MAPTest

June 17, 2019

Title MAP Test for detection DE genes

Version 0.0.0.9000

Description MAP test for analyzing temporal gene expression data.

Depends R (>= 3.4.0)

License GPL-2

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

Imports EQL,MASS,foreach,mclust,mvtnorm,parallel,randtoolbox,stats,matlib

R topics documented:

`data_generation` *Simulate MAP data set:*

Description

Simulate MAP data set:

Usage

```
data_generation(G = 100, n_control = 10, n_treat = 10, n_rep = 3,
  k_real = 4, sigma2_r = rep(1, 2), sigma1_2_r = 1,
  sigma2_2_r = c(3, 2), mu1_r = 4, phi_g_r = rep(1, 100),
  p_k_real = c(0.7, 0.1, 0.1, 0.1), x = x)
```

Arguments

G	Number of genes to simulate
n_control	Number of time points in control group
n_treat	Number of time points in treatment group
n_rep	Number of replicates in both group
k_real	Always be 4
sigma2_r	Variance parameter for τ_g
sigma1_2_r	Variance parameter for η_{g1}
sigma2_2_r	Variance parameter for η_{g2}
mu1_r	Mean parameter for η_{g1}
phi_g_r	Dispersion parameter
p_k_real	True proportion for each mixture component
x	Time structured design for the simulated data

Details

The vector of read counts for gene g, treatment group i, replicate j, at time point t, $Y_{gij}(t)$, follows a Negative Binomial distribution parameterized mean λ_{gi} and ϕ_g, where $E[Y_{gij}(t)] = \lambda_{gi}(t)$. $\lambda_{gi}(t)$ is further modeled as $\lambda_{gi}(t) = S_{ij} \exp[\eta_{g1} I_{i=2} + B'(t)\eta_{g2} I_{i=2} + B'(t)\tau_g]$. We have $B'(t)$ are design matrix, which is constructed by 2 orthorgonal polynomial bases.

- t = 1,..., n_treat (or n_control if control group);
- j = 1,..., n_rep;
- g = 1,...,G; and
- $[\eta_{g1}, \eta_{g2}, \tau_g]$ ~ 4-component gausssian mixture model

Value

- Y1 Simulated data

Examples

```
library(matlib)
n_basis = 2
n_control = 10
n_treat   = 10
n_rep = 3
tt_treat  = c(1:n_treat)/n_treat
nt = length(tt_treat)
ind_t = sort(sample(c(1:nt), n_control))
tt = tt_treat[ind_t]
tttt = c(rep(tt, n_rep), rep(tt_treat, n_rep))
z = x = matrix(0, length(tttt), n_basis)
z[,1] = 1.224745*tttt
z[,2] = -0.7905694 + 2.371708*tttt^2
x[,1] = z[,1] - Proj(z[,1], rep(1, length(tttt)))
x[,2] = z[,2] - Proj(z[,2], rep(1, length(tttt))) - Proj(z[,2], x[,1])
Y1 = data_generation(G = 100,
                     n_control = n_control,
                     n_treat   = n_treat,
```

```
n_rep     = n_rep,
k_real = 4,
sigma2_r = rep(1, 2),
sigma1_2_r = 1,
sigma2_2_r = c(3,2),
mu1_r = 4,
phi_g_r = rep(1, 100),
p_k_real = c(0.7, 0.1, 0.1, 0.1),
x = x)
```

`estimation_fun` *Estimation for MAP data set:*

Description

Estimation for MAP data set:

Usage

```
estimation_fun(n_control = 10, n_treat = 10, n_rep = 3, x, Y1,
  k = 4, nn = 300, phi = NULL, type = NULL, tttt)
```

Arguments

`n_control`	Number of time points in control group
`n_treat`	Number of time points in treatment group
`n_rep`	Number of replicates in both group
`x`	Time structured design for the simulated data
`Y1`	RNA-seq data set: G by (n_control + n_treat) * n_rep matrix
`k`	Always 4
`nn`	Number of QMC nodes used in likelihood estimation
`phi`	the dispersion parameter
`type`	dispersion parameter estimation type
`tttt`	Time points used in the design

Details

The vector of read counts for gene g, treatment group i, replicate j, at time point t, $Y_{gij}(t)$, follows a Negative Binomial distribution parameterized mean λ_{gi} and ϕ_g, where $E[Y_{gij}(t)] = \lambda_{gi}(t)$. $\lambda_{gi}(t)$ is further modeled as $\lambda_{gi}(t) = S_{ij} \exp[\eta_{g1} I_{i=2} + B'(t)\eta_{g2} I_{i=2} + B'(t)\tau_g]$. We have $B'(t)$ are design matrix, which is constructed by 2 orthorgonal polynomial bases.

- t = 1,..., n_treat (or n_control if control group);
- j = 1,..., n_rep;
- g = 1,...,G; and
- $[\eta_{g1}, \eta_{g2}, \tau_g]$ ~ 4-component gausssian mixture model. We used latented negative binomial model with EM algorithm to estimate the paramters of mixture model.

Value

A list of parameters that we need to use for DE analysis.

data_use List includes the data information

- Y1 RNA-seq data set: G by (n_control + n_treat) * n_rep matrix
- start Starting point of the EM algorithm
- k always 4
- n_basis Number of basis function have been used to construct the design matrix
- X1 Vector indicate control data (0) or treatment data (1)
- x Design matrix been used for estimation
- tttt Time points used in the design

result1 List includes the Parameters estimations

- mu1 Mean parameter for η_{g1}
- sigma1_2 Variance parameter for η_{g1}
- sigma2 Variance parameter for τ_g
- sigma2_2 Variance parameter for η_{g2}
- p_k Proportion for each mixture component
- aa Dispertion parameter

Examples

```
library(matlib)
n_basis = 2
n_control = 10
n_treat   = 10
n_rep = 3
tt_treat  = c(1:n_treat)/n_treat
nt = length(tt_treat)
ind_t = sort(sample(c(1:nt), n_control))
tt = tt_treat[ind_t]
tttt = c(rep(tt, n_rep), rep(tt_treat, n_rep))
z = x = matrix(0, length(tttt), n_basis)
z[,1] = 1.224745*tttt
z[,2] = -0.7905694 + 2.371708*tttt^2
x[,1] = z[,1] - Proj(z[,1], rep(1, length(tttt)))
x[,2] = z[,2] - Proj(z[,2], rep(1, length(tttt))) - Proj(z[,2], x[,1])
Y1 = data_generation(G = 100,
                     n_control = n_control,
                     n_treat   = n_treat,
                     n_rep     = n_rep,
                     k_real = 4,
                     sigma2_r = rep(1, 2),
                     sigma1_2_r = 1,
                     sigma2_2_r = c(3,2),
                     mu1_r = 4,
                     phi_g_r = rep(1,100),
                     p_k_real = c(0.7, 0.1, 0.1, 0.1),
                     x = x)
```

```
aaa <- proc.time()
est_result <- estimation_fun(n_control = n_control,
                             n_treat = n_treat,
                             n_rep = n_rep,
                             x = x,
                             Y1 = Y1,
                             nn = 300,
                             k = 4,
                             phi = NULL,
                             type = 2,
                             tttt = tttt)
aaa1 <- proc.time()
aaa1 - aaa
#------------gaussian basis construction------------------
# method <- "gaussian"
# n_basis <- 2
# a = 1/4
# b = 2
# c = sqrt(a^2 + 2 * a * b)
# A = a + b + c
# B = b/A
# phi_cal = function(k, x){
#   Lambda_k =sqrt(2 * a / A) * B^k
#   1/(sqrt(sqrt(a/c) * 2^k * gamma(k+1))) *
#      exp(-(c-a) * x^2) * hermite(sqrt(2 * c) * x, k, prob = F)
#
# }
# z = do.call(cbind, lapply(c(1:n_basis), phi_cal, x = (tttt - mean(tttt))))
# x = matrix(0, length(tttt), n_basis)
# if(n_basis == 2){
#   x[,1] = z[,1] - Proj(z[,1], rep(1, length(tttt)))
#   x[,2] = z[,2] - Proj(z[,2], rep(1, length(tttt))) - Proj(z[,2], x[,1])
#
# }else{
#   x[,1] = z[,1] - Proj(z[,1], rep(1, length(tttt)))
#   x[,2] = z[,2] - Proj(z[,2], rep(1, length(tttt))) - Proj(z[,2], x[,1])
#  x[,3] = z[,3] - Proj(z[,3], rep(1, length(tttt))) - Proj(z[,3], x[,1]) - Proj(z[,3], x[,2])
# }
```

`MAP_test` *MAP test*

Description

MAP test

Usage

```
MAP_test(est_result, dd = NULL, Type = c(1:6), nn = 6000)
```

Arguments

`est_result`	estimation_fun result: A List contains data_use (the data information) and result1 (Parameters estimations)

dd	True index to check the true FDR
Type	Type of hypothesis • Type = 1 general DE detection • Type = 2 PDE with no time-by-treatment and NPDE with both treatment and time-by-treatment • Type = 3 NPDE with or without time-by-treatment • Type = 4 PDE with no time-by-treatment • Type = 5 NPDE with only time-by-treatment • Type = 6 NPDE with both treatment and time-by-treatment
nn	Number of QMC nodes used to estimate the test statistics

Details

The vector of read counts for gene g, treatment group i, replicate j, at time point t, $Y_{gij}(t)$, follows a Negative Binomial distribution parameterized mean λ_{gi} and ϕ_g, where $E[Y_{gij}(t)] = \lambda_{gi}(t)$. $\lambda_{gi}(t)$ is further modeled as $\lambda_{gi}(t) = S_{ij} \exp[\eta_{g1} I_{i=2} + B'(t)\eta_{g2} I_{i=2} + B'(t)\tau_g]$. We have $B'(t)$ are design matrix, which is constructed by 2 orthorgonal polynomial bases.

- t = 1,..., n_treat (or n_control if control group);
- j = 1,..., n_rep;
- g = 1,...,G; and
- $[\eta_{g1}, \eta_{g2}, \tau_g]$ ~ 4-component gausssian mixture model. After the parameter estimation we run the MAP test to detect the DE genes.

Value

A list of results for DE analysis.

- Type Type of hypothesis of interest
- FDR_final FDR estimation of each hypothesis
- FDR_real The true FDR if dd is known
- power Power if dd is known
- T_x Likelihood result for each component
- test_stat Test statistics for each hypothesis
- ct_final_all Test statistics cut-off value at each estimated FDR level

Examples

```
library(matlib)
n_basis = 2
n_control = 10
n_treat   = 10
n_rep = 3
tt_treat  = c(1:n_treat)/n_treat
nt = length(tt_treat)
ind_t = sort(sample(c(1:nt), n_control))
tt = tt_treat[ind_t]
tttt = c(rep(tt, n_rep), rep(tt_treat, n_rep))
z = x = matrix(0, length(tttt), n_basis)
```

```
z[,1] = 1.224745*tttt
z[,2] = -0.7905694 + 2.371708*tttt^2
x[,1] = z[,1] - Proj(z[,1], rep(1, length(tttt)))
x[,2] = z[,2] - Proj(z[,2], rep(1, length(tttt))) - Proj(z[,2], x[,1])
Y1 = data_generation(G = 100,
                     n_control = n_control,
                     n_treat   = n_treat,
                     n_rep     = n_rep,
                     k_real = 4,
                     sigma2_r = rep(1, 2),
                     sigma1_2_r = 1,
                     sigma2_2_r = c(3,2),
                     mu1_r = 4,
                     phi_g_r = rep(1,100),
                     p_k_real = c(0.7, 0.1, 0.1, 0.1),
                     x = x)
aaa <- proc.time()
est_result <- estimation_fun(n_control = n_control,
                             n_treat = n_treat,
                             n_rep = n_rep,
                             x = x,
                             Y1 = Y1,
                             nn = 300,
                             k = 4,
                             phi = NULL,
                             type = 2,
                             tttt = tttt)
aaa1 <- proc.time()
aaa1 - aaa

G <- 100
k_real <- 4
p_k_real <- c(0.7, 0.1, 0.1, 0.1)
dd = rep(c(0:(k_real-1)), p_k_real * G)
result = MAP_test(est_result = est_result, Type = c(1:6), dd = dd, nn = 300)
```

`Summary_MAP` *Summary for MAP Test*

Description

Summary for MAP Test

Usage

```
Summary_MAP(test_result, alpha = 0.05)
```

Arguments

`test_result`	MAP_test result: A list of results for DE analysis
`alpha`	Nominal level of FDR

Details

The vector of read counts for gene g, treatment group i, replicate j, at time point t, $Y_{gij}(t)$, follows a Negative Binomial distribution parameterized mean λ_{gi} and ϕ_g, where $E[Y_{gij}(t)] = lambda_{gi}(t)$. $\lambda_{gi}(t)$ is further modeled as $\lambda_{gi}(t) = S_{ij}exp[\eta_{g1}I_{i=2} + B'(t)\eta_{g2}I_{i=2} + B'(t)\tau_g]$ We have B'(t) are design matrix, which is constructed by 2 orthorgonal polynomial bases.

- t = 1,..., n_treat (or n_control if control group);
- j = 1,..., n_rep;
- g = 1,...,G; and
- $[\eta_{g1}, \eta_{g2}, \tau_g]$ ~ 4-component gausssian mixture model. After the parameter estimation we run the MAP test to detect the DE genes. At given nominal level, a list of differential genes are returned at a given hypothesis.

Value

A list of results for DE analysis.

- Type Type of hypothesis
- Reject_index DE genes list
- FDR_hat Estimated FDR

Examples

```
library(matlib)
set.seed(1)
n_basis = 2
n_control = 10
n_treat   = 10
n_rep = 3
tt_treat  = c(1:n_treat)/n_treat
nt = length(tt_treat)
ind_t = sort(sample(c(1:nt), n_control))
tt = tt_treat[ind_t]
tttt = c(rep(tt, n_rep), rep(tt_treat, n_rep))
z = x = matrix(0, length(tttt), n_basis)
z[,1] = 1.224745*tttt
z[,2] = -0.7905694 + 2.371708*tttt^2
x[,1] = z[,1] - Proj(z[,1], rep(1, length(tttt)))
x[,2] = z[,2] - Proj(z[,2], rep(1, length(tttt))) - Proj(z[,2], x[,1])
Y1 = data_generation(G = 100,
                     n_control = n_control,
                     n_treat   = n_treat,
                     n_rep     = n_rep,
                     k_real = 4,
                     sigma2_r = rep(1, 2),
                     sigma1_2_r = 1,
                     sigma2_2_r = c(3,2),
                     mu1_r = 4,
                     phi_g_r = rep(1,100),
                     p_k_real = c(0.7, 0.1, 0.1, 0.1),
                     x = x)
aaa <- proc.time()
est_result <- estimation_fun(n_control = n_control,
```

```
                              n_treat = n_treat,
                              n_rep = n_rep,
                              x = x,
                              Y1 = Y1,
                              nn = 300,
                              k = 4,
                              phi = NULL,
                              type = 2,
                              tttt = tttt)
aaa1 <- proc.time()
aaa1 - aaa

G <- 100
k_real <- 4
p_k_real <- c(0.7, 0.1, 0.1, 0.1)
dd = rep(c(0:(k_real-1)), p_k_real * G)
result = MAP_test(est_result = est_result, Type = c(1:6), dd = dd, nn = 300)
ss <- Summary_MAP(result)
```

Index