Example 8.1

Disease mapping: from foundations to multidimensional modeling

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

This document reproduces the analysis made at Example 8.1 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

```r
# Libraries loading
#-----------------
if (!require(RColorBrewer)) {
    install.packages("RColorBrewer")
    library(RColorBrewer)
}
if (!require(rgdal)) {
    install.packages("rgdal")
    library(rgdal)
}
if (!require(pbugs)) {
    if (!require(devtools)) {
        install.packages("devtools")
        devtools::install_github("fisabio/pbugs")
    } else {
        install_github("fisabio/pbugs")
    }
}

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

```r
load("../Data/ExpTrivariate.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")
}
```

IND Model: Independent BYM models

```r
# Univariate BYM model WinBUGS code
IND.model = function() {
    for (i in 1:nregions) {
        for (j in 1:ndiseases) {
            Obs[i, j] ~ dpois(lambda[i, j])
            # Modelling of the mealambda=n for every municipality and disease
            log(lambda[i, j]) <- log(Exp[i, j]) + log.theta[i, j]
            log.theta[i, j] <- inter[j] + sd.het[j] * het[i, j] + sd.spat[j] *
                spat[j, i]
            het[i, j] ~ dnorm(0, 1)
            # sSMR for every municipality and period
            sSMR[i, j] <- 100 * exp(log.theta[i, j])
        }
    }

    for (j in 1:ndiseases) {
        spat[j, 1:nregions] ~ car.normal(adj[], w[], num[], 1)
    }
    for (i in 1:nneigh) {
        w[i] <- 1
    }

    # Prior distributions
    for (j in 1:ndiseases) {
        inter[j] ~ dflat()
        sd.het[j] ~ dunif(0, 10)
        sd.spat[j] ~ dunif(0, 10)
    }
}

ndiseases = 3
nregions = length(VR.cart)

data = list(Obs = Obs.mv3, Exp = Exp.mv3, nregions = nregions, ndiseases = ndiseases,
    num = VR.wb$num, adj = VR.wb$adj, nneigh = length(VR.wb$adj))
inits = function() {
    list(inter = rnorm(ndiseases), sd.het = runif(ndiseases), sd.spat = runif(ndiseases),
        het = matrix(rnorm(data$nregions * data$ndiseases), ncol = ndiseases),
        spat = matrix(rnorm(data$nregions * data$ndiseases), nrow = ndiseases))
}
param = c("inter", "sd.het", "sd.spat", "sSMR")

IND.resul = pbugs(data = data, inits = inits, parameters = param, model.file = IND.model,
    n.iter = 10000, n.burnin = 2000, DIC = F, bugs.seed = 1)

# Computing time
IND.resul$exec_time
```

```
## Time difference of 3.54286 mins
```

```
# Result summaries
summary(IND.resul$summary[, "Rhat"])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.9995  1.0004  1.0015  1.0020  1.0029  1.0150
```

```
summary(IND.resul$summary[, "n.eff"])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   130.0   670.0  1000.0   838.9  1000.0  1000.0
```

MV1 model: MCAR model with fully independent heterogenous random effects

```
MV1.model = function() {
    for (i in 1:nregions) {
        for (j in 1:ndiseases) {
            Obs[i, j] ~ dpois(lambda[i, j])
            # Modelling of the mean for every municipality and disease
            log(lambda[i, j]) <- log(Exp[i, j]) + log.theta[i, j]
            log.theta[i, j] <- inter[j] + sd.het[j] * het[i, j] + spat[j,
                i]
            het[i, j] ~ dnorm(0, 1)
            # sSMR for every municipality and period
            sSMR[i, j] <- 100 * exp(log.theta[i, j])
        }
    }

    spat[1:ndiseases, 1:nregions] ~ mv.car(adj[], w[], num[], omega.spat[,
        ])
    omega.spat[1:ndiseases, 1:ndiseases] ~ dwish(R.spat[, ], ndiseases)
    sigma.spat[1:ndiseases, 1:ndiseases] <- inverse(omega.spat[, ])
    for (i in 1:nneigh) {
        w[i] <- 1
    }

    # Prior distributions
    for (j in 1:ndiseases) {
        inter[j] ~ dflat()
        sd.het[j] ~ dunif(0, 5)
    }
}

R.spat = matrix(diag(rep(1, ndiseases)), nrow = ndiseases)

data = list(Obs = Obs.mv3, Exp = Exp.mv3, nregions = nregions, ndiseases = ndiseases,
    num = VR.wb$num, adj = VR.wb$adj, nneigh = length(VR.wb$adj), R.spat = R.spat)
inits = function() {
    list(inter = rnorm(ndiseases), sd.het = runif(ndiseases), het = matrix(rnorm(data$nregions *
        data$ndiseases), ncol = ndiseases), spat = matrix(rnorm(data$nregions *
        data$ndiseases), nrow = ndiseases))
}
param = c("inter", "sd.het", "sigma.spat", "sSMR")

MV1.1.resul = pbugs(data = data, inits = inits, parameters = param, model.file = MV1.model,
    n.iter = 10000, n.burnin = 2000, DIC = F, bugs.seed = 1)

# Computing time
MV1.1.resul$exec_time
```

```
## Time difference of 2.768515 mins
```

```
# Result summaries
summary(MV1.1.resul$summary[, "Rhat"])
```

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

```
summary(MV1.1.resul$summary[, "n.eff"])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   160.0   750.0  1000.0   857.7  1000.0  1000.0
```

```
R.spat = matrix(diag(rep(1, ndiseases)), nrow = ndiseases)/100

data = list(Obs = Obs.mv3, Exp = Exp.mv3, nregions = nregions, ndiseases = ndiseases,
    num = VR.wb$num, adj = VR.wb$adj, nneigh = length(VR.wb$adj), R.spat = R.spat)

MV1.01.resul = pbugs(data = data, inits = inits, parameters = param, model.file = MV1.model,
    n.iter = 25000, n.burnin = 5000, DIC = F, bugs.seed = 1)
```

```r
# Computing time
MV1.01.resul$exec_time
```

```
## Time difference of 6.928261 mins
```

```r
# Result summaries
summary(MV1.01.resul$summary[, "Rhat"])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##  0.9995  1.0007  1.0021  1.0031  1.0042  1.0809
```

```r
summary(MV1.01.resul$summary[, "n.eff"])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##    32.0   480.0   950.0   750.4  1000.0  1000.0
```

MV2 model: MCAR model with heterogenous random effects dependent accounting for dependence between diseases

```r
MV2.model = function() {
    for (i in 1:nregions) {
        for (j in 1:ndiseases) {
            Obs[i, j] ~ dpois(lambda[i, j])
            # Modelling of the mean for every municipality and disease
            log(lambda[i, j]) <- log(Exp[i, j]) + log.theta[i, j]
            log.theta[i, j] <- inter[j] + het[i, j] + spat[j, i]
            # sSMR for every municipality and period
            sSMR[i, j] <- 100 * exp(log.theta[i, j])
        }
        het[i, 1:ndiseases] ~ dmnorm(v0[], omega.het[, ])
    }
    for (j in 1:ndiseases) {
        v0[j] <- 0
    }

    spat[1:ndiseases, 1:nregions] ~ mv.car(adj[], w[], num[], omega.spat[,
        ])
    for (i in 1:nneigh) {
        w[i] <- 1
    }

    # Prior distributions
    for (j in 1:ndiseases) {
        inter[j] ~ dflat()
    }
    omega.het[1:ndiseases, 1:ndiseases] ~ dwish(R.het[, ], ndiseases)
    sigma.het[1:ndiseases, 1:ndiseases] <- inverse(omega.het[, ])
    omega.spat[1:ndiseases, 1:ndiseases] ~ dwish(R.spat[, ], ndiseases)
    sigma.spat[1:ndiseases, 1:ndiseases] <- inverse(omega.spat[, ])
}

R.spat = matrix(diag(rep(1, ndiseases))/100, nrow = ndiseases)
R.het = matrix(diag(rep(1, ndiseases))/100, nrow = ndiseases)
```

```
data = list(Obs = Obs.mv3, Exp = Exp.mv3, nregions = nregions, ndiseases = ndiseases,
    num = VR.wb$num, adj = VR.wb$adj, nneigh = length(VR.wb$adj), R.spat = R.spat,
    R.het = R.het)
inits = function() {
    list(inter = rnorm(ndiseases), het = matrix(rnorm(data$nregions * data$ndiseases),
        ncol = ndiseases), spat = matrix(rnorm(data$nregions * data$ndiseases),
        nrow = ndiseases))
}
param = c("inter", "sigma.het", "sigma.spat", "sSMR")

MV2.01.resul = pbugs(data = data, inits = inits, parameters = param, model.file = MV2.model,
    n.iter = 25000, n.burnin = 5000, DIC = F, bugs.seed = 1)

# Computing time
MV2.01.resul$exec_time
```

```
## Time difference of 9.016497 mins
```

```
# Result summaries
summary(MV2.01.resul$summary[, "Rhat"])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.9995  1.0006  1.0018  1.0027  1.0039  1.0261
```

```
summary(MV2.01.resul$summary[, "n.eff"])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    96.0   570.0  1000.0   794.7  1000.0  1000.0
```

```
MV2.cond.model = function() {
    for (i in 1:nregions) {
        for (j in 1:ndiseases) {
            Obs[i, j] ~ dpois(lambda[i, j])
            # Modelling of the mean for every municipality and disease
            log(lambda[i, j]) <- log(Exp[i, j]) + log.theta[i, j]
            log.theta[i, j] <- inter[j] + het[i, j] + spat[i, j]
            # sSMR for every municipality and period
            sSMR[i, j] <- 100 * exp(log.theta[i, j])
        }
        het[i, 1:ndiseases] ~ dmnorm(v0[], omega.het[, ])
        # Spatial process defined as a multivariate Normal distribution given
        # the neighbours
        spat[i, 1:ndiseases] ~ dmnorm(mu.spat[i, ], omega.spat.aux[i, ,
            ])
        for (j in 1:ndiseases) {
            mu.spat[i, j] <- sum(spat.adj[(index[i] + 1):index[i + 1],
                j])/(num[i])
            for (k in 1:ndiseases) {
                omega.spat.aux[i, j, k] <- num[i] * omega.spat[j, k]
            }
        }
    }
    for (i in 1:ndiseases) {
        v0[i] <- 0
```

```r
    }
    for (i in 1:nneigh) {
        for (j in 1:ndiseases) {
            spat.adj[i, j] <- spat[adj[i], j]
        }
    }

    # Prior distributions
    for (j in 1:ndiseases) {
        inter[j] ~ dflat()
    }
    omega.spat[1:ndiseases, 1:ndiseases] ~ dwish(R.spat[, ], ndiseases)
    sigma.spat[1:ndiseases, 1:ndiseases] <- inverse(omega.spat[, ])
    omega.het[1:ndiseases, 1:ndiseases] ~ dwish(R.het[, ], ndiseases)
    sigma.het[1:ndiseases, 1:ndiseases] <- inverse(omega.het[, ])
}

R.spat = matrix(diag(rep(1, ndiseases))/100, nrow = ndiseases)
R.het = matrix(diag(rep(1, ndiseases))/100, nrow = ndiseases)
index = c(0, cumsum(VR.wb$num))

data = list(Obs = Obs.mv3, Exp = Exp.mv3, nregions = nregions, ndiseases = ndiseases,
    num = VR.wb$num, adj = VR.wb$adj, nneigh = length(VR.wb$adj), R.spat = R.spat,
    R.het = R.het, index = index)
inits = function() {
    list(inter = rnorm(ndiseases), het = matrix(rnorm(data$nregions * data$ndiseases),
        ncol = ndiseases), spat = matrix(rnorm(data$nregions * data$ndiseases),
        ncol = ndiseases))
}
param = c("inter", "sigma.het", "sigma.spat", "sSMR")

MV2.cond.01.resul = pbugs(data = data, inits = inits, parameters = param,
    model.file = MV2.cond.model, n.iter = 1e+05, n.burnin = 20000, DIC = F,
    bugs.seed = 1)

# Computing time
MV2.cond.01.resul$exec_time
```

```
## Time difference of 58.6249 mins
```

```r
# Result summaries
summary(MV2.cond.01.resul$summary[, "Rhat"])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.9995  1.0006  1.0020  1.0096  1.0041  6.7145
```

```r
summary(MV2.cond.01.resul$summary[, "n.eff"])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     3.0   520.0  1000.0   768.1  1000.0  1000.0
```

Some summary statistics for the MV1.01, MV1.1 and MV2.01 models

```r
# Maximum sSMR
max(MV1.01.resul$mean$sSMR)
```

```
## [1] 179.3599
```

```r
# Standard deviation of the log-sSMRs
sd(log(MV1.01.resul$mean$sSMR))
```

```
## [1] 0.2622893
```

```r
# Covariance matrix between diseases (posterior mean)
MV1.01.resul$mean$sigma.spat
```

```
##            [,1]       [,2]      [,3]
## [1,] 0.07512141 0.09542026 0.1227508
## [2,] 0.09542026 0.14378013 0.1659365
## [3,] 0.12275078 0.16593649 0.2226462
```

```r
# Correlation matrix between diseases (posterior mean) for the MV1.01
# model
cov2cor(MV1.01.resul$mean$sigma.spat)
```

```
##           [,1]      [,2]      [,3]
## [1,] 1.0000000 0.9181408 0.9491498
## [2,] 0.9181408 1.0000000 0.9274378
## [3,] 0.9491498 0.9274378 1.0000000
```

```r
# Correlation matrix between diseases (posterior mean) for the MV1.1
# model
cov2cor(MV1.1.resul$mean$sigma.spat)
```

```
##           [,1]      [,2]      [,3]
## [1,] 1.0000000 0.5864766 0.6587472
## [2,] 0.5864766 1.0000000 0.7503973
## [3,] 0.6587472 0.7503973 1.0000000
```

```r
# Correlation matrix between diseases (posterior mean) for the MV2.01
# model. Spatial component.
cov2cor(MV2.01.resul$mean$sigma.spat)
```

```
##           [,1]      [,2]      [,3]
## [1,] 1.0000000 0.8921104 0.9195808
## [2,] 0.8921104 1.0000000 0.9206488
## [3,] 0.9195808 0.9206488 1.0000000
```

```r
# Correlation matrix between diseases (posterior mean) for the MV2.01
# model. Heterogenous component.
cov2cor(MV2.01.resul$mean$sigma.het)
```

```
##           [,1]      [,2]      [,3]
## [1,] 1.0000000 0.4667863 0.6268464
## [2,] 0.4667863 1.0000000 0.5722528
## [3,] 0.6268464 0.5722528 1.0000000
```

DICs for the models fitted

```r
# IND
DICPoisson(IND.resul$sims.matrix[, grep("sSMR", dimnames(IND.resul$sims.matrix)[[2]])],
    t(Obs.mv3), t(Exp.mv3))
```

```
## D= 6977.274 pD= 463.6162 DIC= 7440.891
```

```r
# MV1.01
DICPoisson(MV1.01.resul$sims.matrix[, grep("sSMR", dimnames(MV1.01.resul$sims.matrix)[[2]])],
    t(Obs.mv3), t(Exp.mv3))
```

```
## D= 6963.872 pD= 396.2621 DIC= 7360.134
```

```r
# MV1.1
DICPoisson(MV1.1.resul$sims.matrix[, grep("sSMR", dimnames(MV1.1.resul$sims.matrix)[[2]])],
    t(Obs.mv3), t(Exp.mv3))
```

```
## D= 6927.473 pD= 443.2164 DIC= 7370.689
```

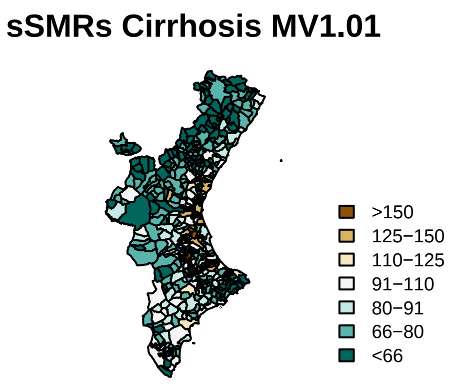
```r
# MV2.01
DICPoisson(MV2.01.resul$sims.matrix[, grep("sSMR", dimnames(MV2.01.resul$sims.matrix)[[2]])],
    t(Obs.mv3), t(Exp.mv3))
```

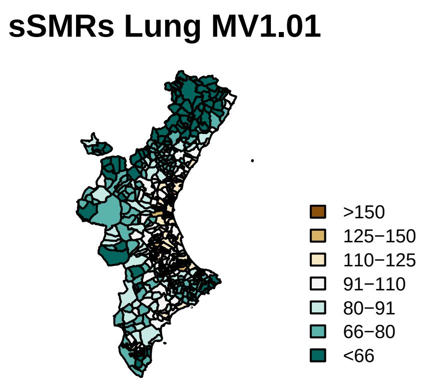
```
## D= 6961.202 pD= 397.3123 DIC= 7358.515
```

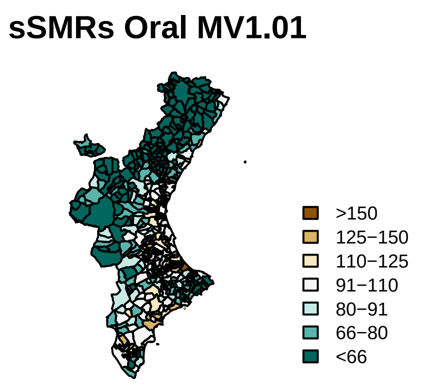
Choropleth maps

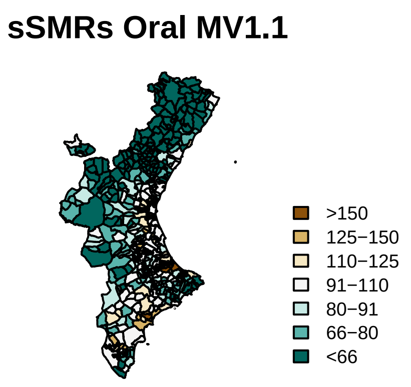
```r
colors = brewer.pal(7, "BrBG")[7:1]
cuts = 100 * c(-0.1, 1/1.5, 1/1.25, 1/1.1, 1.1, 1.25, 1.5, 100)

par(mfrow = c(2, 2))
par(mar = c(1, 1, 2, 1) + 0.1)

sSMR.cut = as.numeric(cut(MV1.01.resul$mean$sSMR[, 1], cuts))
label.model = "MV1.01"
plot(VR.cart, col = colors[sSMR.cut])
title(paste("sSMRs Cirrhosis", label.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 = 0.75, inset = 0.03,
    bty = "n")

sSMR.cut = as.numeric(cut(MV1.01.resul$mean$sSMR[, 2], cuts))
plot(VR.cart, col = colors[sSMR.cut])
title(paste("sSMRs Lung", label.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 = 0.75, inset = 0.03,
    bty = "n")

sSMR.cut = as.numeric(cut(MV1.01.resul$mean$sSMR[, 3], cuts))
plot(VR.cart, col = colors[sSMR.cut])
title(paste("sSMRs Oral", label.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 = 0.75, inset = 0.03,
    bty = "n")
```

```r
sSMR.cut = as.numeric(cut(MV1.1.resul$mean$sSMR[, 3], cuts))
label.model <- "MV1.1"
plot(VR.cart, col = colors[sSMR.cut])
title(paste("sSMRs Oral", label.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 = 0.75, inset = 0.03,
    bty = "n")
```


sSMRs Cirrhosis MV1.01
>150
125–150
110–125
91–110
80–91
66–80
<66


sSMRs Lung MV1.01
>150
125–150
110–125
91–110
80–91
66–80
<66


sSMRs Oral MV1.01
>150
125–150
110–125
91–110
80–91
66–80
<66


sSMRs Oral MV1.1
>150
125–150
110–125
91–110
80–91
66–80
<66