Using scalableBayesEEG

Shariq Mohammed

R package for “Mohammed, S. and Dey, D.K., 2020. Scalable spatio-temporal Bayesian analysis of high-dimensional electroencephalography data. The Canadian Journal of Statistics, pp.1-20.”

Code to run the model in (Mohammed et.al, 2020) on the EEG data using the package scalableBayesEEG.

Contents

1. Load EEG data and divide it into training and test.
2. Specify temporal and spatial covariance matrices
3. Build the model proposed in the manuscript with the training data and predict for the test data set.

```
# install the package (devtools package needed)
if(!require(devtools)) install.packages("devtools")
devtools::install_github('shariq-mohammed/scalableBayesEEG')

# Load the package
library(scalableBayesEEG)
```

Load EEG data and divide it into training and test

Load EEG Data

```
X = scalableBayesEEG::X_eeg
```

Load location indices for which distance matrix is available

```
ind64to57 = scalableBayesEEG::ind64to57_eeg
```

Update EEG data for locations above

```
X = X[,ind64to57,]
```

Load binary response

```
y = t(scalableBayesEEG::y_eeg)
```

Load distance matrix

```
dist.mat = scalableBayesEEG::dist.mat_eeg
```

Dimensions of EEG data

```
tau = dim(X)[1]
p = dim(X)[2]
n = dim(X)[3]
```

Sample indices for training data

```
trn.ind = sample.int(n, size = 100)
```

Responses for training and testing

```
y.trn = y[trn.ind] # responses from training data
y.test = y[-trn.ind] # responses from test data
```

EEG data for training and testing

```
X.trn = aperm(X[,,trn.ind], c(3,2,1)) # training EEG data
X.test = aperm(X[,,-trn.ind], c(3,2,1)) # test EEG data
```

Specify temporal and spatial covariance matrices

Specify temporal covariance matrix

```
prior.temp = matrix(0, nrow = tau, ncol = tau)
for(t1 in 1:tau) for(t2 in t1:tau){
  if(t1 == t2) prior.temp[t1,t1] = 2
  if(t2 == t1+1) prior.temp[t1,t2] = prior.temp[t2,t1] = -1
  if(t1 == 1 & t2 == 1) prior.temp[t1,t1] = 1
  if(t1 == tau & t2 == tau) prior.temp[t1,t1] = 1
}
T.mat = prior.temp+diag(0.6,tau)
Temp.cov = cov2cor(chol2inv(chol(T.mat))) # temporal covariance matrix
```

Specify spatial covariance matrix

```
dmat = (dist.mat+t(dist.mat))/2
W.mat = exp(-(dmat^2)/0.1)
D.mat = diag(rowSums(W.mat)+0.6)
G.mat = D.mat-W.mat
Spat.cov = cov2cor(chol2inv(chol(G.mat))) # spatial covariance matrix
```

Build the model proposed in the manuscript with the training data and predict for the test data set

Run Bayesian modeling with approximate estimation

```
finalModel = spTempBayes(y.trn, X.trn, c_init = rep(1, tau), beta_init = NULL,
                         zeta_init = NULL, nusq_init = NULL, l_init = NULL,
                         prior.mu = rep(0, p*tau), Spat.cov, Temp.cov,
                         v0 = 0.005, a1 = 5, a2 = 15, q = 1,
                         Nmcmc = 210, # takes about 120 seconds
                         burnin = 10, X.test = X.test, type='approx')
```

Plot heatmap of posterior estimate of the betas (rows are time points, columns are locations)

```
if(!require(lattice)) install.packages("lattice")
lattice::levelplot(finalModel$b.strucBayes)
```