

```

## Load libraries
library(splines)
library(MASS)
library(swfdr)

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

## 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

##don't need doRNG here, but easier to keep it in

```

Function to pull out means and variances across simulations:

```

pullMeansVars <- function(pi0EstSim)
{
  ##pull out estimates at lambda=0.8, lambda=0.9, and final estimate
  pi0hat0.8 <- sapply(pi0EstSim, function(x){x[[1]]})
  pi0hat0.9 <- sapply(pi0EstSim, function(x){x[[2]]})
  pi0hatFinal <- sapply(pi0EstSim, function(x){x[[3]]})

  ##get means across simulations
  pi0hatMean0.8 <- rowMeans(pi0hat0.8)
  pi0hatMean0.9 <- rowMeans(pi0hat0.9)
  pi0hatMeanFinal <- rowMeans(pi0hatFinal)

  ##also get variances across simulations
  pi0hatVar0.8 <- apply(pi0hat0.8,1,var)
  pi0hatVar0.9 <- apply(pi0hat0.9,1,var)
  pi0hatVarFinal <- apply(pi0hatFinal,1,var)

  return(list(pi0hatMean0.8=pi0hatMean0.8,
              pi0hatMean0.9=pi0hatMean0.9,
              pi0hatMeanFinal=pi0hatMeanFinal,

```

```

        pi0hatVar0.8=pi0hatVar0.8,
        pi0hatVar0.9=pi0hatVar0.9,
        pi0hatVarFinal=pi0hatVarFinal))
}

```

## 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]])

##sequence of lambdas
lambdas <- round(seq(0.05, 0.95, 0.05),2)
which.0.8 <- which(lambdas==0.8)
which.0.9 <- which(lambdas==0.9)
which.0.8

## [1] 16

which.0.9

## [1] 18

## 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 computer
registerDoParallel(cl)

pi0EstSim <- foreach(sim = 1:nSims, .packages=c("swfdr")) %dorng% {
  res <- lm_pi0(pValuesSims[[sim]], lambda=lambdas, X=tme,
               smooth.df=3, threshold=TRUE);
  res.pi0.lambda <- res$pi0.lambda;
  list(res.pi0.lambda[,which.0.8],
       res.pi0.lambda[,which.0.9],
       res$pi0)}

```

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

##pull out means and variances of estimates at lambda=0.8, lambda=0.9, and final estimate
pi0MeansVars <- pullMeansVars(pi0EstSim)

##save results
save(file="simResults_pi0x_thresh_1.RData",
      list=c("tme", "pi0", "pi0MeansVars"))
```

## 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]])

##sequence of lambdas
lambdas <- round(seq(0.05, 0.95, 0.05),2)
which.0.8 <- which(lambdas==0.8)
which.0.9 <- which(lambdas==0.9)
which.0.8

## [1] 16

which.0.9

## [1] 18

## 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
registerDoParallel(cl)

pi0EstSim.lin <- foreach(sim = 1:nSims, .packages=c("swfdr")) %dornrg% {
```

```

res <- lm_pi0(pValuesSims[[sim]], lambda=lambdas, X=tme,
             smooth.df=3, threshold=TRUE);
res.pi0.lambda <- res$pi0.lambda;
list(res.pi0.lambda[,which.0.8],
     res.pi0.lambda[,which.0.9],
     res$pi0)}

##close the cluster
stopCluster(cl)

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

pi0EstSim.spl <- foreach(sim = 1:nSims, .packages=c("swfdr")) %dornrg% {
  res <- lm_pi0(pValuesSims[[sim]], lambda=lambdas, X=splineMat,
               smooth.df=3, threshold=TRUE);
  res.pi0.lambda <- res$pi0.lambda;
  list(res.pi0.lambda[,which.0.8],
       res.pi0.lambda[,which.0.9],
       res$pi0)}

##close the cluster
stopCluster(cl)

##pull out means and variances of estimates at lambda=0.8, lambda=0.9, and final estimate
pi0Lin.MeansVars <- pullMeansVars(pi0EstSim.lin)
pi0Spl.MeansVars <- pullMeansVars(pi0EstSim.spl)

##save results
save(file="simResults_pi0x_thresh_2.RData",
     list=c("tme", "pi0", "pi0Lin.MeansVars", "pi0Spl.MeansVars"))

```

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

Load simulations and (re)define some variables:

```

load("simResults_3.RData")

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

##sequence of lambdas

```

```

lambdas <- round(seq(0.05, 0.95, 0.05),2)
which.0.8 <- which(lambdas==0.8)
which.0.9 <- which(lambdas==0.9)
which.0.8

## [1] 16

which.0.9

## [1] 18

## 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
registerDoParallel(cl)

pi0EstSim3 <- foreach(sim = 1:nSims, .packages=c("swfdr")) %dorng% {
  res <- lm_pi0(pValuesSims[[sim]], lambda=lambdas, X=splineMat3,
               smooth.df=3, threshold=TRUE);
  res.pi0.lambda <- res$pi0.lambda;
  list(res.pi0.lambda[,which.0.8],
       res.pi0.lambda[,which.0.9],
       res$pi0)}

##close the cluster
stopCluster(cl)

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

pi0EstSim20 <- foreach(sim = 1:nSims, .packages=c("swfdr")) %dorng% {
  res <- lm_pi0(pValuesSims[[sim]], lambda=lambdas, X=splineMat20,
               smooth.df=3, threshold=TRUE);
  res.pi0.lambda <- res$pi0.lambda;

```

```

    list(res.pi0.lambda[,which.0.8],
         res.pi0.lambda[,which.0.9],
         res$pi0)}

##close the cluster
stopCluster(cl)

##pull out means and variances of estimates at lambda=0.8, lambda=0.9, and final estimate
pi0_3.MeanVars <- pullMeansVars(pi0EstSim3)
pi0_20.MeanVars <- pullMeansVars(pi0EstSim20)

##save results
save(file="simResults_pi0x_thresh_3.RData",
     list=c("tme1", "tme2", "pi0",
            "pi0_3.MeanVars", "pi0_20.MeanVars"))

```

Session info:

```

devtools::session_info()

## Session info -----
##  setting
##  version
##  system
##  ui
##  language
##  collate
##  tz
##  date
##  value
##  R Under development (unstable) (2016-12-08 r71762)
##  x86_64, mingw32
##  RTerm
##  (EN)
##  English_United States.1252
##  America/New_York
##  2016-12-29

## Packages -----
##  package      * version date          source
##  assertthat    0.1      2013-12-06 CRAN (R 3.3.2)
##  codetools      0.2-15   2016-10-05 CRAN (R 3.4.0)
##  colorspace     1.3-1    2016-11-18 CRAN (R 3.3.2)
##  DBI             0.5-1    2016-09-10 CRAN (R 3.3.2)
##  devtools       1.12.0    2016-06-24 CRAN (R 3.3.2)

```

##	digest	0.6.10	2016-08-02	CRAN	(R 3.3.2)
##	doParallel	* 1.0.10	2015-10-14	CRAN	(R 3.3.2)
##	doRNG	* 1.6	2014-03-07	CRAN	(R 3.4.0)
##	dplyr	0.5.0	2016-06-24	CRAN	(R 3.3.2)
##	evaluate	0.10	2016-10-11	CRAN	(R 3.3.2)
##	foreach	* 1.4.3	2015-10-13	CRAN	(R 3.3.2)
##	ggplot2	2.2.0	2016-11-11	CRAN	(R 3.3.2)
##	gtable	0.2.0	2016-02-26	CRAN	(R 3.3.2)
##	highr	0.6	2016-05-09	CRAN	(R 3.3.2)
##	iterators	* 1.0.8	2015-10-13	CRAN	(R 3.3.2)
##	knitr	* 1.15.1	2016-11-22	CRAN	(R 3.3.2)
##	lazyeval	0.2.0	2016-06-12	CRAN	(R 3.3.2)
##	magrittr	1.5	2014-11-22	CRAN	(R 3.3.2)
##	MASS	* 7.3-45	2016-04-21	CRAN	(R 3.4.0)
##	memoise	1.0.0	2016-01-29	CRAN	(R 3.3.2)
##	munsell	0.4.3	2016-02-13	CRAN	(R 3.3.2)
##	pkgmaker	* 0.22	2014-05-14	CRAN	(R 3.3.2)
##	plyr	1.8.4	2016-06-08	CRAN	(R 3.3.2)
##	R6	2.2.0	2016-10-05	CRAN	(R 3.3.2)
##	Rcpp	0.12.8	2016-11-17	CRAN	(R 3.3.2)
##	registry	* 0.3	2015-07-08	CRAN	(R 3.3.2)
##	reshape2	1.4.2	2016-10-22	CRAN	(R 3.3.2)
##	rngtools	* 1.2.4	2014-03-06	CRAN	(R 3.3.2)
##	scales	0.4.1	2016-11-09	CRAN	(R 3.3.2)
##	stringi	1.1.2	2016-10-01	CRAN	(R 3.3.2)
##	stringr	1.1.0	2016-08-19	CRAN	(R 3.3.2)
##	swfdr	* 0.99.9	2016-12-28	Bioconductor	
##	tibble	1.2	2016-08-26	CRAN	(R 3.3.2)
##	withr	1.0.2	2016-06-20	CRAN	(R 3.3.2)
##	xtable	1.8-2	2016-02-05	CRAN	(R 3.3.2)