OpenBUGS example

Patrick Brown

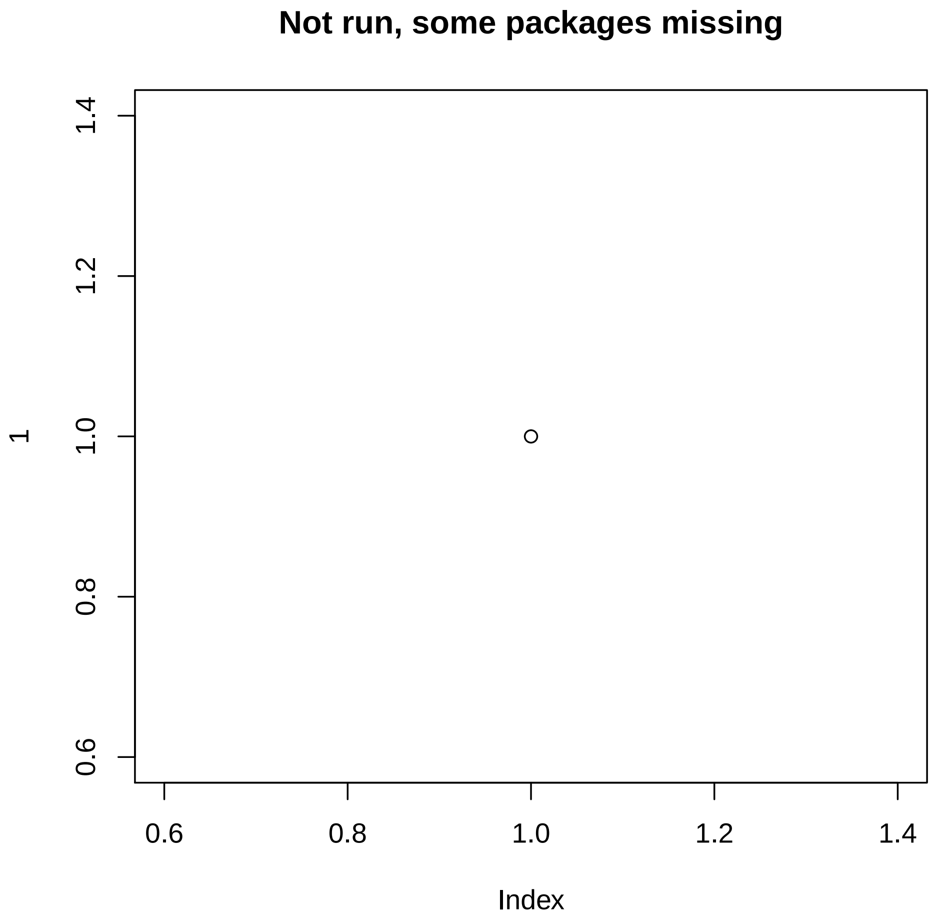
21 September 2016

```
> library('glmmBUGS')
> haveBugs = require('R2OpenBUGS', quietly=TRUE)
> print(haveBugs)

[1] FALSE
```

1 Bacteria

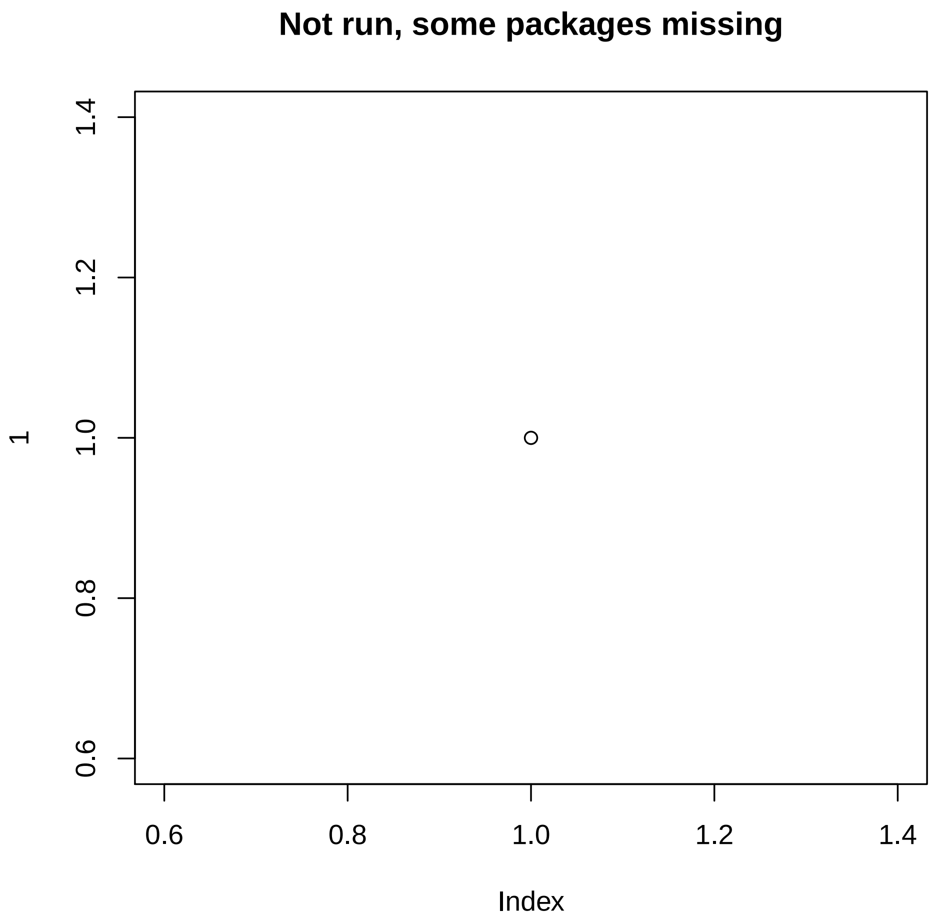
```
> data('bacteria', package='MASS')
> bacterianew <- bacteria
> bacterianew$yInt = as.integer(bacterianew$y == "y")
> levels(bacterianew$trt) <- c("placebo", "drug", "drugplus")

> bacrag <- glmmBUGS(formula = yInt ~ trt + week,
+     data = bacterianew,
+     effects = "ID", modelFile = "bacteria.txt",
+     family = "bernoulli",brugs=TRUE)
> source("getInits.R")
> startingValues = bacrag$startingValues

> bacResult = NULL
> if(haveBugs) {
+          bacResult = try(
+                        R2OpenBUGS::bugs(
+                                        bacrag$ragged, inits=getInits,
+                                        model.file = "bacteria.txt",
+                                        n.chain = 3,
+                                        n.iter = 600, n.burnin = 10,
+                                        parameters = names(getInits()),
+                                        n.thin = 4, OpenBUGS.pgm = obExec),
+                        silent=TRUE)
+         if(class(bacResult)=='try-error') {
+                 bacResult = NULL
+         }
+ }
```

```
> if(!is.null(bacResult)) {
+         bacParams = restoreParams(
+                         bacResult, bacrag$ragged)
+ }

> if(!is.null(bacResult)) {
+         bacsummary = summaryChain(bacParams)
+         bacsummary$betas[,c('mean', 'sd')]
+ }

> if(!is.null(bacResult)) {
+         checkChain(bacParams, c("intercept", "SDID"),oneFigure=TRUE)
+ } else {
+         plot(1, main='Not run, some packages missing')
+ }
```

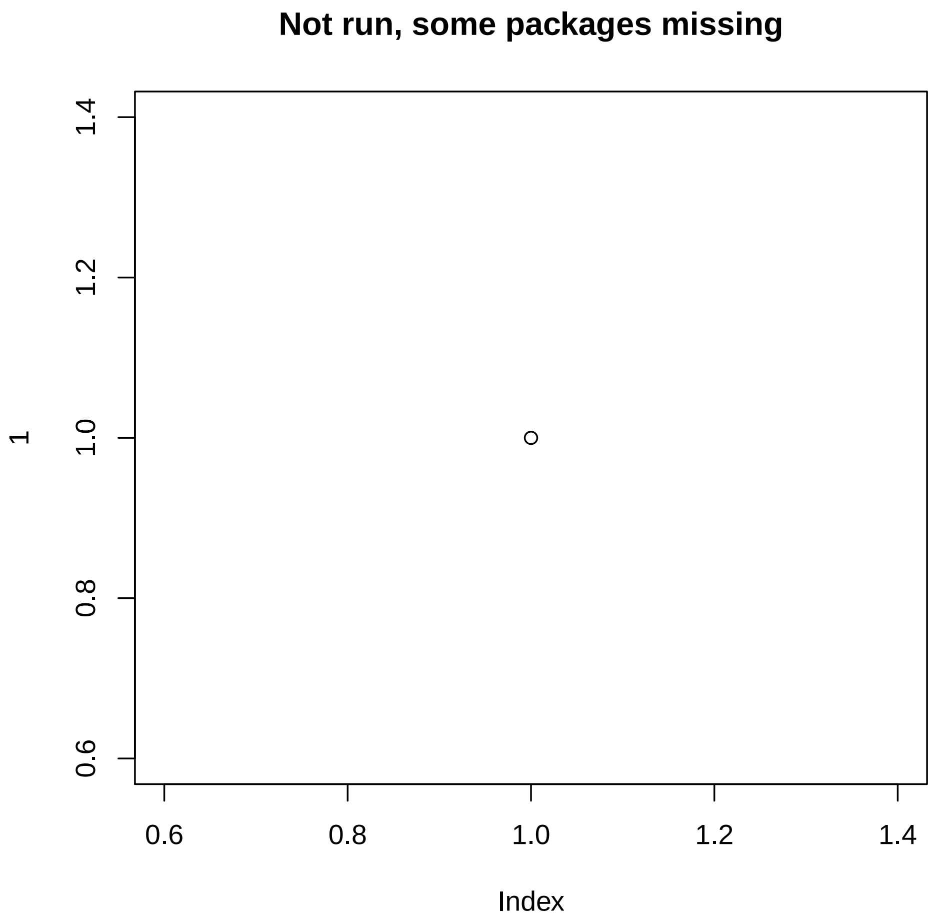


2 BYM model

```
> havePackages = c(
+     'diseasemapping'=require('diseasemapping', quietly=TRUE),
+     "spdep"=require('spdep', quietly=TRUE),
+     'R2OpenBUGS'= haveBugs
+ )
> print(havePackages)

diseasemapping          spdep     R2OpenBUGS
          TRUE           TRUE          FALSE

> if(all(havePackages)){
+   data('kentucky', package='diseasemapping')
+   larynxRates = structure(c(0, 0, 0, 0, 1e-06, 6e-06, 2.3e-05, 4.5e-05, 9.9e-05,
+           0.000163, 0.000243, 0.000299, 0.000343, 0.000308, 0.000291, 0.000217,
+           0, 0, 0, 1e-06, 1e-06, 3e-06, 8e-06, 1.3e-05, 2.3e-05, 3.5e-05,
+           5.8e-05, 6.8e-05, 7.5e-05, 5.5e-05, 4.1e-05, 3e-05), .Names = c("M_10",
+           "M_15", "M_20", "M_25", "M_30", "M_35", "M_40", "M_45", "M_50",
+           "M_55", "M_60", "M_65", "M_70", "M_75", "M_80", "M_85", "F_10",
+           "F_15", "F_20", "F_25", "F_30", "F_35", "F_40", "F_45", "F_50",
+           "F_55", "F_60", "F_65", "F_70", "F_75", "F_80", "F_85"))
+
+         kentucky = getSMR(kentucky, larynxRates, larynx,
+             regionCode="County")
+
+         kAdjMat = spdep::poly2nb(kentucky,
+             row.names=as.character(kentucky$County))
+ }

> if(all(havePackages)){
+         forBugs = glmmBUGS(observed + logExpected ~ poverty,
+             effects="County", family="poisson",
+             spatial=kAdjMat,
+             modelFile='bym.txt',
+             data=kentucky@data
+         )
+
+         startingValues = forBugs$startingValues
+
+         source("getInits.R")
+
+         # OpenBUGS wont run unless
+     # all the starting values for RCountySpatial are zero
+         int2 = function() {
+           res = getInits()
+           res$RCountySpatial = rep(0,
+                                 length(res$RCountySpatial))
+           res
+         }
+ }

> kResult = NULL
> if(all(havePackages)){
+         kResult = try(
+                 R2OpenBUGS::bugs(forBugs$ragged,
+                         inits=int2,
+                         model.file = "bym.txt",
+                         n.chain = 2,
+                         n.iter = 500, n.burnin = 10,
+                         parameters = names(int2()),
+                         n.thin = 10, OpenBUGS.pgm = obExec
+                         ), silent=TRUE)
+         if(class(bacResult)=='try-error') {
+                 kResult = NULL
+         }
+ }

> if(!is.null(kResult)){
+         kParams = restoreParams(kResult,
+             forBugs$ragged)
+ }

> if(!is.null(kResult)){
+         kSummary = summaryChain(kParams)
+         kSummary$scalars[,c('mean', 'sd')]
+ }

> if(!is.null(kResult)){
+         checkChain(kParams, c("intercept", "SDCountySpatial"))
+ } else {
+         plot(1, main='Not run, some packages missing')
+ }
```



```
> if(!is.null(kResult)){
+         kentucky = mergeBugsData(kentucky, kSummary)
+         print(spplot(kentucky, "FittedRateCounty.mean"))
+ } else {
+         plot(1, main='Not run, some packages missing')
+ }
```



Not run, some packages missing