R code used for Section 5 in the paper: 'Some links between conditional and coregionalized multivariate Gaussian Random fields'

Miguel A. Martinez-Beneito

April, 17th 2018

Load libraries, cartography, data and related functions

```r
# Load Libraries
library(R2WinBUGS)
library(pbugs)
source("C:/MiguePaloma/Libro/Material/Pbugs/Pbugs.0.3.r")

# Load Valencian Region cartography data
load("VR.Rdata")

# Load expected cases
load("Exp.Rdata")
colnames(Exp)<-c("Cirrhosis","Lung","Oral")

# Load fake mortality data.

# This fake dataset has been generated to preserve the privacy of the original dataset
# while making possible to execute the whole code.
# The fake dataset has been generated by executing the following code which tries to
# mimick the original data. In the following code, Obs, Exp stands for the 540X3 matrix
# with the original Observed and Expected cases per municipality.
# In the accompanying dataset, in "ObsFake.Rdata", Obs has been substituted by the
# result of the following code:
#  Fake<-matrix(nrow=540,ncol=3)
#  for(i in 1:540){
#    dif.min<-10000
#    set.seed(1)
#    for(j in 1:ceiling(sum(Exp[i,]))){
#      generated<-rpois(3,as.vector(Exp[i,]))
#      dif.obs<-sum(abs(Obs[i,]-generated))
#      if(dif.obs<dif.min){
#        This<-generated
#        dif.min<-dif.obs
#      }
#    }
#    Fake[i,]<-This
#  }
# Obs<-Fake
# save(Obs,file="ObsFake.Rdata")

# Load Fake observed deaths
```

```r
load("ObsFake.Rdata")
colnames(Obs)<-c("Cirrhosis","Lung","Oral")

# Function for calculating DIC
getDIC<-function(Simu,O,E,save=FALSE){
  Mu<-t(t(Simu$sims.matrix[,which(substr(dimnames(Simu$sims.matrix)[[2]],1,2)=="RR")])*
          as.vector(t(E)))
  D<-apply(Mu,1,function(x){-2*sum(dpois(as.vector(t(O)),x,log=T))})
  DMean<-mean(D)
  MuMean<-apply(Mu,2,mean)
  DInMean<- -2*sum(dpois(as.vector(t(O)),MuMean,log=T))
  if(save==TRUE){return(c(DMean,DMean-DInMean,2*DMean-DInMean))}
  cat("D=",DMean,"pD=",DMean-DInMean,"DIC=",2*DMean-DInMean,"\n")
}
```

M-model

```r
M.model<-function(){
  # Likelihood
  for (i in 1:Nareas) {
    for (k in 1:Ndiseases) {
      Y[i, k] ~ dpois(mu[i, k])
      log(mu[i, k]) <- log(E[i, k]) + alpha[k] + S[i,k]
      RR[i,k] <- exp(alpha[k] + S[i,k])
    }
  }

  for (i in 1:Nareas){
    for(k in 1:Ndiseases){
      S[i,k]<-inprod2(tDelta[,i],SM[,k])
    }
  }
  for(j in 1:Ndiseases){
    tDelta[j, 1:Nareas]~car.proper(zeros[],C[],adj[],num[],M[],1,gamma[j])
    gamma[j]~dunif(gamma.inf,gamma.sup)
  }
  for(i in 1:Nareas){zeros[i]<-0}
  gamma.inf<-min.bound(C[],adj[],num[],M[])
  gamma.sup<-0.99

  # Square matrix Ndiseases x Ndiseases
  for (i in 1:Ndiseases){
    for (j in 1:Ndiseases){
      SM[i,j] ~ dnorm(0,prec)
    }
  }
  prec<-pow(sdstruct,-2)
  sdstruct~dunif(0,10)

  # Other priors
  for (k in 1:Ndiseases){
    alpha[k] ~ dflat()
  }
}

# Run M-model
parameters<-c("mu","alpha","RR","gamma","SM","sdstruct")

data<-list(Y=Obs,E=Exp, adj=VR.wb$adj, num=VR.wb$num, C=rep(1/VR.wb$num,VR.wb$num),
           M=1/VR.wb$num, Nareas=540, Ndiseases=3)

inits<-function(){
  list(SM=matrix(rnorm(3*3,0,1),nrow=3), tDelta=matrix(rnorm(540*3),nrow=3),
       alpha=runif(3,-0.5,0), sdstruct=runif(1,0,5))
}

res.M<-Pbugs(data=data,inits=inits,par=parameters,model=M.model, n.iter=20000,
             n.burnin = 5000, n.sims=3000, n.chains = 3)

getDIC(O = Obs,E = Exp,Simu = res.M)
# DIC for the original dataset:
# D= 6968.566 pD= 419.806 DIC= 7388.372
```

permuted QsR-models

```
pQsR.model<-function(){
  # Likelihood
  for (i in 1:Nareas) {
    for (k in 1:Ndiseases) {
      Y[i, k] ~ dpois(mu[i, k])
      log(mu[i, k]) <- log(E[i, k]) + alpha[k] + S[i,k]
      RR[i,k] <- exp(alpha[k] + S[i,k])
    }
  }

  for (i in 1:Nareas){
    for(k in 1:Ndiseases){
      S[i,k]<-inprod2(tDelta[,i],SM.star[i,,k])
    }
  }
  for(j in 1:Ndiseases){
    tDelta[j, 1:Nareas]~car.proper(zeros[],C[],adj[],num[],M[],1,gamma[j])
    gamma[j]~dunif(gamma.inf,gamma.sup)
  }
  for(i in 1:Nareas){zeros[i]<-0}
  gamma.inf<-min.bound(C[],adj[],num[],M[])
  gamma.sup<-0.99

  for(i in 1:Nareas){
    for(j in 1:Ndiseases){
      for(k in 1:Ndiseases){
        SM.star[i,j,k]<-SM[perm[i,j],k]
      }
    }
  }

  for(i in 1:Nareas){
    for(j in 1:Ndiseases){
      perm[i,j]<-permutas[cual[i],j]
    }
    cual[i]~dcat(p[1:ncual])
  }
  for(i in 1:ncual){p[i]<-1/ncual}

  # Square matrix Ndiseases x Ndiseases
  for (i in 1:Ndiseases){
    for (j in 1:Ndiseases){
      SM[i,j] ~ dnorm(0,prec)
    }
  }
  prec<-pow(sdstruct,-2)
  sdstruct~dunif(0,10)

  # Other priors
  for (k in 1:Ndiseases){
    alpha[k] ~ dflat()
  }
}

# Run permuted QsR-model

# Full-QsR
param<-c("alpha","RR","gamma","SM","sdstruct","perm","cual")

data<-list(Y=Obs,E=Exp,adj=VR.wb$adj,num=VR.wb$num,ncual=6,
           C=rep(1/VR.wb$num,VR.wb$num),M=1/VR.wb$num,Nareas=540,Ndiseases=3,
           permutas=matrix(c(1,2,3,1,3,2,2,1,3,2,3,1,3,1,2,3,2,1),ncol=3,byrow=T))

inits<-function(){
  list(SM=matrix(rnorm(3*3,0,1),nrow=3), tDelta=matrix(rnorm(540*3),nrow=3),
alpha=runif(3,-0.5,0), sdstruct=runif(1,0,5), cual=sample(1:6,540,replace=T))
}

res.QsR.full<-Pbugs(data=data,inits=inits,par=param,model=pQsR.model,n.iter=20000,
                    n.burnin = 5000,DIC=F,n.sims=3000,n.chains = 3)

getDIC(O=Obs,E=Exp,Simu=res.QsR.full)
# DIC for the original data:
# D= 6962.519 pD= 432.7378 DIC= 7395.257


# (2->3) QsR-model
data<-list(Y=Obs,E=Exp,adj=VR.wb$adj,num=VR.wb$num,ncual=2,
           C=rep(1/VR.wb$num,VR.wb$num),M=1/VR.wb$num,Nareas=540,Ndiseases=3,
           permutas=matrix(c(1,2,3,1,3,2),ncol=3,byrow=T))
```

```
inits<-function(){
  list(SM=matrix(rnorm(3*3,0,1),nrow=3),
       tDelta=matrix(rnorm(540*3),nrow=3),
       alpha=runif(3,-0.5,0),
       sdstruct=runif(1,0,5),cual=sample(1:2,540,replace=T))
}

res.QsR.23<-Pbugs(data=data,inits=inits,par=param,model=pQsR.model,n.iter=20000,
                  n.burnin = 5000,n.sims=3000,DIC=F,n.chains = 3)

getDIC(O=Obs,E=Exp,Simu=res.QsR.23)
# DIC for the original data:
# D= 6952.565 pD= 418.3594 DIC= 7370.924


# (3->1->2) QsR-model
data<-list(Y=Obs,E=Exp,adj=VR.wb$adj,num=VR.wb$num,ncual=2,
           C=rep(1/VR.wb$num,VR.wb$num),M=1/VR.wb$num,Nareas=540,Ndiseases=3,
           permutas=matrix(c(1,2,3,3,1,2),ncol=3,byrow=T))

res.QsR.132<-Pbugs(data=data,inits=inits,par=param,model=pQsR.model,n.iter=20000,
                   n.burnin = 5000,n.sims=3000,DIC=F,n.chains = 3)

getDIC(O=Obs,E=Exp,Simu=res.QsR.132)
# DIC for the original data:
# D= 6953.785 pD= 434.6633 DIC= 7388.448
```

Some results for the (2->3) QsR-model

Proportion of posterior samples that each Valencia neighbour's M matrix is flipped in the same manner as for Valencia (Table 2 in the paper)

```
# Valencia Neighbours (Valencia is the municipality number 526)
neigh<-VR.nb[[526]]

# Probability of having the same M matrix as Valencia for each of its neighbours.
P.equal.val.neigh<-apply(res.QsR.23$sims.list$cual[,neigh],2,
                         function(x){
                           mean(x==res.QsR.23$sims.list$cual[,526])
                          })

apply(Obs[neigh[which(P.equal.val.neigh<0.2)],],2,sum)
# Results for the original data
# Cirrhosis      Lung            Oral
# 120            641             65

apply(Obs[neigh[which(P.equal.val.neigh>0.8)],],2,sum)
# Results for the original data
# Cirrhosis      Lung            Oral
# 542           1439            126
```

```
Obs[526,]
# Results for the original data
# Cirrhosis      Lung            Oral
# 2722           7938            734
```

Cross-covariance matrices for Valencia and some of its neighbours (Table 3 in the paper)

```
cual.max<-vector()
neigh<-VR.nb[[526]]
perm23<-matrix(c(1,2,3,1,3,2),ncol=3,byrow=T)
n.it<-dim(res.QsR.23$sims.list$SM)[1]
Cov<-array(dim=c(n.it,3,3))
Covs.mean<-list()
for(j in 1:length(neigh)){
  for(i in 1:n.it){
    Cov[i,,]<-t(res.QsR.23$sims.list$SM[i,perm23[res.QsR.23$sims.list$cual[i,526],],])%*%
      diag(res.QsR.23$sims.list$gamma[i,])%*%
      (res.QsR.23$sims.list$SM[i,perm23[res.QsR.23$sims.list$cual[i,neigh[j]],],])
  }
  Covs.mean[[j]]<-apply(Cov,c(2,3),mean)
}

which.max(sapply(Covs.mean,function(x){y<-x-t(x);sum(abs(y))/2}))
# Result for the original data
# 29
which.min(P.equal.val.neigh)
# Result for the original data
# 29
# The more asymmetric cross-covariance matrix is that of the neighbouring municipality
# with lowest probability of having the same M matrix as Valencia.

# Cross-covariance for Valencia and its neighbour with lowest probability of having
# the same M matrix as Valencia
Covs.mean[[which.min(P.equal.val.neigh)]]
# Results for the original data
#              [,1]      [,2]      [,3]
# [1,] -0.01900943 0.1248847 0.1083489
# [2,]  0.07589639 0.1482197 0.1595531
# [3,]  0.10960737 0.1698762 0.2449156

# Cross-covariance for Valencia and its neighbour with highest probability of having
# the same M matrix as Valencia
Covs.mean[[which.max(P.equal.val.neigh)]]
# Results for the original data
#            [,1]      [,2]      [,3]
# [1,] 0.2065186 0.1323064 0.1562186
# [2,] 0.1321415 0.1507940 0.1723278
# [3,] 0.1564089 0.1723547 0.2578943
```