,z=eagles$n,P=eagles$P,A=
```