```
## Load libraries
library(splines)
library(MASS)
library(FDRreg)

## Loading required package:  fda
## Loading required package:  Matrix
##
## Attaching package:  'fda'
## The following object is masked from 'package:graphics':
##
##    matplot
## Loading required package:  BayesLogit
## Loading required package:  mvtnorm

library(curl)

library(doParallel) ##to make cluster (on Windows)

## Warning:  package 'doParallel' was built under R version 3.3.2
## Loading required package:  foreach
## Loading required package:  iterators
## Loading required package:  parallel

library(foreach) ##to use foreach function that does the parallel processing
library(doRNG) ##for reproducible seeds when doing parallel processing

## Loading required package:  rngtools
## Loading required package:  pkgmaker
## Loading required package:  registry
##
## Attaching package:  'pkgmaker'
## The following object is masked from 'package:base':
##
##    isNamespaceLoaded
```

1 Probability of being a false positive as a linear function of time

Load simulations and (re)define some variables:

```
load("simResults_1.RData")

nSims <- length(pValuesSims)
ntest <- length(pValuesSims[[1]])
```

```
## Set up the time vector and the probability of being null
tme <- seq(-1,2,length=ntest)
pi0 <- 1/4*tme+1/2
```

Perform estimation and save estimates:

```
cl<-makeCluster(8) ##specify number of cores less than or equal to number of cores on your c
registerDoParallel(cl)

set.seed(31084)

pi0hatScottMat <- foreach(sim=1:nSims, .combine="rbind", .packages="FDRreg", .errorhandling=
  ##first transform the p-values into z-scores
  ##randomly assign positive or negative sign
  zScores <- qnorm(1-pValuesSims[[sim]]/2)##*sample(c(-1,1), replace=TRUE, size=length(pVal
  fdr <- FDRreg(zScores, matrix(tme, ncol=1),
                nulltype = 'empirical',
                control=list(lambda=1));
  pi0hatScott.sim <- 1-fdr$priorprob
}

##close the cluster
stopCluster(cl)

dim(pi0hatScottMat)

## [1] 8883 1000

pi0hatScottMean <- colMeans(pi0hatScottMat)
pi0hatScottVar <- apply(pi0hatScottMat,2,var)

length(pi0hatScottMean)

## [1] 1000

##save results
save(file="simResults_pi0x_Scott_1.RData",
     list=c("tme", "pi0",
            "pi0hatScottMean","pi0hatScottVar"))
```

2 Probability of being a false positive as a smooth function of time

Load simulations and (re)define some variables:

```
load("simResults_2.RData")

nSims <- length(pValuesSims)
ntest <- length(pValuesSims[[1]])

## Set up the time vector and the probability of being null
tme <- seq(-1,2,length=ntest)
pi0 <- pnorm(tme)

splineMat <- ns(tme,df=3)
```

Perform estimation and save estimates:

```
cl<-makeCluster(8) ##specify number of cores less than or equal to number of cores on your c
registerDoParallel(cl)

set.seed(31084)

pi0hatScottMatFitLin <- foreach(sim=1:nSims, .combine="rbind", .packages="FDRreg", .errorhan
  zScores <- qnorm(1-pValuesSims[[sim]]/2)##*sample(c(-1,1), replace=TRUE, size=length(pVal
  fdr <- FDRreg(zScores, tme,
                nulltype = 'empirical',
                control=list(lambda=1));
  pi0hatScottMatFitLin.sim  <- 1-fdr$priorprob
}

##close the cluster
stopCluster(cl)

cl<-makeCluster(8) ##specify number of cores less than or equal to number of cores on your c
registerDoParallel(cl)

set.seed(31084)

pi0hatScottMatFitSpl <- foreach(sim=1:nSims, .combine="rbind", .packages="FDRreg", .errorhan
  zScores <- qnorm(1-pValuesSims[[sim]]/2)##*sample(c(-1,1), replace=TRUE, size=length(pVal
  fdr <- FDRreg(zScores, splineMat,
                nulltype = 'empirical',
                control=list(lambda=1));
  pi0hatScottMatFitSpl.sim  <- 1-fdr$priorprob
}

##close the cluster
stopCluster(cl)
```

```r
dim(pi0hatScottMatFitLin)

## [1] 8779 1000

dim(pi0hatScottMatFitSpl)

## [1] 8779 1000

pi0hatLin.ScottMean <- colMeans(pi0hatScottMatFitLin)
pi0hatLin.ScottVar <- apply(pi0hatScottMatFitLin,2,var)

pi0hatSpl.ScottMean <- colMeans(pi0hatScottMatFitSpl)
pi0hatSpl.ScottVar <- apply(pi0hatScottMatFitSpl,2,var)

length(pi0hatLin.ScottMean)

## [1] 1000

length(pi0hatSpl.ScottMean)

## [1] 1000

##save results
save(file="simResults_pi0x_Scott_2.RData",
     list=c("tme", "pi0",
            "pi0hatLin.ScottMean", "pi0hatLin.ScottVar",
            "pi0hatSpl.ScottMean", "pi0hatSpl.ScottVar"))
```

3 Probability of being a false positive as a sine + step function

Load simulations and (re)define some variables:

```r
load("simResults_3.RData")

nSims <- length(pValuesSims)
ntest <- length(pValuesSims[[1]])

## Set up the time vector and the probability of being null
tme1 <- seq(-1*pi,2*pi,length=ntest)
tme2 <- rep(1:0, each=ntest/2)

pi0 <- 1/4*sin(tme1) + tme2/4 + 1/2
range(pi0)
```

```
## [1] 0.2500028 0.9999972

splineMat3 <- cbind(ns(tme1,df=3), tme2)
splineMat20 <- cbind(ns(tme1,df=20), tme2)
```

Perform estimation and save estimates:

```
cl<-makeCluster(8) ##specify number of cores less than or equal to number of cores on your c
registerDoParallel(cl)

set.seed(31084)

pi0hatScottMat3 <- foreach(sim=1:nSims, .combine="rbind", .packages="FDRreg", .errorhandling
  zScores <- qnorm(1-pValuesSims[[sim]]/2)##*sample(c(-1,1), replace=TRUE, size=length(pVal
  fdr <- FDRreg(zScores, splineMat3,
                nulltype = 'empirical',
                control=list(lambda=1));
  pi0hatScottMat3.sim  <- 1-fdr$priorprob
}

##close the cluster
stopCluster(cl)

cl<-makeCluster(8) ##specify number of cores less than or equal to number of cores on your c
registerDoParallel(cl)

set.seed(31084)

pi0hatScottMat20 <- foreach(sim=1:nSims, .combine="rbind", .packages="FDRreg", .errorhandlin
  zScores <- qnorm(1-pValuesSims[[sim]]/2)##*sample(c(-1,1), replace=TRUE, size=length(pVal
  fdr <- FDRreg(zScores, splineMat20,
                nulltype = 'empirical',
                control=list(lambda=1));
  pi0hatScottMat20.sim  <- 1-fdr$priorprob
}

##close the cluster
stopCluster(cl)

dim(pi0hatScottMat3)

## [1] 9045 1000

dim(pi0hatScottMat20)

## [1] 9045 1000
```

```
pi0hat3.ScottMean <- colMeans(pi0hatScottMat3)
pi0hat3.ScottVar <- apply(pi0hatScottMat3,2,var)

pi0hat20.ScottMean <- colMeans(pi0hatScottMat20)
pi0hat20.ScottVar <- apply(pi0hatScottMat20,2,var)

length(pi0hat3.ScottMean)

## [1] 1000

length(pi0hat20.ScottMean)

## [1] 1000

##save results
save(file="simResults_pi0x_Scott_3.RData",
     list=c("tme1", "tme2", "pi0",
            "pi0hat3.ScottMean", "pi0hat3.ScottVar",
            "pi0hat20.ScottMean", "pi0hat20.ScottVar"))
```

Session info:

```
devtools::session_info()

## Session info -------------------------------------------------

##  setting  value
##  version  R version 3.3.1 (2016-06-21)
##  system   x86_64, mingw32
##  ui       RTerm
##  language (EN)
##  collate  English_United States.1252
##  tz       America/New_York
##  date     2016-12-31

## Packages -----------------------------------------------------

##  package    * version date
##  assertthat   0.1     2013-12-06
##  BayesLogit * 0.5.1   2014-07-21
##  codetools    0.2-14  2015-07-15
##  colorspace   1.2-6   2015-03-11
##  curl       * 1.0     2016-07-24
##  DBI          0.4-1   2016-05-08
##  devtools     1.12.0  2016-06-24
##  digest       0.6.9   2016-01-08
##  doParallel * 1.0.10  2015-10-14
```

```
##  doRNG      * 1.6     2014-03-07
##  dplyr        0.5.0   2016-06-24
##  evaluate     0.10    2016-10-11
##  fda        * 2.4.4   2014-12-16
##  FDRreg     * 0.2-1   2016-08-30
##  foreach    * 1.4.3   2015-10-13
##  ggdendro     0.1-20  2016-04-27
##  ggplot2      2.1.0   2016-03-01
##  gridExtra    2.2.1   2016-02-29
##  gtable       0.2.0   2016-02-26
##  highr        0.6     2016-05-09
##  iterators  * 1.0.8   2015-10-13
##  knitr      * 1.15.1  2016-11-22
##  lattice      0.20-33 2015-07-14
##  lazyeval     0.2.0   2016-06-12
##  magrittr     1.5     2014-11-22
##  MASS       * 7.3-45  2016-04-21
##  Matrix     * 1.2-6   2016-05-02
##  memoise      1.0.0   2016-01-29
##  mosaic       0.14.4  2016-07-29
##  mosaicData   0.14.0  2016-06-17
##  munsell      0.4.3   2016-02-13
##  mvtnorm    * 1.0-5   2016-02-02
##  pkgmaker   * 0.22    2014-05-14
##  plyr         1.8.4   2016-06-08
##  R6           2.1.2   2016-01-26
##  Rcpp         0.12.6  2016-07-19
##  registry   * 0.3     2015-07-08
##  rngtools   * 1.2.4   2014-03-06
##  scales       0.4.0   2016-02-26
##  stringi      1.1.1   2016-05-27
##  stringr      1.0.0   2015-04-30
##  tibble       1.1     2016-07-04
##  tidyr        0.5.1   2016-06-14
##  withr        1.0.2   2016-06-20
##  xtable       1.8-2   2016-02-05
##  source
##  CRAN (R 3.3.1)
##  CRAN (R 3.3.0)
##  CRAN (R 3.3.1)
##  CRAN (R 3.3.1)
##  CRAN (R 3.3.1)
##  CRAN (R 3.3.1)
##  CRAN (R 3.3.1)
##  CRAN (R 3.3.1)
```

```
##  CRAN (R 3.3.2)
##  CRAN (R 3.3.1)
##  CRAN (R 3.3.1)
##  CRAN (R 3.3.2)
##  CRAN (R 3.3.1)
##  Github (jgscott/FDRreg@8025d1a)
##  CRAN (R 3.3.1)
##  CRAN (R 3.3.1)
##  CRAN (R 3.3.1)
##  CRAN (R 3.3.1)
##  CRAN (R 3.3.1)
##  CRAN (R 3.3.1)
##  CRAN (R 3.3.0)
##  CRAN (R 3.3.2)
##  CRAN (R 3.3.1)
##  CRAN (R 3.3.1)
##  CRAN (R 3.3.1)
##  CRAN (R 3.3.1)
##  CRAN (R 3.3.1)
##  CRAN (R 3.3.1)
##  CRAN (R 3.3.1)
##  CRAN (R 3.3.1)
##  CRAN (R 3.3.1)
##  CRAN (R 3.3.1)
##  CRAN (R 3.3.0)
##  CRAN (R 3.3.1)
##  CRAN (R 3.3.1)
##  CRAN (R 3.3.1)
##  CRAN (R 3.3.1)
##  CRAN (R 3.3.1)
##  CRAN (R 3.3.1)
##  CRAN (R 3.3.1)
##  CRAN (R 3.3.1)
##  CRAN (R 3.3.0)
##  CRAN (R 3.3.1)
##  CRAN (R 3.3.1)
##  CRAN (R 3.3.1)
##  CRAN (R 3.3.1)
##  CRAN (R 3.3.1)
```