Example 8.2

Disease mapping: from foundations to multidimensional modeling

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

This document reproduces the analysis made at Example 8.2 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)
}
# For generating random samples for multivariate Normal distributions
# (required for sampling from a PCAR distribution).
if (!require(MASS)) {
    install.packages("MASS")
    library(MASS)
}
if (!require(corrplot)) {
    install.packages("corrplot")
    library(corrplot)
}
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')
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.

```
# 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")
}
```

Function monitoring the deviance convergence in multivariate models

```
plot.deviance <- function(resul, Obs, Exp) {
    deviances <- matrix(nrow = dim(resul$sims.array)[1], ncol = dim(resul$sims.array)[2])
    for (j in 1:(dim(resul$sims.array)[2])) {
        for (i in 1:(dim(resul$sims.array)[1])) {
            sSMR <- matrix(resul$sims.array[i, j, grep("sSMR", dimnames(resul$summary)[[1]])],
                nrow = nrow(Obs), byrow = T)
            deviances[i, j] <- -2 * sum(dpois(as.vector(Obs), as.vector((sSMR/100) *
                Exp), log = T))
        }
    }
    plot(deviances[, 1], type = "l", ylim = c(min(deviances), max(deviances)),
        ylab = "Deviance", xlab = "Iteration saved")
    for (j in 2:(dim(resul$sims.array)[2])) {
        lines(deviances[, j], col = j)
    }
}
```

BYM M-model

```
Mmodel.BYM = function() {
    # Likelihood
    for (i in 1:Nareas) {
        for (j in 1:Ndiseases) {
            Obs[i, j] ~ dpois(lambda[i, j])
            log(lambda[i, j]) <- log(Exp[i, j]) + log.theta[i, j]
            log.theta[i, j] <- alpha[j] + S[i, j]
            S[i, j] <- inprod2(Delta[i, ], M.mat[, j])

            sSMR[i, j] <- 100 * exp(log.theta[i, j])
        }
    }

    # Spatial underlying patterns
    for (j in 1:Ndiseases) {
        Spatial[j, 1:Nareas] ~ car.normal(adj[], weights[], num[], 1)
        for (i in 1:Nareas) {
            Het[i, j] ~ dnorm(0, 1)
            Delta[i, j] <- Spatial[j, i]
        }
    }
    for (i in 1:nneigh) {
        weights[i] <- 1
    }

    # Heterogenous underlying patterns
    for (j in (Ndiseases + 1):(2 * Ndiseases)) {
        for (i in 1:Nareas) {
            Delta[i, j] <- Het[i, (j - Ndiseases)]
        }
    }

    # M matrix
    for (i in 1:(2 * Ndiseases)) {
        for (j in 1:Ndiseases) {
            M.mat[i, j] ~ dnorm(0, prec[i])
        }
        prec[i] <- pow(sdstruct[i], -2)
        sdstruct[i] ~ dunif(0, 10)
    }

    # Prior distributions
    for (j in 1:Ndiseases) {
        alpha[j] ~ dflat()
    }
}

nregions = length(VR.cart)
ndiseases = ncol(Obs.mv3)

data = list(Obs = Obs.mv3, Exp = Exp.mv3, adj = VR.wb$adj, num = VR.wb$num,
    nneigh = length(VR.wb$adj), Nareas = nregions, Ndiseases = ndiseases)
inits = function() {
    list(M.mat = matrix(rnorm(ndiseases * ndiseases * 2, 0, 1), nrow = ndiseases *
        2), Spatial = matrix(rnorm(nregions * ndiseases), nrow = ndiseases),
        Het = matrix(rnorm(nregions * ndiseases), ncol = ndiseases), alpha = runif(ndiseases,
            -0.5, 0), sdstruct = runif(2 * ndiseases, 0, 1))
}
param = c("alpha", "sSMR", "M.mat", "sdstruct")

ResM.BYM = pbugs(data = data, inits = inits, par = param, model = Mmodel.BYM,
    n.iter = 10000, n.burnin = 2000, DIC = FALSE, bugs.seed = 1)
```

Some results for the BYM M-model of interest for Example 8.2

```
# Computing time
ResM.BYM$exec_time
```

```
## Time difference of 9.451913 mins
```

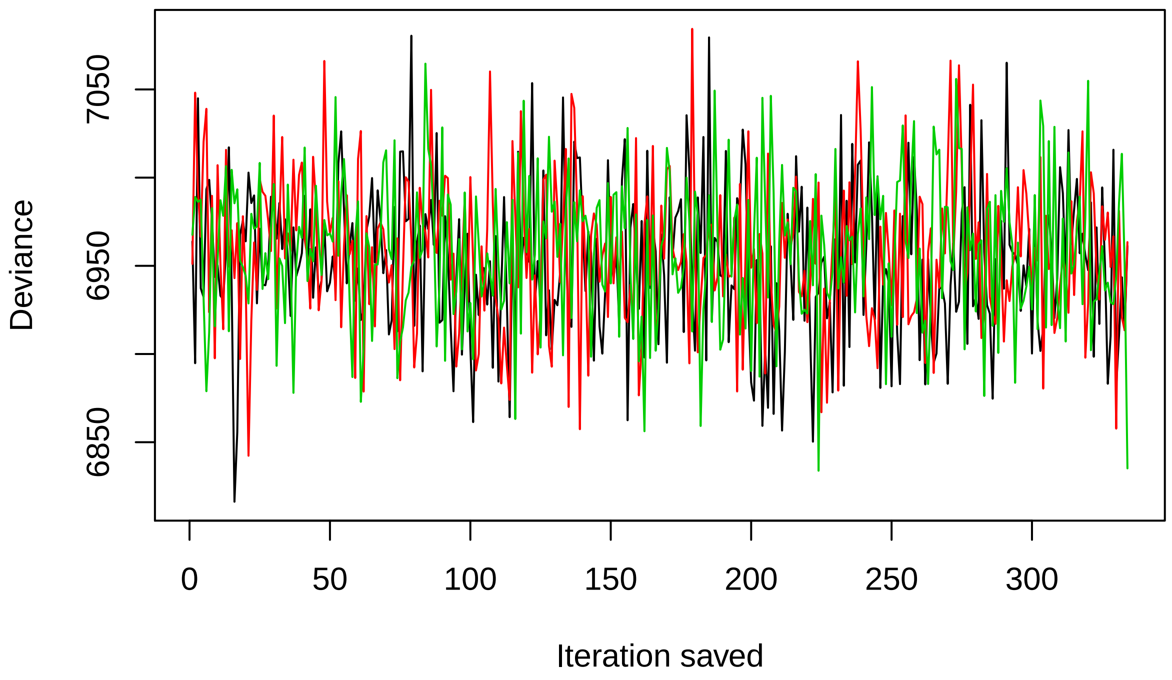
```
# Result summaries for identifiable parameters
aux1 <- grep("alpha", dimnames(ResM.BYM$summary)[[1]])
aux2 <- grep("sSMR", dimnames(ResM.BYM$summary)[[1]])
aux <- c(aux1, aux2)

summary(ResM.BYM$summary[aux, "Rhat"])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.9995  1.0003  1.0014  1.0020  1.0030  1.0143
```

```
summary(ResM.BYM$summary[aux, "n.eff"])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   140.0   680.0  1000.0   835.5  1000.0  1000.0
```

```
plot.deviance(ResM.BYM, Obs.mv3, Exp.mv3)
```



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

```
## D= 6955.779 pD= 403.7926 DIC= 7359.571
```

```r
# Variance-covariance and correlation matrices
nsimu = dim(ResM.BYM$sims.list$M.mat)[1]
matcov.spat = array(dim = c(nsimu, 3, 3))
matcor.spat = array(dim = c(nsimu, 3, 3))
matcov.het = array(dim = c(nsimu, 3, 3))
matcor.het = array(dim = c(nsimu, 3, 3))
for (i in 1:nsimu) {
    matcov.spat[i, , ] = t(ResM.BYM$sims.list$M.mat[i, 1:3, ]) %*% (ResM.BYM$sims.list$M.mat[i,
        1:3, ])
    matcor.spat[i, , ] = cov2cor(matcov.spat[i, , ])
    matcov.het[i, , ] = t(ResM.BYM$sims.list$M.mat[i, 4:6, ]) %*% (ResM.BYM$sims.list$M.mat[i,
        4:6, ])
    matcor.het[i, , ] = cov2cor(matcov.het[i, , ])
}
matcov.spat.mean = apply(matcov.spat, c(2, 3), mean)
matcor.spat.mean = apply(matcor.spat, c(2, 3), mean)
matcov.spat.mean
```

```
##            [,1]       [,2]      [,3]
## [1,] 0.06424436 0.07842037 0.1012726
## [2,] 0.07842037 0.13474327 0.1470649
## [3,] 0.10127261 0.14706489 0.2035109
```

```
matcor.spat.mean
```

```
##           [,1]      [,2]      [,3]
## [1,] 1.0000000 0.8588654 0.8979637
## [2,] 0.8588654 1.0000000 0.8923681
## [3,] 0.8979637 0.8923681 1.0000000
```

```
matcov.het.mean = apply(matcov.het, c(2, 3), mean)
matcor.het.mean = apply(matcor.het, c(2, 3), mean)
matcov.het.mean
```

```
##             [,1]        [,2]        [,3]
## [1,] 0.059539526 0.007125973 0.016001328
## [2,] 0.007125973 0.003481779 0.005438116
## [3,] 0.016001328 0.005438116 0.012786972
```

```
matcor.het.mean
```

```
##           [,1]      [,2]      [,3]
## [1,] 1.0000000 0.5496897 0.6361714
## [2,] 0.5496897 1.0000000 0.7737816
## [3,] 0.6361714 0.7737816 1.0000000
```

QR model

```
QRmodel = function() {
    # Likelihood
    for (i in 1:Nareas) {
        for (j in 1:Ndiseases) {
            Obs[i, j] ~ dpois(lambda[i, j])
            log(lambda[i, j]) <- log(Exp[i, j]) + log.theta[i, j]
            log.theta[i, j] <- alpha[j] + S[i, j]
            S[i, j] <- inprod2(tDelta[, i], tCholDisRotated[, j])
            sSMR[i, j] <- 100 * exp(alpha[j] + S[i, j])
        }
    }

    # Underlying (pCAR) spatial processes
    for (j in 1:Ndiseases) {
        tDelta[j, 1:Nareas] ~ car.proper(ceros[], C[], adj[], num[], M[],
            1, gamma.ord[j])
        order[j] <- Ndiseases + 1 - j
        gamma.ord[j] <- ranked(gamma[], order[j])
        gamma[j] ~ dunif(gamma.inf, gamma.sup)
    }
    for (i in 1:Nareas) {
        ceros[i] <- 0
    }
    gamma.inf <- min.bound(C[], adj[], num[], M[])
    gamma.sup <- max.bound(C[], adj[], num[], M[])

    # Rotation of CholDis (Cholesky decomposition for the covariance matrix
    # between diseases) by means of an orthogonal transformation.
    # P=Orthogonal matrix.
    P[1, 1] <- cos(theta12) * cos(theta13)
    P[1, 2] <- sin(theta12) * cos(theta23) - cos(theta12) * sin(theta13) *
        sin(theta23)
    P[1, 3] <- sin(theta12) * sin(theta23) + cos(theta12) * sin(theta23) *
        cos(theta23)
    P[2, 1] <- -sin(theta12) * cos(theta13)
    P[2, 2] <- cos(theta12) * cos(theta23) + sin(theta12) * sin(theta13) *
        sin(theta23)
    P[2, 3] <- cos(theta12) * sin(theta23) - sin(theta12) * sin(theta13) *
        cos(theta23)
    P[3, 1] <- -sin(theta13)
    P[3, 2] <- -cos(theta13) * sin(theta23)
    P[3, 3] <- cos(theta13) * cos(theta23)
    for (i in 1:3) {
        for (j in 1:3) {
            tCholDisRotated[j, i] <- inprod2(CholDis[i, 1:i], P[1:i, j])
        }
    }
    # Prior distribution for the Givens rotations generating the P matrix
    theta12 ~ dunif(0, 1.5708)
    theta13 ~ dunif(0, 1.5708)
    theta23 ~ dunif(0, 1.5708)

    # Between-diseases covariance structure: CholDis=Cholesky triangle of a
    # general correlation matrix. diagonal cells
    CholDis[1, 1] <- sigma[1]
    for (j in 2:Ndiseases) {
        diag[j] <- pow(sigma[j], 2) - inprod2(CholDis[j, 1:(j - 1)], CholDis[j,
            1:(j - 1)])
        CholDis[j, j] <- sqrt(abs(diag[j]))
    }

    # Lower triangle First column
    for (i in 2:Ndiseases) {
        CholDis[i, 1] <- sigma[i] * roDis[i, 1]
    }
    # Rest of columns
    for (j in 2:(Ndiseases - 1)) {
        for (i in (j + 1):Ndiseases) {
            CholDis[i, j] <- (roDis[i, j] * sigma[i] * sigma[j] - inprod2(CholDis[i,
                1:(j - 1)], CholDis[j, 1:(j - 1)]))/CholDis[j, j]
        }
    }

    # Correlation matrix between diseases
    for (j in 1:Ndiseases) {
        roDis[1, j] <- 0
    }
    for (i in 2:Ndiseases) {
        for (j in 1:(i - 1)) {
            roDis[i, j] ~ dunif(-1, 1)
        }
```

```
        for (j in i:Ndiseases) {
            roDis[i, j] <- 0
        }
    }

    # Posite definiteness of the covariance matrix
    one <- 1
    one ~ dbern(condition)
    condition <- step(sum(subcondition[2:Ndiseases]) - (Ndiseases - 1))
    for (j in 2:Ndiseases) {
        subcondition[j] <- step(diag[j])
    }

    # Other priors
    for (k in 1:Ndiseases) {
        alpha[k] ~ dflat()
        sigma[k] ~ dunif(0, 10)
    }
}

data = list(Obs = Obs.mv3, Exp = Exp.mv3, adj = VR.wb$adj, num = VR.wb$num,
    C = rep(1/VR.wb$num, VR.wb$num), M = 1/VR.wb$num, Nareas = nregions,
    Ndiseases = ndiseases)
inits = function() {
    list(tDelta = matrix(rnorm(nregions * ndiseases), nrow = ndiseases),
        sigma = runif(ndiseases, 0, 1), alpha = runif(ndiseases, -0.5,
            0), roDis = matrix(c(rep(NA, 3), runif(1), rep(NA, 2), runif(2),
            NA), ncol = 3, byrow = T))
}
param = c("alpha", "sigma", "theta12", "theta13", "theta23", "gamma", "gamma.ord",
    "roDis", "sSMR")

ResQR = pbugs(data = data, inits = inits, par = param, model = QRmodel,
    n.iter = 30000, n.burnin = 5000, DIC = F, bugs.seed = 1)
```

```
# Computing time
ResQR$exec_time
```

```
## Time difference of 2.851602 hours
```

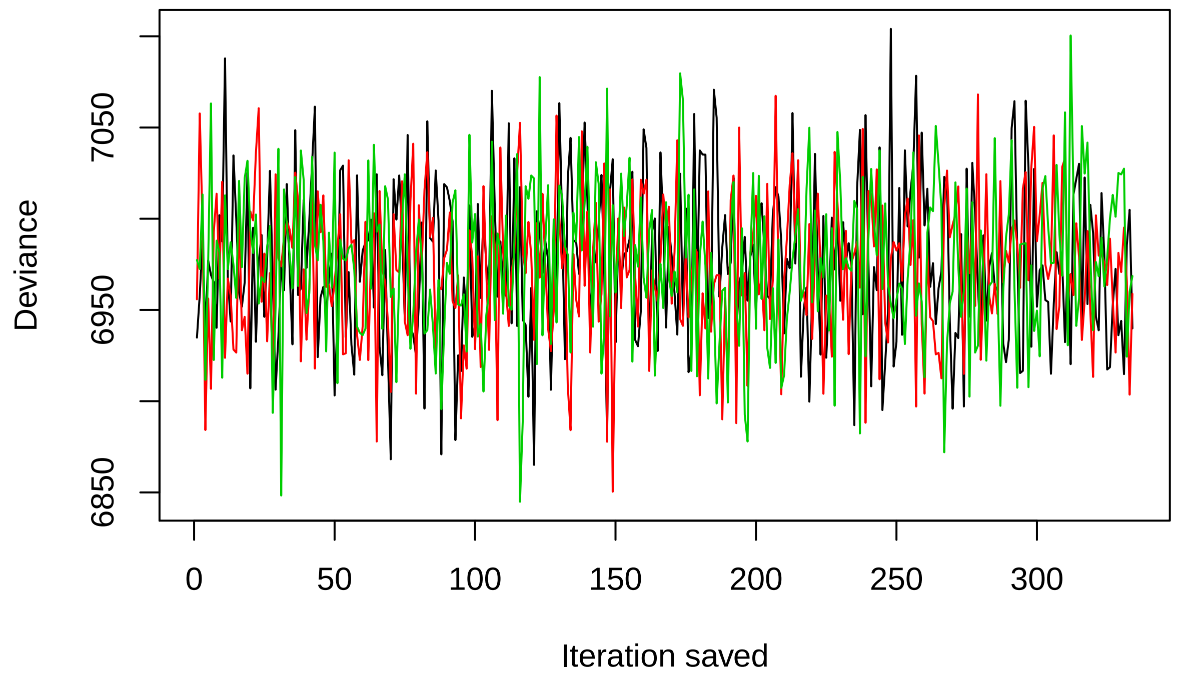
```
# Result summaries
summary(ResQR$summary[, "Rhat"])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.9995  1.0004  1.0013  1.0087  1.0030  4.4290
```

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

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     3.0   710.0  1000.0   841.4  1000.0  1000.0
```

```
plot.deviance(ResQR, Obs.mv3, Exp.mv3)
```



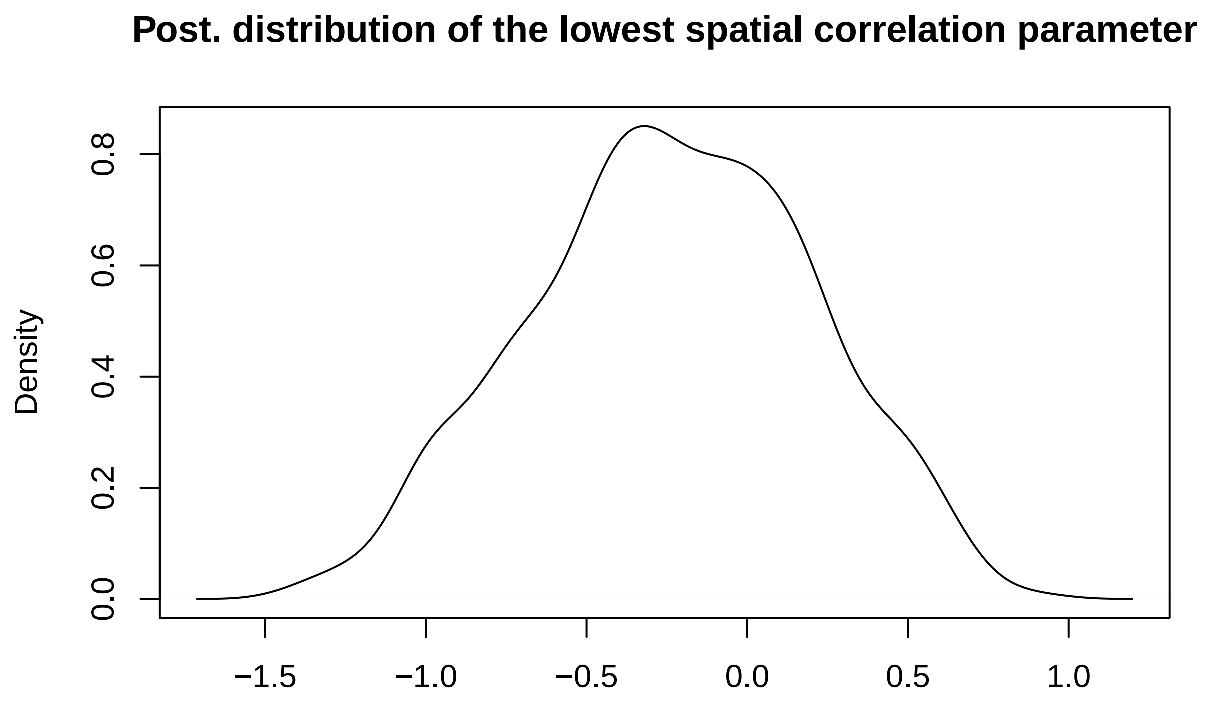
```
# DIC
DICPoisson(ResQR$sims.matrix[, grep("sSMR", dimnames(ResQR$sims.matrix)[[2]])],
    t(Obs.mv3), t(Exp.mv3))
```

```
## D= 6974.966 pD= 409.074 DIC= 7384.04
```

```
# Posterior mean of the correlation parameters of each underlying
# pattern
ResQR$mean$gamma.ord
```

```
## [1]  0.9975441  0.9933360 -0.2350018
```

```
# Posterior density plot for the third of these parameters
plot(density(ResQR$sims.list$gamma.ord[, 3]), main = "Post. distribution of the lowest spatial correlat
    xlab = "")
```



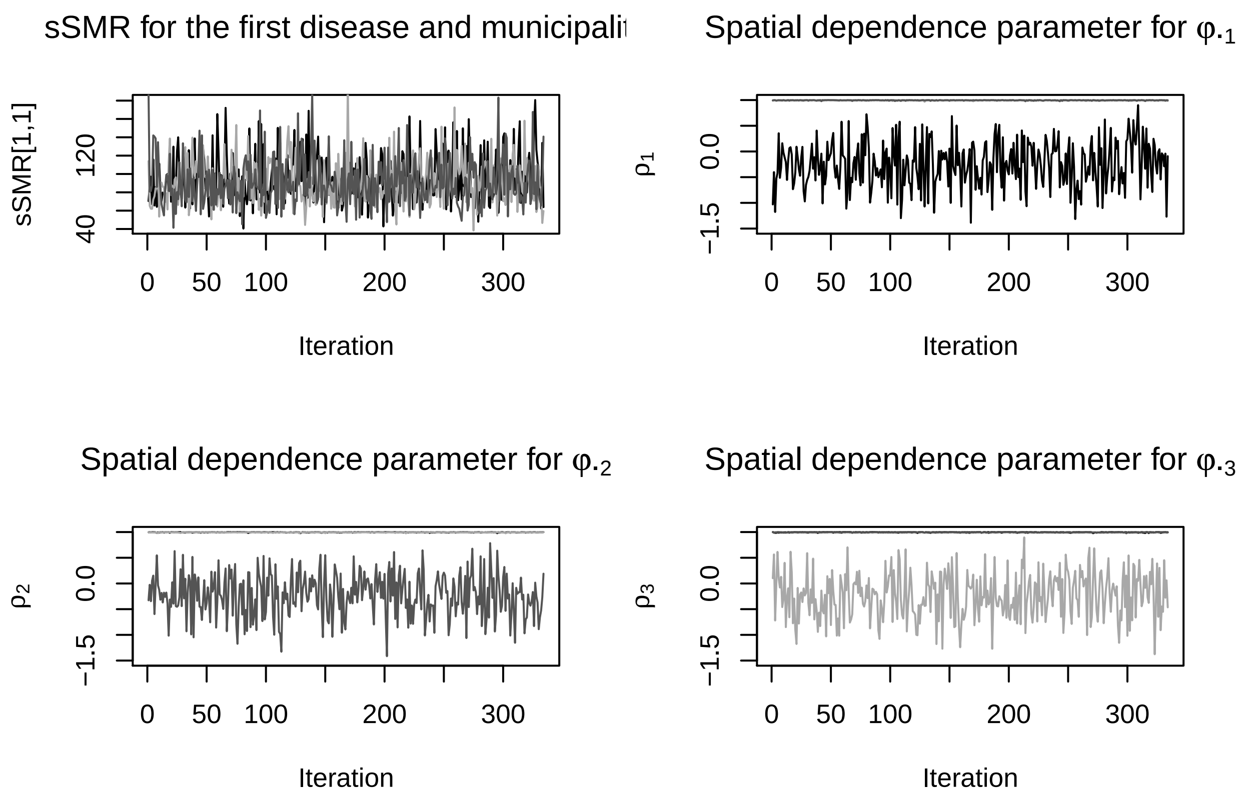
```r
# Correlation matrix for the log-sSMRs for the different diseases
cors.iter <- apply(log(ResQR$sims.list$sSMR), 1, function(x) {
    cor(x)
})
matrix(apply(cors.iter, 1, mean), ncol = 3)
```

```
##           [,1]      [,2]      [,3]
## [1,] 1.0000000 0.5934114 0.6266221
## [2,] 0.5934114 1.0000000 0.8814423
## [3,] 0.6266221 0.8814423 1.0000000
```

Convergence assessment (Figure 8.2 in the example)

```r
par(mfrow = c(2, 2))
plot(ResQR$sims.array[, 1, "sSMR[1,1]"], type = "l", ylab = "sSMR[1,1]",
    xlab = "Iteration", main = expression("sSMR for the first disease and municipality"))
lines(ResQR$sims.array[, 2, "sSMR[1,1]"], col = gray(0.66))
lines(ResQR$sims.array[, 3, "sSMR[1,1]"], col = gray(0.33))

plot(ResQR$sims.array[, 1, "gamma[1]"], type = "l", ylim = c(-1.5, 1),
    ylab = expression(rho[1]), xlab = "Iteration", main = expression(paste("Spatial dependence paramete
        varphi, "."[1])))
lines(ResQR$sims.array[, 2, "gamma[1]"], col = gray(0.66))
lines(ResQR$sims.array[, 3, "gamma[1]"], col = gray(0.33))
```

```r
plot(ResQR$sims.array[, 1, "gamma[2]"], type = "l", ylim = c(-1.5, 1),
    ylab = expression(rho[2]), xlab = "Iteration", main = expression(paste("Spatial dependence paramete
        varphi, "."[2])))
lines(ResQR$sims.array[, 2, "gamma[2]"], col = gray(0.66))
lines(ResQR$sims.array[, 3, "gamma[2]"], col = gray(0.33))

plot(ResQR$sims.array[, 1, "gamma[3]"], type = "l", ylim = c(-1.5, 1),
    ylab = expression(rho[3]), xlab = "Iteration", main = expression(paste("Spatial dependence paramete
        varphi, "."[3])))
lines(ResQR$sims.array[, 2, "gamma[3]"], col = gray(0.66))
lines(ResQR$sims.array[, 3, "gamma[3]"], col = gray(0.33))
```



RVA M-model

```r
Mmodel.RVA <- function() {
    # Likelihood
    for (i in 1:Nareas) {
        for (j in 1:Ndiseases) {
            Obs[i, j] ~ dpois(lambda[i, j])
            log(lambda[i, j]) <- log(Exp[i, j]) + log.theta[i, j]
            log.theta[i, j] <- alpha[j] + S[i, j]
            S[i, j] <- inprod2(tDelta[, i], M.mat[, j])
            sSMR[i, j] <- 100 * exp(alpha[j] + S[i, j])
        }
    }

    # Underlying (pCAR) spatial processes
    for (j in 1:Ndiseases) {
        tDelta[j, 1:Nareas] ~ car.proper(ceros[], C[], adj[], num[], M[],
            1, gamma[j])
        gamma[j] ~ dunif(gamma.inf, gamma.sup)
    }
    for (i in 1:Nareas) {
        ceros[i] <- 0
    }
    gamma.inf <- min.bound(C[], adj[], num[], M[])
    gamma.sup <- max.bound(C[], adj[], num[], M[])

    # M matrix with different variances for each row
    for (i in 1:Ndiseases) {
        for (j in 1:Ndiseases) {
            M.mat[i, j] ~ dnorm(0, prec[i])
        }
    }

    # Prior distributions
    for (i in 1:Ndiseases) {
        alpha[i] ~ dflat()
        prec[i] <- pow(sdstruct[i], -2)
        sdstruct[i] ~ dunif(0, 10)
    }
}
```

RVA M-model for the original 3 diseases and 1 simulated pattern

```
set.seed(1)
D = diag(VR.wb$num)
W = matrix(0, nrow = nregions, ncol = nregions)
cont = 1
for (i in 1:length(VR.wb$num)) {
    for (j in 1:(VR.wb$num[i])) {
        W[i, VR.wb$adj[cont]] = 1
        cont = cont + 1
    }
}
sd.PCAR = 0.4
rho = 0.9
precMatrix.PCAR = (sd.PCAR^(-2)) * (D - rho * W)
covMatrix.PCAR = solve(precMatrix.PCAR)

# Expected cases, the same than for lung cancer
Expected = Exp.mv3[1]

ndiseases = 4
inits = function() {
```

```r
    list(M.mat = matrix(rnorm(ndiseases * ndiseases, 0, 0.2), nrow = ndiseases),
        tDelta = matrix(rnorm(nregions * ndiseases), nrow = ndiseases),
        alpha = runif(ndiseases, -0.5, 0), sdstruct = runif(ndiseases,
            0, 5))
}
param = c("alpha", "log.theta", "M.mat", "sdstruct", "gamma")

cors = array(dim = c(10, 4, 4))
ResM4.RVA <- list()
for (i in 1:10) {
    set.seed(i)
    RR = exp(mvrnorm(n = 1, mu = rep(0, 540), Sigma = covMatrix.PCAR))
    ObsSynthetic = rpois(length(VR.cart), Expected * RR)

    Obs4 = cbind(Obs.mv3, ObsSynthetic)
    Exp4 = cbind(Exp.mv3, Expected)

    data = list(Obs = Obs4, Exp = Exp4, adj = VR.wb$adj, num = VR.wb$num,
        C = rep(1/VR.wb$num, VR.wb$num), M = 1/VR.wb$num, Nareas = nregions,
        Ndiseases = ndiseases)

    ResM4.RVA[[i]] = pbugs(data = data, inits = inits, par = param, model = Mmodel.RVA,
        n.iter = 20000, n.burnin = 5000, bugs.seed = 1)
}
```

```r
cor.mat <- array(dim = c(10, 4, 4))
for (i in 1:10) {
    cor.iter <- apply(ResM4.RVA[[i]]$sims.list$log.theta, 1, function(x) {
        cor(x)
    })
    cor.mat[i, , ] <- matrix(apply(cor.iter, 1, mean), ncol = 4)
}
apply(cor.mat, c(2, 3), mean)
```

```
##             [,1]        [,2]        [,3]        [,4]
## [1,]  1.00000000  0.61480780  0.67579164 -0.04916463
## [2,]  0.61480780  1.00000000  0.86568187 -0.04862207
## [3,]  0.67579164  0.86568187  1.00000000 -0.07678642
## [4,] -0.04916463 -0.04862207 -0.07678642  1.00000000
```

```r
# cors[i, , ] = cor(ResM4.RVA$sims.list$log.theta) apply(cors, c(2, 3),
# mean)
```

RVA M-model for the 21 diseases analysis

The data sets for reproducing this analysis are not shared since they are basically the set of main mortality causes in the Valencian Region. Thus, this data set is not shared in order to preserve the confidenciality of such an amount of data.

```r
load("../Data/Obs21.Rdata")
load("../Data/Exp21.Rdata")

ndiseases = 21
```

```
data = list(Obs = Obs21, Exp = Exp21, adj = VR.wb$adj, num = VR.wb$num,
    C = rep(1/VR.wb$num, VR.wb$num), M = 1/VR.wb$num, Nareas = nregions,
    Ndiseases = ndiseases)
# We set moderate starting values in order to reduce potential
# convergence problems
inits = function() {
    list(M.mat = matrix(rnorm(ndiseases * ndiseases, 0, 0.2), nrow = ndiseases),
        tDelta = matrix(rnorm(nregions * ndiseases), nrow = ndiseases),
        alpha = runif(ndiseases, -0.5, 0), sdstruct = runif(ndiseases,
            0, 1))
}
param = c("alpha", "log.theta", "M.mat", "sdstruct", "gamma", "sSMR")
ResM21.RVA <- pbugs(data = data, inits = inits, par = param, model = Mmodel.RVA,
    n.iter = 20000, n.burnin = 5000, DIC = F, bugs.seed = 1)
```

```
# Computing time
ResM21.RVA$exec_time
```

```
## Time difference of 8.40498 hours
```

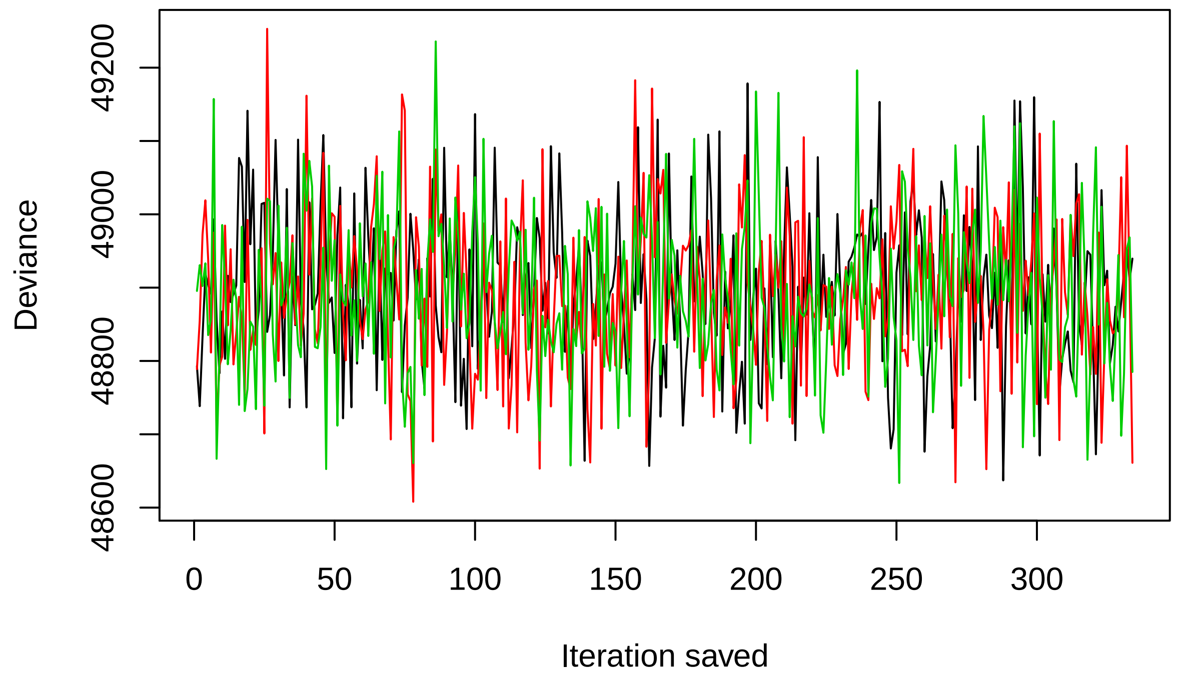
```
# Result summaries for identifiable parameters
aux1 <- grep("alpha", dimnames(ResM21.RVA$summary)[[1]])
aux2 <- grep("sSMR", dimnames(ResM21.RVA$summary)[[1]])
aux <- c(aux1, aux2)
summary(ResM21.RVA$summary[aux, "Rhat"])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.9995  1.0004  1.0014  1.0021  1.0030  1.1022
```

```
summary(ResM21.RVA$summary[aux, "n.eff"])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    32.0   680.0  1000.0   837.8  1000.0  1000.0
```

```
plot.deviance(ResM21.RVA, Obs21, Exp21)
```



Variability of the sSMRs for some different models

```r
sd(log(ResM21.RVA$mean$sSMR[, 1]))
```

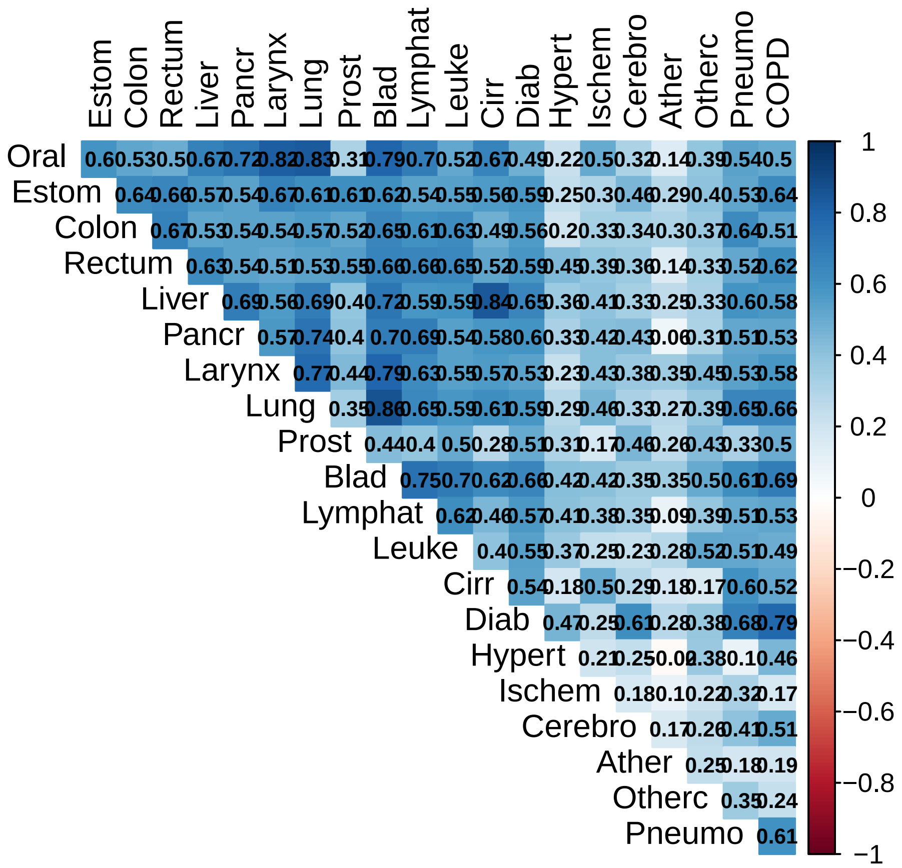
```
## [1] 0.2989862
```

```r
sd(log(ResM.BYM$mean$sSMR[, 1]))
```

```
## [1] 0.2082414
```

Correlations matrix plot

```r
aux = apply(ResM21.RVA$sims.list$sSMR, 1, function(x) {
    cor(log(x))
})
corr21 = matrix(apply(aux, 1, mean), ncol = 21)
dimnames(corr21) = list(dimnames(Obs21)[[2]], dimnames(Obs21)[[2]])

corrplot(corr21, method = "color", type = "upper", number.cex = 0.7, addCoef.col = "black",
    tl.col = "black", tl.srt = 90, insig = "blank", diag = FALSE)
```


Oral
Estom
Colon
Rectum
Liver
Pancr
Larynx
Lung
Prost
Blad
Lymphat
Leuke
Cirr
Diab
Hypert
Ischem
Cerebro
Ather
Otherc
Pneumo
COPD
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1