Example Running Region Inference

Nov 10, 2017

This document gives an example how to run the region inference function for two-group data.

```
# You need to install the following CRAN packages:
library('MASS')
```

```
## Warning: package 'MASS' was built under R version 3.3.2
```

```
library('truncnorm')
library('restrictedMVN')
```

```
## Warning: package 'restrictedMVN' was built under R version 3.3.2
```

```
#
```

```
# Main functions
setwd('regionInference')
source('regionInference.R')

# Simulation function
source('../simulationCode/simFuncs.R')
```

Generate some multivariate data with a specific detected region.

```
true_val= 0.08
mudiff =  c(0,rep(true_val,3), 0)

sigCorrel = matrix(c(0.04, 0.02, 0.006, 0.0, 0.0,
                  0.02,  0.04, 0.016, 0.0, 0.0,
                  0.006,  0.016,  0.03, 0, 0,
                  0., 0.0,  0, 0.04, 0.01,
                  0.0, 0.0, 0 , 0.01 , 0.03),nrow= 5)


# Conditioning: -1 has to be below thresh
#                1 has to be above thresh
sampParams_diff = setSamplingParameters(na=16, nb = 16,
                        mudiff= mudiff,
                        sig = sigCorrel,
                        conditioning = c(-1,1,1,1,-1))

thresh = 0.08

set.seed(100)
sample_diff = condSample(B = 1, # construct single sample
                         sP = sampParams_diff,
                         thresh = thresh,
                         returndat = TRUE) # return the original values?
```

Decide on the parameters of the run.

The parameters of the function:

- alpha: a parameter determining interval coverage (1-alpha)

- m_values: the sequence of thetas which are evaluated
- nsamp: number of samples from the Gibbs which theta is evaluated
- searchsteps: steps for searching for theta_0 (the sampling value for gibbs)

```
CIparams = setCIparams(alpha=0.1, m_values= seq(-0.2,0.7,0.005),
                       nsamp = 12000,searchstep = 0.01)
```

Choose what type of covariance to use.

```
# If we want to use the known sigma, we adjust the
# covariance of individual samples into the covariance
# of the average
useSigma = diffCov(sigCorrel,sample_diff$groups)
# Alternatively, we can choose to use the sample covariance
# with or without inflation
useSigma = NULL
# inflation parameter
regularize_sig = 0
```

Parameters of the main inference function:

1. the data of the sample
2. division into groups
3. threshold
4. CIparams
5. tilt (T/F) should we use a single sample and tilt, or resample for each value. Tilting far far faste
6. conditioning cis a vector that codes the positive and negative truncations. See above.
7. inner_shape - the shape of the mean vector. If NULL or missing, we use $\mathbf{s} = \mathbf{s}_\Sig
8. regularize_sig - the diagonal of the covariance is inflated by (1+regularize_sig) factor

```
regularize_sig = 0
tilt =TRUE
inner_shape = 1
confInt = regionConfInt(sample_diff$condY[,,1],
                        sample_diff$groups,
                        threshold = thresh,
                        CIparams = CIparams,
                        tilt = TRUE,
                        conditioning = c(-1,1,1,1,-1),
                        inner_shape = NULL,
                        regularize_sig = regularize_sig);
```

Reading the results

```
# Estimate of conditional mean
mean_est = confInt$cond_mean_est

# Estimate of interval
interval = c(confInt$lower_bound, confInt$upper_bound)

# Estimated p-value
pval = confInt$p_value

# Direction - was threshold positive or negative
dir = confInt$direction

cat(dir, pval, interval,mean_est)
```

```
## 1 0.005238891 0.055 0.21 0.14
```

```r
# Internals:
# baseline[condition==1] - the shape of the mean used.
# vals - the m-values
# quants[i] - the probability of getting the observed value or higher under
#               theta=vals[i]
# rej[i]    - was the two-sided test rejected for theta=vals[i]
# stat_obs  - the observed sum of the difference vector (not mean)
plot(confInt$vals, confInt$quants,col = c(3,2)[confInt$rej+1],
     xlab = "Theta values")
abline(v = c(confInt$lower_bound,confInt$cond_mean_est,confInt$upper_bound),
      col =1, lw = 3,lt=c(3,2,3))
# Add true value
abline(v = true_val,col =4, lw = 1,lt=c(1))
```