Comparison of Bayesian software

In this example, we compare JAGS to other Bayesian software to samples from a random slopes models. Let Y_{ij} be the j^{th} measurement of jaw bone density for patient i. In this model we allow bone density to increase linearly in time and each patient has their own slope and intercept. The random slope model is

$$Y_{ij} \,|\, \alpha_{i1}, \alpha_{i2} \sim \text{Normal}(\alpha_{i1} + age_j \alpha_{i2}, \sigma_3^2) \text{ where } (\alpha_{1i}, \alpha_{2i})^T \sim \text{Normal}(\mu, \Sigma).$$

The random effects α_{i1} and α_{i2} are the subject-specific intercept and slope, respectively. The population of intercepts and slopes are assumed to be $\alpha_{ji} \sim \text{Normal}(\mu_j, \sigma_j)$.

This model is fit below using

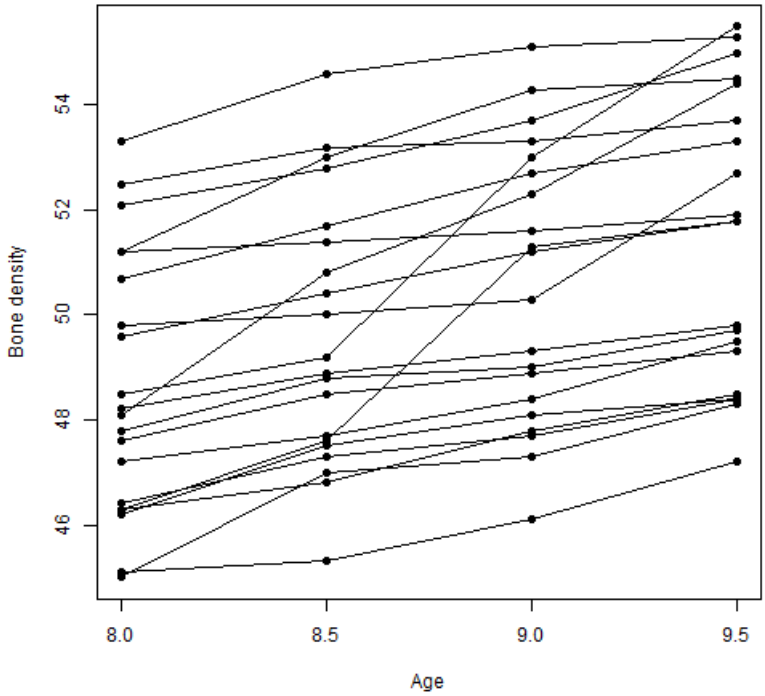
1. JAGS
2. OpenBUGS
3. STAN
4. NIMBLE

Load jaw data

```
m   <- 4
n   <- 20
age <- c(8.0, 8.5, 9.0, 9.5)
Y   <- c(47.8, 48.8, 49.0, 49.7,
         46.4, 47.3, 47.7, 48.4,
         46.3, 46.8, 47.8, 48.5,
         45.1, 45.3, 46.1, 47.2,
         47.6, 48.5, 48.9, 49.3,
         52.5, 53.2, 53.3, 53.7,
         51.2, 53.0, 54.3, 54.5,
         49.8, 50.0, 50.3, 52.7,
         48.1, 50.8, 52.3, 54.4,
         45.0, 47.0, 47.3, 48.3,
         51.2, 51.4, 51.6, 51.9,
         48.5, 49.2, 53.0, 55.5,
         52.1, 52.8, 53.7, 55.0,
         48.2, 48.9, 49.3, 49.8,
         49.6, 50.4, 51.2, 51.8,
         50.7, 51.7, 52.7, 53.3,
         47.2, 47.7, 48.4, 49.5,
         53.3, 54.6, 55.1, 55.3,
         46.2, 47.5, 48.1, 48.4,
         46.3, 47.6, 51.3, 51.8)

Y <- matrix(Y,20,4,byrow=TRUE)

plot(NA,xlim=range(age),ylim=range(Y),xlab="Age",ylab="Bone density")
for(i in 1:n){
  lines(age,Y[i,])
  points(age,Y[i,],pch=19)
}
```



(1) Mixed model in JAGS

```
library(rjags)
```

```
## Loading required package: coda
```

```
## Linked to JAGS 4.2.0
```

```
## Loaded modules: basemod,bugs
```

```
set.seed(0820)

tick <- proc.time()[3]
model_string <- textConnection("model{
  # Likelihood
   for(i in 1:n){for(j in 1:m){
     Y[i,j] ~ dnorm(alpha1[i]+alpha2[i]*age[j],tau3)
   }}

  # Random effects
   for(i in 1:n){
     alpha1[i] ~ dnorm(mu1,tau1)
     alpha2[i] ~ dnorm(mu2,tau2)
   }

   mu1 ~ dnorm(0,0.0001)
   mu2 ~ dnorm(0,0.0001)
  tau1 ~ dgamma(0.1,0.1)
  tau2 ~ dgamma(0.1,0.1)
  tau3 ~ dgamma(0.1,0.1)
}")

data    <- list(Y=Y,age=age,n=n,m=m)
params  <- c("mu1","mu2","tau1","tau2","tau3")
model   <- jags.model(model_string,data = data, n.chains=2,quiet=TRUE)
update(model, 10000, progress.bar="none")
samples <- coda.samples(model, variable.names=params,
                        n.iter=90000, progress.bar="none")
tock <- proc.time()[3]
tock-tick
```

```
## elapsed
##    1.83
```

```
effectiveSize(samples)
```

```
##        mu1        mu2       tau1       tau2       tau3
##   292.9739   335.4025  3406.6998  6325.8104 15608.3856
```

(2) Mixed model in OpenBUGS

```
library(R2OpenBUGS)
```

```
## Warning: package 'R2OpenBUGS' was built under R version 3.3.3
```

```
set.seed(0820)

tick <- proc.time()[3]

model_string <- function(){
  # Likelihood
   for(i in 1:n){for(j in 1:m){
     Y[i,j]   ~ dnorm(mn[i,j],tau3)
     mn[i,j] <- alpha1[i]+alpha2[i]*age[j]
   }}

  # Random effects
   for(i in 1:n){
     alpha1[i] ~ dnorm(mu1,tau1)
     alpha2[i] ~ dnorm(mu2,tau2)
   }

  # Priors
   mu1 ~ dnorm(0,0.0001)
   mu2 ~ dnorm(0,0.0001)
  tau1 ~ dgamma(0.1,0.1)
  tau2 ~ dgamma(0.1,0.1)
  tau3 ~ dgamma(0.1,0.1)
}

data    <- list(Y=Y,age=age,n=n,m=m)
params  <- c("mu1","mu2","tau1","tau2","tau3")
inits <- function(){list(mu1=0,mu2=0,tau1=.1,tau2=.2,tau3=.2)}
fit   <- bugs(model.file=model_string,
              data=data,inits=inits,
              parameters.to.save=params,DIC=FALSE,
              n.iter=100000,n.chains=2,n.burnin=10000)
tock <- proc.time()[3]
tock-tick
```

```
## elapsed
##   10.17
```

```
fit$summary[,9]
```

```
##    mu1    mu2   tau1   tau2   tau3
##   1300   1200 180000  11000 180000
```

(3) Mixed model in STAN

```
library(rstan)
```

```
## Warning: package 'rstan' was built under R version 3.3.3
```

```
## Loading required package: ggplot2
```

```
## Use suppressPackageStartupMessages() to eliminate package startup
## messages.
```

```
## Loading required package: StanHeaders
```

```
## Warning: package 'StanHeaders' was built under R version 3.3.3
```

```
## rstan (Version 2.17.3, GitRev: 2e1f913d3ca3)
```

```
## For execution on a local, multicore CPU with excess RAM we recommend calling
## options(mc.cores = parallel::detectCores()).
## To avoid recompilation of unchanged Stan programs, we recommend calling
## rstan_options(auto_write = TRUE)
```

```
##
## Attaching package: 'rstan'
```

```
## The following object is masked from 'package:R2OpenBUGS':
##
##     monitor
```

```
## The following object is masked from 'package:coda':
##
##     traceplot
```

```
set.seed(0820)

tick <- proc.time()[3]
stan_model <- "

  data {
   int<lower=0> n;
   int<lower=0> m;
   vector [m] age;
   matrix [n,m] Y;
  }

  parameters {
    vector [n] alpha1;
    vector [n] alpha2;
    real mu1;
    real mu2;
    real<lower=0> sigma3;
    real<lower=0> sigma2;
    real<lower=0> sigma1;
  }

  model {
    real mu;
    alpha1 ~ normal(0,sigma1);
    alpha2 ~ normal(0,sigma2);
    sigma1 ~ cauchy(0.0,1000);
    sigma2 ~ cauchy(0.0,1000);
    sigma3 ~ cauchy(0.0,1000);
    mu1    ~ normal(0,10000);
    mu2    ~ normal(0,10000);

    for(i in 1:n){for(j in 1:m){
      mu      = alpha1[i] + alpha2[i]*age[j];
      Y[i,j]  ~ normal(mu,sigma3);
    }}
  }
"

data    <- list(Y=Y,age=age,n=n,m=m)
fit_stan <- stan(model_code = stan_model,
                 data = data,
                 chains=2, warmup = 10000, iter = 100000)
```

```
## In file included from C:/Program Files/R/R-3.3.2/library/BH/include/boost/config.hpp:39:0,
##                  from C:/Program Files/R/R-3.3.2/library/BH/include/boost/math/tools/config.hpp:13,
##                  from C:/Program Files/R/R-3.3.2/library/StanHeaders/include/stan/math/rev/core/var.hpp:7,
##                  from C:/Program Files/R/R-3.3.2/library/StanHeaders/include/stan/math/rev/core/gevv_vvv_var
```

```
##                  from C:/Program Files/R/R-3.3.2/library/StanHeaders/include/stan/math/rev/core.hpp:12,
##                  from C:/Program Files/R/R-3.3.2/library/StanHeaders/include/stan/math/rev/mat.hpp:4,
##                  from C:/Program Files/R/R-3.3.2/library/StanHeaders/include/stan/math.hpp:4,
##                  from C:/Program Files/R/R-3.3.2/library/StanHeaders/include/src/stan/model/model_header.hpp
##                  from file175c253b4319.cpp:8:
## C:/Program Files/R/R-3.3.2/library/BH/include/boost/config/compiler/gcc.hpp:186:0: warning: "BOOST_NO_CXX11_
##  #  define BOOST_NO_CXX11_RVALUE_REFERENCES
##  ^
## <command-line>:0:0: note: this is the location of the previous definition
## In file included from C:/Program Files/R/R-3.3.2/library/StanHeaders/include/stan/math/rev/core.hpp:44:0,
##                  from C:/Program Files/R/R-3.3.2/library/StanHeaders/include/stan/math/rev/mat.hpp:4,
##                  from C:/Program Files/R/R-3.3.2/library/StanHeaders/include/stan/math.hpp:4,
##                  from C:/Program Files/R/R-3.3.2/library/StanHeaders/include/src/stan/model/model_header.hpp
##                  from file175c253b4319.cpp:8:
## C:/Program Files/R/R-3.3.2/library/StanHeaders/include/stan/math/rev/core/set_zero_all_adjoints.hpp:14:17: w
##      static void set_zero_all_adjoints() {
##                  ^
##
## SAMPLING FOR MODEL '4bf61f87d83553920d54d39a64269860' NOW (CHAIN 1).
##
## Gradient evaluation took 0 seconds
## 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Adjust your expectations accordingly!
##
##
## Iteration:     1 / 100000 [  0%]  (Warmup)
## Iteration: 10000 / 100000 [ 10%]  (Warmup)
## Iteration: 10001 / 100000 [ 10%]  (Sampling)
## Iteration: 20000 / 100000 [ 20%]  (Sampling)
## Iteration: 30000 / 100000 [ 30%]  (Sampling)
## Iteration: 40000 / 100000 [ 40%]  (Sampling)
## Iteration: 50000 / 100000 [ 50%]  (Sampling)
## Iteration: 60000 / 100000 [ 60%]  (Sampling)
## Iteration: 70000 / 100000 [ 70%]  (Sampling)
## Iteration: 80000 / 100000 [ 80%]  (Sampling)
## Iteration: 90000 / 100000 [ 90%]  (Sampling)
## Iteration: 100000 / 100000 [100%]  (Sampling)
##
##  Elapsed Time: 15.162 seconds (Warm-up)
##                189.403 seconds (Sampling)
##                204.565 seconds (Total)
##
##
## SAMPLING FOR MODEL '4bf61f87d83553920d54d39a64269860' NOW (CHAIN 2).
##
## Gradient evaluation took 0 seconds
## 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Adjust your expectations accordingly!
##
##
## Iteration:     1 / 100000 [  0%]  (Warmup)
## Iteration: 10000 / 100000 [ 10%]  (Warmup)
## Iteration: 10001 / 100000 [ 10%]  (Sampling)
## Iteration: 20000 / 100000 [ 20%]  (Sampling)
## Iteration: 30000 / 100000 [ 30%]  (Sampling)
## Iteration: 40000 / 100000 [ 40%]  (Sampling)
## Iteration: 50000 / 100000 [ 50%]  (Sampling)
## Iteration: 60000 / 100000 [ 60%]  (Sampling)
## Iteration: 70000 / 100000 [ 70%]  (Sampling)
## Iteration: 80000 / 100000 [ 80%]  (Sampling)
## Iteration: 90000 / 100000 [ 90%]  (Sampling)
## Iteration: 100000 / 100000 [100%]  (Sampling)
##
##  Elapsed Time: 16.439 seconds (Warm-up)
##                173.239 seconds (Sampling)
```

```
##                189.678 seconds (Total)
```

```
tock <- proc.time()[3]
tock-tick
```

```
## elapsed
##  424.27
```

```
summary(fit_stan)$summary[41:45,9:10]
```

```
##            n_eff      Rhat
## mu1    180000.00 1.0000077
## mu2    180000.00 1.0000082
## sigma3  79991.02 0.9999959
## sigma2 180000.00 0.9999899
## sigma1 180000.00 0.9999975
```

(4) Mixed model in NIMBLE

```
library(nimble)
```

```
## nimble version 0.6-11 is loaded.
## For more information on NIMBLE and a User Manual,
## please visit http://R-nimble.org.
```

```
##
## Attaching package: 'nimble'
```

```
## The following object is masked from 'package:stats':
##
##     simulate
```

```
set.seed(0820)

tick <- proc.time()[3]

model_string <- nimbleCode({
  # Likelihood
   for(i in 1:n){for(j in 1:m){
     Y[i,j]   ~ dnorm(mn[i,j],tau3)
     mn[i,j] <- alpha1[i]+alpha2[i]*age[j]
   }}

  # Random effects
   for(i in 1:n){
     alpha1[i] ~ dnorm(mu1,tau1)
     alpha2[i] ~ dnorm(mu2,tau2)
   }

  # Priors
   mu1 ~ dnorm(0,0.0001)
   mu2 ~ dnorm(0,0.0001)
  tau1 ~ dgamma(0.1,0.1)
  tau2 ~ dgamma(0.1,0.1)
  tau3 ~ dgamma(0.1,0.1)
})

consts   <- list(n=n,m=m,age=age)
data     <- list(Y=Y)
inits    <- function(){list(mu1=rnorm(1),mu2=rnorm(1),tau1=10,tau2=10,tau3=10)}
samples  <- nimbleMCMC(model_string, data = data, inits = inits,
                       constants=consts,
                       monitors = c("mu1", "mu2","tau1","tau2","tau3"),
                       samplesAsCodaMCMC=TRUE,WAIC=FALSE,
                       niter = 100000, nburnin = 10000, nchains = 2)
```

```
## defining model...
```

```
## building model...
```

```
## setting data and initial values...
```

```
## running calculate on model (any error reports that follow may simply
## reflect missing values in model variables) ...
```

```
##
```

```
## checking model sizes and dimensions...
```

```
## This model is not fully initialized. This is not an error. To see which
## variables are not initialized, use model$initializeInfo(). For more
## information on model initialization, see help(modelInitialization).
```

```
##
```

```
## checking model calculations...
```

```
## NAs were detected in model variables: alpha1, logProb_alpha1, alpha2, logProb_alpha2, mn, logProb_Y.
```

```
## model building finished.
```

```
## compiling... this may take a minute. Use 'showCompilerOutput = TRUE' to see C++ compiler details.
```

```
## compilation finished.
```

```
## running chain 1...
```

```
## |-------------|-------------|-------------|-------------|
## |-----------------------------------------------------|
```

```
## running chain 2...
```

```
## |-------------|-------------|-------------|-------------|
## |-----------------------------------------------------|
```

```
tock <- proc.time()[3]
tock-tick
```

```
## elapsed
##   26.49
```

```
effectiveSize(samples)
```

```
##        mu1        mu2       tau1       tau2       tau3
##   283.4198   310.7662  3448.1076  6237.2704 13035.0535
```