Smooth Scalar-on-Image Regression via Spatial Bayesian Variable Selection: Supplement B

By: Jeff Goldsmith, Lei Huang, Ciprian Crainiceanu

B R Code for Single-site Sampler

The following code implements the Gibbs sampler used for two-dimensional scalar-on-image regression; straightforward extensions allow for higher dimensions. In this code, the quantities `N1` and `N2` are the dimensions of the image predictors; `xbeta` is the vector of dot products $\boldsymbol{X} \cdot \boldsymbol{\beta}$ and is initialized to 0, as is $\boldsymbol{\beta}$; and `sigeps.inv` and `sigbeta.inv` are $\frac{1}{\sigma^2_\epsilon}$ and $\frac{1}{\sigma^2_{\boldsymbol{\beta}}}$, respectively. This code sweeps over all image locations, at each point making a Bernoulli choice between a zero and nonzero coefficient based on the prior information and the relative impact on the likelihood comparing a zero to a nonzero coefficient. Coefficients `alpha` for fixed effects `FixEf` are updated after the sweep over image locations.

```
for(i in N1:1){
  for(j in 1:N2){
    # define delta neighborhood
    delta = cbind(c(i-1, i+1, i,i), c(j,j, j-1, j+1))
    delta = replace(delta, which(delta<=0), NA)
    delta[,1] = replace(delta[,1], which(delta[,1]>N1), NA)
    delta[,2] = replace(delta[,2], which(delta[,2]>N2), NA)
    dl = sum(complete.cases(delta))
    Xl=X[i,j,]
    xbeta0 = xbeta - betaHat[i,j]*Xl;  txbeta0 = t(xbeta0)

    # generate nonzero coefficient
    betabar = mean(betaHat[delta], na.rm=TRUE)
    Sigl = 1/(sigeps.inv * InProd[i,j]  + dl * sigbeta.inv)
    Mul = Sigl * ( sigeps.inv * (tY - t(FixEf%*%alpha) - txbeta0)%*%Xl
               + dl * sigbeta.inv * betabar   )
    betaStar = rnorm(1, Mul, sqrt(Sigl))

    xbeta1 = xbeta0+Xl*betaStar; txbeta1=t(xbeta1)

    # compute posterior probability g_l
    IND = sum(1-2*z[delta], na.rm=TRUE)
    g = sqrt(2*pi*SigB/dl)*exp((-.5*sigeps.inv)*(txbeta0%*%xbeta0
          + 2*(txbeta1-txbeta0)%*%(Y+FixEf%*%alpha)
          - txbeta1%*%xbeta1) + .5*sigbeta.inv*dl*(betaStar-betabar)^2
          - A[i,j] + b*IND )
    ppos = 1/(1+g)

    # Bernoulli choice based on posterior probability
    if(rbinom(1, 1, ppos)==1) {z[i,j] = 1; betaHat[i,j] = betaStar}
    else {z[i,j] = betaHat[i,j] = 0}
```

```
    # update X * Beta based on current iteration
    xbeta = xbeta0 + betaHat[i,j]*Xl
  }
}
```