Example 4.4

Disease mapping: from foundations to multidimensional modeling

Martinez-Beneito M.A. and Botella-Rocamora P.

This document reproduces the analysis made at Example 4.4 of the book: "Disease mapping: from foundations to multidimensional modeling" by Martinez-Beneito M.A. and Botella-Rocamora P., published by CRC press in 2019. You can watch the analysis made with full detail at this pdf document, or even execute it if you want with the material available at https://github.com/MigueBeneito/DMBook. Anyway, this pdf file should be enough for following most of the details of the analysis made for this example.

The statistical analysis below has been run in `R`, by additionally using the library `Rmarkdown`, so be sure that you have this software installed if you want to reproduce by yourself the content of this document. In that case we advise you to download first the annex material at https://github.com/MigueBeneito/DMBook, open with `Rstudio` the corresponding `.Rproj` file that you will find at the folder corresponding to this example and compile the corresponding `.Rmd` document. This will allow you to reproduce the whole statistical analysis below.

This document has been executed with real data that are not provided in order to preserve their confidentiality. Slightly modified data are provided instead, as described in Chapter 1 of the book. Thus, when reproducing this document you will not obtain exactly the same results, although they should be very close to those shown here.

Libraries and data loading

```
# Libraries loading
#-----------------
if (!require(RColorBrewer)) {
    install.packages("RColorBrewer")
    library(RColorBrewer)
}
if (!require(rgdal)) {
    install.packages("rgdal")
    library(rgdal)
}
if (!require(DCluster)) {
    install.packages("DCluster")
    library(DCluster)
}
if (!require(INLA)) {
    install.packages("INLA", repos = c(getOption("repos"), INLA = "https://inla.r-inla-download.org/R/s
        dep = TRUE)
    library(INLA)
}
if (!require(pbugs)) {
    if (!require(devtools)) {
        install.packages("devtools")
        devtools::install_github("fisabio/pbugs")
    } else {
        install_github("fisabio/pbugs")
    }
}
```

```r
# Data loading
#------------
# For reproducing the document, the following line should be changed to
# load('../Data/ObsOral-mod.Rdata') since that file contains the
# modified data making it possible to reproduce this document.
load("../Data/ObsOral.Rdata")
# load('../Data/ObsOral-mod.Rdata')
load("../Data/ExpOral.Rdata")
load("../Data/Population.Rdata")
load("../Data/VR.Rdata")
```

R function for calculating the DIC criterion of the models fitted

The function below computes the DIC criterion for disease mapping models fitted with WinBUGS. It returns DIC values comparable to those reported by INLA, in contrast to WinBUGS. See annex material for Example 4.3.

```r
# Arguments: Simu.sSMRs: matrix of dimensions n.IterXn.Units where
# n.Iter are the number of MCMC iterations saved and n.Units the number
# of spatial units in the analysis. You will typically find this as a
# submatrix of the sims.matrix element of any bugs object. O: Vector of
# length n.Units with the observed deaths per spatial unit. E: Vector
# of length n.Units with the expected deaths per spatial unit.
DICPoisson = function(Simu.sSMRs, O, E) {
    mu = t(apply(Simu.sSMRs/100, 1, function(x) {
        x * E
    }))
    D = apply(mu, 1, function(x) {
        -2 * sum(O * log(x) - x - lfactorial(O))
    })
    Dmean = mean(D)
    mumean = apply(Simu.sSMRs/100, 2, mean) * E
    DinMean = -2 * sum(O * log(mumean) - mumean - lfactorial(O))
    # if(save==TRUE){return(c(Dmedia,Dmedia-DenMedia,2*Dmedia-DenMedia))}
    cat("D=", Dmean, "pD=", Dmean - DinMean, "DIC=", 2 * Dmean - DinMean,
        "\n")
}
```

Poisson-gamma model

Model fitted by empirical Bayes methods.

```r
PoisGamma <- empbaysmooth(Obs.muni, Exp.muni, maxiter = 100)
# Posterior mean of the sSMRs
PoisGamma$nu/PoisGamma$alpha
```

```
## [1] 0.9113144
```

```r
# Posterior sd of the sSMRs
sqrt(PoisGamma$nu/(PoisGamma$alpha^2))
```

```
## [1] 0.2937582
```

```r
# sSMR fitted for this model
PoisGamma.sSMR <- 100 * PoisGamma$smthrr
# Random sample of values of the posterior distribution for calculating
# its DIC and comparing it to alternative models.
PoisGamma.sSMR.sample <- 100 * apply(cbind(Obs.muni + PoisGamma$nu, Exp.muni +
    PoisGamma$alpha), 1, function(x) {
    rgamma(1000, x[1], x[2])
})
# DIC
DICPoisson(PoisGamma.sSMR.sample, Obs.muni, Exp.muni)
```

```
## D= 1691.704 pD= 112.5128 DIC= 1804.217
```

Poisson-logNormal model

```r
# WinBUGS for the Poisson-logNormal model
ModelLogNormal = function() {
    for (i in 1:n) {
        O[i] ~ dpois(lambda[i])
        log(lambda[i]) <- log(E[i]) + log.theta[i]
        log.theta[i] <- mu + het[i]
        het[i] ~ dnorm(0, tau)
        sSMR[i] <- 100 * exp(log.theta[i])
        P.sSMR[i] <- step(sSMR[i] - 100)
    }
    mu ~ dflat()
    tau <- pow(sd.het, -2)
    sd.het ~ dunif(0, 10)
}

# Call to the WinBUGS model above to compute the sSMRs corresponding to
# this model.
data <- list(O = Obs.muni, E = Exp.muni, n = 540)
inits <- function() {
    list(het = rnorm(540), mu = rnorm(1))
}
param <- c("sSMR", "mu", "sd.het")
ResulLN <- pbugs(data = data, inits = inits, param = param, n.iter = 2200,
    n.burnin = 200, model.file = ModelLogNormal, bugs.seed = 1)
# Computing time
ResulLN$exec_time
```

```
## Time difference of 13.46477 secs
```

```r
# Result summaries
summary(ResulLN$summary[, "Rhat"])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.9995  1.0003  1.0013  1.0018  1.0028  1.0133
```

```r
summary(ResulLN$summary[, "n.eff"])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   170.0   720.0  1000.0   854.6  1000.0  1000.0
```

```
round(ResulLN$summary[c("mu", "sd.het"), ], 1)
```

```
##        mean sd 2.5%  25%  50%  75% 97.5% Rhat n.eff
## mu     -0.1  0 -0.2 -0.1 -0.1 -0.1  -0.1    1  1000
## sd.het  0.2  0  0.2  0.2  0.2  0.3   0.3    1  1000
```

```
# DIC
DICPoisson(ResulLN$sims.matrix[, grep("sSMR", dimnames(ResulLN$sims.matrix)[[2]])],
    Obs.muni, Exp.muni)
```

```
## D= 1727.075 pD= 88.0676 DIC= 1815.143
```

```
# smoothed SMRs for Castell de Cabres and Oliva for this model
VR.cart@data[VR.cart$NOMBRE == "Castell de Cabres", ]
```

```
##           AREA PERIMETER MUNI_ MUNI_ID CODMUNI CODPROV CODAUTO CODCOMAR
## 12037 30777133  27462.59  5119    5119   12037      12      12       02
##                  NOMBRE POB91 POB95 POB95M POB95F
## 12037 Castell de Cabres    24    21     11     10
```

```
PoisGamma.sSMR[which(dimnames(PopM)[[1]] == "12037")]
```

```
## [1] 100.3617
```

```
100 * Obs.muni[which(dimnames(PopM)[[1]] == "12037")]/Exp.muni[which(dimnames(PopM)[[1]] ==
    "12037")]
```

```
##    12037
## 3977.445
```

```
VR.cart@data[VR.cart$NOMBRE == "Oliva", ]
```

```
##           AREA PERIMETER MUNI_ MUNI_ID CODMUNI CODPROV CODAUTO CODCOMAR
## 46181 60080946  41274.36  6928    6928   46181      46      12       00
##       NOMBRE POB91 POB95 POB95M POB95F
## 46181  Oliva 20289 20828  10255  10573
```

```
PoisGamma.sSMR[which(dimnames(PopM)[[1]] == "46181")]
```

```
## [1] 174.5226
```

```
100 * Obs.muni[which(dimnames(PopM)[[1]] == "46181")]/Exp.muni[which(dimnames(PopM)[[1]] ==
    "46181")]
```

```
##   46181
## 214.764
```

Alternative INLA fitting of the Poisson-logNormal model

```
# Uniform prior distribution for the standard deviation of the random
# effects
sdunif = "expression:
\tlogdens = -log_precision/2;
\treturn(logdens)"

data = data.frame(O = Obs.muni, E = Exp.muni, id.node = 1:540)
form = O ~ f(id.node, model = "iid", hyper = list(prec = list(prior = sdunif)))
resul.INLA = inla(form, family = "poisson", data = data, E = E, control.compute = list(dic = TRUE))
```

```r
# Computing time
resul.INLA
```

```
##
## Call:
## c("inla(formula = form, family = \"poisson\", data = data, E = E, ",  "    control.compute = list(di
##
## Time used:
##  Pre-processing    Running inla Post-processing           Total
##       1.3918679       1.2177379       0.1219301       2.7315359
##
## Integration Strategy: Model contains 1 hyperparameters
## The model contains 1 fixed effect (including a possible intercept)
##
## Likelihood model: poisson
##
## The model has 1 random effects:
## 1.'id.node' is a IID model
```

```r
# DIC
resul.INLA$dic$dic
```

```
## [1] 1817.63
```

```r
# Fit with strategy='laplace' option
resul.INLA2 = inla(form, family = "poisson", data = data, E = E, control.compute = list(dic = TRUE),
    control.inla = list(strategy = "laplace"))
# Computing time
resul.INLA2
```

```
##
## Call:
## c("inla(formula = form, family = \"poisson\", data = data, E = E, ",  "    control.compute = list(di
##
## Time used:
##  Pre-processing    Running inla Post-processing           Total
##       1.1048229       7.9284542       0.1299088       9.1631858
##
## Integration Strategy: Model contains 1 hyperparameters
## The model contains 1 fixed effect (including a possible intercept)
##
## Likelihood model: poisson
##
## The model has 1 random effects:
## 1.'id.node' is a IID model
```

```r
# DIC
resul.INLA2$dic$dic
```

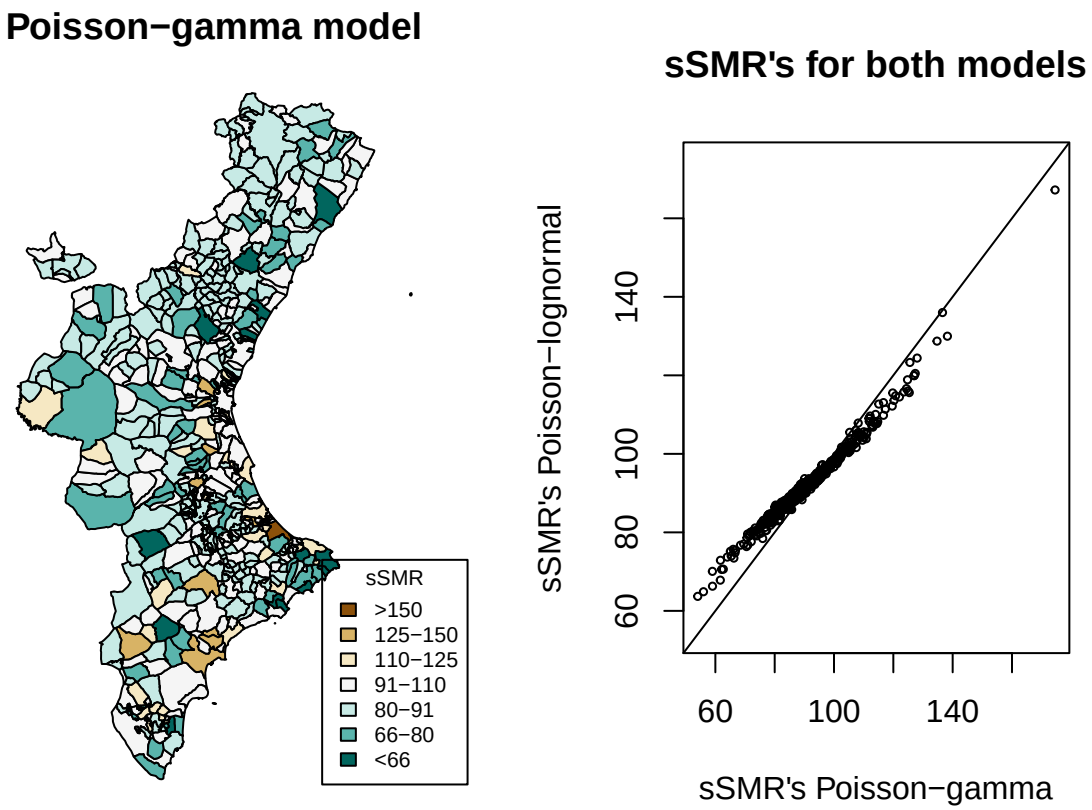
```
## [1] 1817.916
```

```r
# Correlation for the sSMRs of both implementations
cor(ResulLN$mean$sSMR, resul.INLA2$summary.fitted.values[, 1])
```

```
## [1] 0.9977109
```

Comparison between the Poisson-gamma and Poisson-logNormal sSMRs

```
# Figure in the example
colors <- brewer.pal(7, "BrBG")[7:1]
par(mfrow = c(1, 2))
par(mar = c(1, 1, 2, 1) + 0.1)
plot(VR.cart, col = colors[as.numeric(cut(PoisGamma.sSMR, 100 * c(-0.1,
    1/1.5, 1/1.25, 1/1.1, 1.1, 1.25, 1.5, 100)))])
title("Poisson-gamma model", cex = 0.75)
legend(x = "bottomright", fill = colors[7:1], legend = c(">150", "125-150",
    "110-125", "91-110", "80-91", "66-80", "<66"), cex = 0.65, inset = 0.03,
    title = "sSMR")
par(mar = c(5, 4, 4, 2) + 0.1)
plot(PoisGamma.sSMR, ResulLN$mean$sSMR, xlab = "sSMR's Poisson-gamma",
    ylab = "sSMR's Poisson-lognormal", main = "sSMR's for both models",
    cex = 0.5, xlim = c(min(PoisGamma.sSMR), max(PoisGamma.sSMR)), ylim = c(min(PoisGamma.sSMR),
        max(PoisGamma.sSMR)))
abline(c(0, 1))
```




```
# Summaries
sd(PoisGamma.sSMR)
```

```
## [1] 12.88918
```

```
sd(ResulLN$mean$sSMR)
```

```
## [1] 9.6445
```

```r
cor(PoisGamma.sSMR, ResulLN$mean$sSMR)
```

```
## [1] 0.9900888
```