LTMG - DEG track (for differential expression analysis)

Chi Zhang
Yu Zhang

Example data

We will use Melissa Fishel's data as the example data. There are five separated data sets.

Basically, we may need the following steps for this analysis, let's use `H_si` and `H_sc` data as the example.

a standard data loading function and an condition index generation method

matrix_generation

```
tg_keys <- c("Fishel_scFPKM_sc1.txt", "Fishel_scFPKM_sc2.txt", "Fishel_scFPKM_si_APE1.txt",
  "Fishel_scFPKM_h_sc0.txt", "Fishel_scFPKM_h_si_APE1.txt")
tg_conds_meta <- cbind(c(0, 0, 1, 0, 1), c(0, 0, 0, 1, 1))
colnames(tg_conds_meta) <- c("Si", "H")
rownames(tg_conds_meta) <- tg_keys

Data_list <- list()
Stat_list <- list()
Data_0 <- c()
for (i in 1:length(tg_keys)) {
  Data_list[[i]] <- as.matrix(read.delim(tg_keys[i], row.names = 1))
  print(i)
  print(dim(Data_list[[i]]))
  Data_0 <- cbind(Data_0, Data_list[[i]])
}
```

```
## [1] 1
## [1] 18320    23
## [1] 2
## [1] 18320    23
## [1] 3
## [1] 18320    28
## [1] 4
## [1] 18320    48
## [1] 5
## [1] 18320    40
```

running LTMG for the complete data

Select genes

Genes with non-zero expression in more than 5 samples in `Data_0`

```
selected.genes <- which(rowSums(Data_0 > 0) > 5)
print(head(selected.genes))
```

```
## ENSG00000000003 ENSG00000000419 ENSG00000000457
##               1               2               3
## ENSG00000000460 ENSG00000001036 ENSG00000001084
##               4               5               6
```

Run LTMG for the selected genes

Run LTMG for `Data_0` `gene_stat_all` is a `Data_list[[i]]`*1 vector the number of peaks for each gene identified by LTMG over `Data_0`

```
library(LTMGSCA)

x <- Data_0[195, ]

for (k in 1:5) {
  print(LTMGSCA::LogSeparateKRpkmNew(x = x, n = 100, q = min(x[which(x > 0)]), k = k, err = 1e-10))
}
```

```
##        p mean  sd
## [1,] NaN  NaN NaN
##                p       mean        sd
## D12S92 0.7931081 -0.6102489 1.5938134
## E2S13  0.2068919  1.6048043 0.4993873
##                p       mean        sd
## F9S70 0.5806255 -0.9585484 0.7832846
## H6S48 0.3926061  1.3648701 0.6488160
## D2S12 0.0267684  2.6571838 0.6487617
##                     p       mean        sd
## C12S91     0.47635844 -1.1427364 0.8106132
## B2S10      0.26601735  0.5292171 1.0719811
## C3_tophat  0.23805952  1.5320515 0.5206351
## H12_tophat 0.01956469  2.8553404 0.5200554
##                 p        mean        sd
## G4S31  0.41526834 -1.20343113 0.7743622
## D12S92 0.20233370 -0.02906507 1.1657049
## H6S48  0.17389700  0.95321153 0.9365945
## E2S13  0.19136331  1.58686269 0.4754987
## B4S26  0.01713765  2.92507699 0.4814038
```

```
for (gene in head(selected.genes, 3)) {
  x <- Data_0[gene, ]
  for (k in 1:5) {
    print(LTMGSCA::LogSeparateKRpkmNew(x = x, n = 100, q = min(x[which(x > 0)]), k = k, err = 1e-10))
  }
}
```

```
##        p mean  sd
## [1,] NaN  NaN NaN
##                    p      mean        sd
## D11_tophat 0.5723978 -2.635281 3.0894008
## B11_tophat 0.4276022  3.325011 0.8752331
##                   p      mean        sd
## G2_tophat 0.4140136 -2.734409 0.9061793
## E7_tophat 0.3500458  3.624332 0.6632896
## C2_tophat 0.2359406  1.452750 1.0897895
```

```
##                     p      mean        sd
## G4S31     0.41008023 -2.601433 0.7735786
## B9_tophat 0.21378652  1.378048 1.1507408
## A5S33     0.33127201  3.656777 0.6462331
## B6_tophat 0.04486124  2.283542 0.9817910
##                      p       mean        sd
## D9_tophat  0.41497864 -2.5726165 0.7822301
## D11_tophat 0.14963057  0.9059196 0.7901081
## E7_tophat  0.18517254  2.9234073 0.6279871
## B11_tophat 0.21565872  3.9565514 0.5087092
## H6S48      0.03455954  2.4333741 0.7206418
##        p mean  sd
## [1,] NaN  NaN NaN
##                     p     mean        sd
## D10_tophat 0.3642125 2.256809 2.1311411
## H4S32      0.6357875 4.520245 0.6621172
##                     p        mean        sd
## F11_tophat 0.1298682 -0.06811241 2.9693379
## E3S21      0.6848896  4.58279664 0.6663931
## G7S55      0.1852422  2.63915453 0.7126864
##                     p      mean        sd
## E8_tophat 0.07931332 -1.583660 1.7115161
## C8_tophat 0.49267242  3.643572 1.2584320
## H7_tophat 0.39740739  4.721878 0.4584544
## B9_tophat 0.03060687  3.062840 1.2433374
##                      p      mean       sd
## B7S50      0.07272998 -1.549452 1.231939
## D10_tophat 0.31160877  3.158418 1.215911
## E3S21      0.23737559  4.342121 1.017228
## H4S32      0.35983483  4.732033 0.441166
## C8S59      0.01845084  3.049268 1.201308
##        p mean  sd
## [1,] NaN  NaN NaN
##                    p      mean       sd
## E5_tophat 0.8086962 -5.680103 3.489590
## E2_tophat 0.1913038  2.042491 1.083593
##                     p      mean        sd
## C10_tophat 0.6487922 -6.167631 1.3344321
## G5S39      0.1153366 -1.462307 0.6546129
## D11_tophat 0.2358712  1.883137 1.1527627
##                      p      mean        sd
## A11_tophat 0.65651094 -5.554585 1.2465579
## B1S2       0.10188836 -1.445609 0.5104039
## F11S86     0.15750312  1.302297 0.9829494
## H5S40      0.08409757  2.830150 0.8312725
##                     p      mean        sd
## H11S88    0.62666584 -5.654574 1.0556870
## E5_tophat 0.06439847 -2.012180 2.2060316
## G5S39     0.08243355 -1.457194 0.4866108
## E2_tophat 0.17386295  1.781468 1.1688350
## C9_tophat 0.05263919  2.282843 1.0655856
```

Here we have the BIC functions:

```
BIC_f_zcut <- function(y, rrr, Zcut) {
  n <- length(y)
  nparams <- nrow(rrr) * 3
  w <- rrr[, 1]
  u <- rrr[, 2]
  sig <- rrr[, 3]
  cc <- c()
  y0 <- y[which(y >= Zcut)]
  y1 <- y[which(y < Zcut)]
  y1 <- y1 * 0 + Zcut
  for (i in 1:nrow(rrr)) {
    c0 <- dnorm(y0, u[i], sig[i]) * w[i]
    c1 <- (1 - pnorm(y1, u[i], sig[i])) * w[i]
    c <- c(c0, c1)
    cc <- rbind(cc, c)
  }
  d <- apply(cc, 2, sum)
  e <- sum(log(d))
  f <- e * 2 - nparams * log(n)
  return (f)
}

BIC_f_zcut2 <- function(y, rrr, Zcut) {
  n <- length(y)
  nparams <- nrow(rrr) * 3
  w <- rrr[, 1]
  u <- rrr[, 2]
  sig <- rrr[, 3]
  y0 <- y[which(y >= Zcut)]
  cc <- c()
  for (i in 1:nrow(rrr)) {
    c <- dnorm(y0, u[i], sig[i]) * w[i]
    cc <- rbind(cc, c)
  }
  d <- apply(cc, 2, sum)
  e <- sum(log(d))
  f <- e * 2 - nparams * log(n)
  return (f)
}
```

We can now get `f` value using `BIC_f_zcut2()`.

```
x <- log(Data_0[3, ])
for (k in 1:5) {
  rrr <- LTMGSCA::SeparateKRpkmNew(x = x, n = 100, q = min(x[which(x > 0)]), k = k, err = 1e-10)
  print(BIC_f_zcut2(y = x, rrr, 0))
}
```

```
## [1] NaN
## [1] -239.9958
## [1] -249.6868
## [1] -263.7971
## [1] -278.6692
```

```r
GetAll <- function(x, n, q, err = 1e-10){
  max.k = 5
  bics <- rep(NA, max.k)
  results <- vector(mode = "list", length = max.k)
  for (k in 1:max.k) {
    results[[k]] <- LTMGSCA::SeparateKRpkmNew(x = x, n = n, q = q, k = k, err = err)
    bics[k] <- BIC_f_zcut2(y = x, results[[k]], q)
  }
  return(list(bics = bics, results = results))
}
```

```r
for (gene in head(selected.genes)) {
  x <- log(Data_0[gene, ])
  best <- GetAll(x = x, n = 100, q = min(x[which(x > 0)]), err = 1e-10)
  print(best$bics)
}
```

```
## [1]       NaN -416.9971 -430.2137 -444.9029 -459.2205
## [1]       NaN -502.9691 -516.4182 -530.9423 -546.1785
## [1]       NaN -239.9958 -249.6868 -263.7971 -278.6692
## [1]       NaN -240.9116 -254.0739 -269.2577 -285.1277
## [1]       NaN -360.5330 -375.4155 -390.9027 -406.1373
## [1]       NaN -322.4441 -333.2063 -342.9011 -357.0400
```

running LTMG-2LR for the genes fitted with less than 2 peaks in (ii)

For all the genes with `N==1,2`, Run LTMG2LR for all conditions (as an example, just for condition pair 1 and 2)

```r
matrix_generation_old <- function(tg_conds_meta, tg_ids) {
  tg_conds_meta0 <- tg_conds_meta[tg_ids, ]
  tg_cn <- colnames(tg_conds_meta)[1]
  tg_conds_meta1 <- as.matrix(tg_conds_meta0[, 1])
  for (i in 2:ncol(tg_conds_meta1)) {
    current.t0 <- tg_conds_meta0[, i]
    if (sum(abs(summary(lm(current.t0 ~ tg_conds_meta1 + 0))$residuals)) > 1e-10) {
      tg_conds_meta1 <- cbind(tg_conds_meta1, current.t0)
      tg_cn <- c(tg_cn, colnames(tg_conds_meta)[i])
      colnames(tg_conds_meta1) <- tg_cn
    }
  }
  for (i in 1:ncol(tg_conds_meta1)) {
    for (j in 1:ncol(tg_conds_meta1)) {
      if (i < j) {
        current.t0 <- tg_conds_meta1[, i] * tg_conds_meta1[, j]
        if (sum(abs(summary(lm(current.t0 ~ tg_conds_meta1 + 0))$residuals)) > 1e-10) {
          tg_cn <- c(tg_cn, c(paste(colnames(tg_conds_meta)[i],
          colnames(tg_conds_meta)[j], sep = "__")))
          tg_conds_meta1 <- cbind(tg_conds_meta1, current.t0)
          colnames(tg_conds_meta1) <- tg_cn
        }
      }
    }
  }
```

```
  return(tg_conds_meta1)
}

matrix_generation <- function(tg_conds_meta, tg_ids) {
  tg_conds_meta0 <- tg_conds_meta[tg_ids, ]
  if (ncol(tg_conds_meta) == 1) {
    tg_conds_meta0 <- as.matrix(tg_conds_meta0)
    colnames(tg_conds_meta0) <- colnames(tg_conds_meta)
  }
  tg_cn <- c()
  tg_conds_meta1 <- c()
  for (i in 1:ncol(tg_conds_meta0)) {
    current.t0 <- tg_conds_meta0[, i]
    if (length(tg_conds_meta1) == 0) {
      if ((sum(current.t0 == 0) > 0) & (sum(current.t0 == 1) > 0)) {
        tg_conds_meta1 <- cbind(tg_conds_meta1, current.t0)
        tg_cn <- c(tg_cn, colnames(tg_conds_meta)[i])
        colnames(tg_conds_meta1) <- tg_cn
      }
    } else {
      if (sum(abs(summary(lm(current.t0 ~ tg_conds_meta1 + 0))$residuals)) > 1e-05) {
        if ((sum(current.t0 == 0) > 0) & (sum(current.t0 == 1) > 0)) {
          tg_conds_meta1 <- cbind(tg_conds_meta1, current.t0)
          tg_cn <- c(tg_cn, colnames(tg_conds_meta)[i])
          colnames(tg_conds_meta1) <- tg_cn
        }
      }
    }
    # print(c(i,tg_cn))
  }
  colnames(tg_conds_meta1) <- tg_cn
  for (i in 1:ncol(tg_conds_meta0)) {
    for (j in 1:ncol(tg_conds_meta1)) {
      if (i < j) {
        current.t0 <- tg_conds_meta1[, i] * tg_conds_meta1[, j]
        if (sum(abs(summary(lm(current.t0 ~ tg_conds_meta1 + 0))$residuals)) > 1e-10) {
          if ((sum(current.t0 == 0) > 0) & (sum(current.t0 == 1) > 0)) {
          tg_cn <- c(tg_cn, c(paste(colnames(tg_conds_meta)[i],
            colnames(tg_conds_meta)[j], sep = "__")))
          tg_conds_meta1 <- cbind(tg_conds_meta1, current.t0)
          colnames(tg_conds_meta1) <- tg_cn
          }
        }
      }
    }
  }
  colnames(tg_conds_meta1) <- tg_cn
  return(tg_conds_meta1)
}
```

```
tg_ids <- 1:3

tg_keys0 <- c()
N <- 0
```

```
for (i in tg_ids) {
  N <- N + 1
  tg_keys0[[N]] <- tg_keys[i]
}

Data_c0 <- c()
for (i in 1:length(tg_ids)) {
  Data_c0 <- cbind(Data_c0, Data_list[[tg_ids[i]]])
}

tg_genes_n <- apply(Data_c0 != 0, 1, sum)
tg_genes <- names(tg_genes_n)[which(tg_genes_n > 10)]

Design_matrix0 <- matrix_generation(tg_conds_meta, 1:3)
```

Build_design_matrix_data_DGE is a function to generate data list by using the condition information from Design_matrix0.

```
Build_design_matrix_data_DGE <- function(Data_list, tg_conds_meta, tg_ids) {
  Design_matrix0 <- matrix_generation(tg_conds_meta, tg_ids)
  conds_index <- Design_matrix0[, 1] * 0
  for (i in 1:ncol(Design_matrix0)) {
    conds_index <- conds_index + Design_matrix0[, i] * 2 ^ i
  }
  conds_uniq <- unique(conds_index)

  conds_merged_data <- list()
  conds_merged_name <- c()
  Design_matrix_merged <- c()
  for (i in 1:length(conds_uniq)) {
    tg_ii <- which(conds_index == conds_uniq[i])
    data_c <- c()
    for (j in 1:length(tg_ii)) {
      data_c <- cbind(data_c, Data_list[[tg_ids[tg_ii[j]]]])
    }
    conds_merged_data[[i]] <- data_c
    Design_matrix_merged <- rbind(Design_matrix_merged, Design_matrix0[tg_ii[1], ])
    nn <- paste(colnames(Design_matrix0)[1], Design_matrix0[tg_ii[1], 1], sep = "=")
    if (ncol(Design_matrix0) > 1) {
      for (j in 2:ncol(Design_matrix0)) {
        nn <- paste(nn, paste(colnames(Design_matrix0)[j], Design_matrix0[tg_ii[1], j],
                              sep = "="), sep = "|")
      }
    }
    conds_merged_name <- c(conds_merged_name, nn)
  }
  rownames(Design_matrix_merged) <- conds_merged_name
  colnames(Design_matrix_merged) <- colnames(Design_matrix0)
  names(conds_merged_data) <- conds_merged_name
  ret <- list(Design_matrix0, Design_matrix_merged, conds_merged_data)
  names(ret) <- c("Design_matrix0", "Design_matrix_merged", "conds_merged_data")
  return(ret)
}
```

```
tg_ids <- 1:5
ret <- Build_design_matrix_data_DGE(Data_list, tg_conds_meta, tg_ids)
```

```
## Warning in summary.lm(lm(current.t0 ~ tg_conds_meta1 + 0)):
## essentially perfect fit: summary may be unreliable
```

```
Design_matrix0 <- ret[[1]]
Design_matrix_new <- ret[[2]]
Data_list_new <- ret[[3]]

Data_0 <- c()
Conds_meta <- c()
for (i in 1:length(Data_list_new)) {
  Data_0 <- cbind(Data_0, Data_list_new[[i]])
  Conds_meta <- cbind(Conds_meta, matrix(Design_matrix_new[i, ],
    ncol(Design_matrix_new), ncol(Data_list_new[[i]]), byrow = F))
}
rownames(Conds_meta) <- colnames(ret[[2]])
colnames(Conds_meta) <- colnames(Data_0)

print(Design_matrix0)
```

```
##                             Si H Si__H
## Fishel_scFPKM_sc1.txt        0 0     0
## Fishel_scFPKM_sc2.txt        0 0     0
## Fishel_scFPKM_si_APE1.txt    1 0     0
## Fishel_scFPKM_h_sc0.txt      0 1     0
## Fishel_scFPKM_h_si_APE1.txt  1 1     1
```

```
print(Design_matrix_new)
```

```
##                  Si H Si__H
## Si=0|H=0|Si__H=0  0 0     0
## Si=1|H=0|Si__H=0  1 0     0
## Si=0|H=1|Si__H=0  0 1     0
## Si=1|H=1|Si__H=1  1 1     1
```

```
Data_0 <- c()
for (i in 1:length(Data_list_new)) {
  Data_0 <- cbind(Data_0, Data_list_new[[i]])
}
tg_genes <- names(which(apply(Data_0 > 0, 1, sum) > 10))
```

Take the value of Zcut. For each vector, calculate Zcut, then the largest Zcut in these Zcuts can be used as Zcut running LTMG2LR.

```
LTMG2LR_DEG_test_new <- function(Data_conditions, Stat_list, Conds_meta, Design_matrix0, ROUNDS0 = 20) {
  unif_p_all <- generate_unif_p_matrix(Data_conditions, ROUNDS = ROUNDS0)
  print("General Statistics Setup: Done!")
  result_indi_stats <- c()
  result_data_stats <- c()
  length_data_test_stats <- c()
  gene_selected_names <- c()
  print("LTMR2LR DEG test: Start! Progress per 500 genes:")
  for (i in 1:length(Stat_list)) {
    if (length(Stat_list[[i]]) == length(Data_conditions)) {
```

```
    gene_selected_names <- c(gene_selected_names, names(Stat_list)[i])
    result_indi_list <- list()
    result_data_list <- list()
    length_data_test_k <- c()
    for (k in 1:ROUNDS0) {
      indi_test_c <- c()
      data_test_c <- c()
      length_data_test_c <- c()
      indi_all <- c()
      for (j in 1:length(Data_conditions)) {
        # gene_stat_c<-gene_stat_all[i,j]
        gene_data <- Data_conditions[[j]][i, ]
        ccc_stat <- Stat_list[[i]][[j]]
        unif_p_c <- unif_p_all[[j]][k, ]
        indi_c <- c()
        if (sum(gene_data != 0) <= 1) {
        indi_c <- gene_data * 0
        indi_c[which(gene_data != 0)] <- 1
        }
        if (sum(gene_data != 0) > 1) {
        gene_stat_c <- t(ccc_stat)
        y <- gene_data
        y0 <- log(y)
        Zcut0 <- min(y0[which(y != 0)])
        y0[which(y == 0)] <- Zcut0 - 2
        pp <- calculate_prob_sep_Zcut(y0, Zcut0, gene_stat_c[1, ], gene_stat_c[2, ], gene_stat_c[3, ]
        indi_c <- class_determination_2LR(pp, unif_p_c)
        cut_pos<-max(gene_stat_c[2, 2]-2*gene_stat_c[3, 2],gene_stat_c[2, 1])
        indi_c[which((apply(pp,2,sum)==0)&(y0>cut_pos))]<-1
        indi_c[which((apply(pp,2,sum)==0)&(y0<=cut_pos))]<-0
        }
        names(indi_c) <- names(gene_data)
        indi_all <- c(indi_all, indi_c)
        DE_c <- Design_matrix0[j, ]
        if (ncol(Design_matrix0) == 1) {
        names(DE_c) <- colnames(Design_matrix0)
        }
        if ((sum(gene_data > 0) > 2) & (sum(indi_c == 1) > 1)) {
        y <- gene_data
        y0 <- log(y)
        Zcut0 <- min(y0[which(y != 0)])
        y0[which(y == 0)] <- Zcut0 - 2
        log_data_c <- y0[which(indi_c == 1)]
        data_test_c <- rbind(data_test_c, build_design_data(DE_c, log_data_c))
        }
        length_data_test_c <- c(length_data_test_c, sum(indi_c == 1))
      }
      indi_test_c <- t(rbind(indi_all, Conds_meta))
      data_test_c <- as.data.frame(data_test_c)
      colnames(indi_test_c)[1] <- "Gene_data"
      indi_test_c <- as.data.frame(indi_test_c)
      mod <- summary(glm(Gene_data ~ ., family = "binomial", data = indi_test_c))$coefficients
      if (nrow(data_test_c) > 0) {
```

```
      mod2 <- summary(glm(Gene_data ~ ., family = "gaussian", data = data_test_c))$coefficients
    } else {
      mod2 <- ""
    }
    result_indi_list[[k]] <- mod
    result_data_list[[k]] <- mod2
    length_data_test_k <- rbind(length_data_test_k, length_data_test_c)
  }
  tg_r_f <- matrix(0, ncol(Design_matrix0), 2)
  tg_r_f[, 2] <- 2
  rownames(tg_r_f) <- colnames(Design_matrix0)
  colnames(tg_r_f) <- c("Sign", "p.value")
  tg_n <- c()
  tg_r <- c()
  for (ii in 1:ncol(Design_matrix0)) {
    ccc <- c()
    t <- 0
    for (j in 1:length(result_indi_list)) {
      if (sum(rownames(result_indi_list[[j]]) == colnames(Design_matrix0)[ii]) > 0) {
      ccc <- rbind(ccc, result_indi_list[[j]][colnames(Design_matrix0)[ii], c(1, 4)])
      t <- t + 1
      }
    }
    if (t > 0) {
      sign <- mean(sign(ccc[, 1]))
      pp <- median(ccc[, 2])
      tg_n <- c(tg_n, colnames(Design_matrix0)[ii])
      tg_r <- rbind(tg_r, c(sign, pp))
    }
  }
  rownames(tg_r) <- tg_n
  colnames(tg_r) <- c("Sign", "p.value")
  tg_r[which(is.na(tg_r))] <- 1
  tg_indi_r <- tg_r_f
  if (length(tg_n) > 0) {
    tg_indi_r[rownames(tg_r), ] <- tg_r
  }
  tg_n <- c()
  tg_r <- c()
  for (ii in 1:ncol(Design_matrix0)) {
    ccc <- c()
    t <- 0
    for (j in 1:length(result_data_list)) {
      if (sum(rownames(result_data_list[[j]]) == colnames(Design_matrix0)[ii]) > 0) {
      ccc <- rbind(ccc, result_data_list[[j]][colnames(Design_matrix0)[ii], c(1, 4)])
      t <- t + 1
      }
    }
    if (t > 0) {
      sign <- mean(sign(ccc[, 1]))
      pp <- median(ccc[, 2])
      tg_n <- c(tg_n, colnames(Design_matrix0)[ii])
      tg_r <- rbind(tg_r, c(sign, pp))
```

```
        }
      }
      tg_data_r <- tg_r_f
      if (length(tg_n) > 0) {
        rownames(tg_r) <- tg_n
        colnames(tg_r) <- c("Sign", "p.value")
        tg_r[which(is.na(tg_r))] <- 2
        tg_data_r[rownames(tg_r), ] <- tg_r
      }
      ccc1 <- as.vector(t(tg_indi_r))
      ccc2 <- as.vector(t(tg_data_r))
      if (i%%500 == 1) {
        print(i)
      }
      result_indi_stats <- rbind(result_indi_stats, ccc1)
      result_data_stats <- rbind(result_data_stats, ccc2)
      length_data_test_stats <- rbind(length_data_test_stats, apply(length_data_test_k, 2, mean))
    }
  }
  print("Test Done!\nResults Adjustment.")
  cn <- c()
  for (i in 1:ncol(Design_matrix0)) {
    cn <- c(cn, paste(c("Sign", "Pvalue"), colnames(Design_matrix0)[i], sep = "."))
  }
  colnames(result_indi_stats) <- cn
  rownames(result_indi_stats) <- gene_selected_names
  colnames(result_data_stats) <- cn
  rownames(result_data_stats) <- gene_selected_names
  colnames(length_data_test_stats) <- rownames(Design_matrix0)
  rownames(length_data_test_stats) <- gene_selected_names
  Reliable_Data_test_stats <- Reliable_Data_test(length_data_test_stats, Design_matrix0, num_cut = 4)
  result_data_stats_final <- adjust_result_data_stats(result_data_stats, Reliable_Data_test_stats)
  result_indi_stats_final <- adjust_result_indi_stats(result_indi_stats, Reliable_Data_test_stats)
  ccc <- list(result_indi_stats_final, result_data_stats_final)
  names(ccc) <- c("Bimodal test Result", "Expression level test Result")
  return(ccc)
  print("All Analysis Done!")
}

generate_unif_p_matrix <- function(Data_list, ROUNDS = 100) {
  unif_p_all <- list()
  for (i in 1:length(Data_list)) {
    unif_p_c <- matrix(runif(ncol(Data_list[[i]]) * ROUNDS, 0, 1), ROUNDS, ncol(Data_list[[i]]))
    colnames(unif_p_c) <- colnames(Data_list[[i]])
    rownames(unif_p_c) <- 1:ROUNDS
    unif_p_all[[i]] <- unif_p_c
  }
  names(unif_p_all) <- names(Data_list)
  return(unif_p_all)
}

class_determination_2LR <- function(p_table, p_ref) {
  p_table0 <- p_table
```

```
  cc <- apply(p_table, 2, sum)
  for (i in 1:nrow(p_table0)) {
    p_table0[i, ] <- p_table0[i, ]/cc
  }
  return((p_table0[1, ] < p_ref) * 1)
}

build_design_data <- function(Design_c, yy) {
  fff <- cbind(yy, matrix(Design_c, length(yy), length(Design_c), byrow = T))
  colnames(fff) <- c("Gene_data", names(Design_c))
  return(fff)
}

Reliable_Data_test <- function(length_data_test_stats0, Design_matrix0, num_cut = 4) {
  ccc <- c()
  for (i in 1:ncol(Design_matrix0)) {
    tg_s1 <- names(which(Design_matrix0[, i] == 0))
    tg_s2 <- names(which(Design_matrix0[, i] == 1))
    if (length(tg_s1) > 1) {
      ccc1 <- apply(length_data_test_stats0[, tg_s1], 1, sum)
    } else {
      ccc1 <- length_data_test_stats0[, tg_s1]
    }
    if (length(tg_s2) > 1) {
      ccc2 <- apply(length_data_test_stats0[, tg_s2], 1, sum)
    } else {
      ccc2 <- length_data_test_stats0[, tg_s2]
    }
    ccc <- cbind(ccc, (ccc1 >= num_cut) & (ccc2 >= num_cut) * 1)
  }
  ccc <- ccc * 1
  colnames(ccc) <- colnames(Design_matrix0)
  return(ccc)
}

adjust_result_data_stats <- function(result_data_stats, Reliable_Data_test_stats) {
  # par(mfcol=c(3,3))
  result_data_stats0 <- result_data_stats
  result_data_stats1 <- c()
  for (i in 1:ncol(Reliable_Data_test_stats)) {
    tg_id <- paste("Pvalue", colnames(Reliable_Data_test_stats)[i], sep = ".")
    tg_id1 <- paste("Sign", colnames(Reliable_Data_test_stats)[i], sep = ".")
    tg_id2 <- paste("FDR", colnames(Reliable_Data_test_stats)[i], sep = ".")
    # hist(result_data_stats[,tg_id],main=paste(tg_id,'Data Test\nOriginal P'),xlab='p, 2 for NA test',
    result_data_stats0[which(Reliable_Data_test_stats[, i] == 0), tg_id] <- 2
    # hist(result_data_stats0[,tg_id],main=paste(tg_id,'Data Test\nReliable P'),xlab='p, 2 for NA test'
    result_data_stats0[, tg_id1] <- sign(result_data_stats0[, tg_id1])
    result_data_stats1 <- cbind(result_data_stats1, result_data_stats0[, tg_id1])
    colnames(result_data_stats1)[ncol(result_data_stats1)] <- tg_id1
    result_data_stats1 <- cbind(result_data_stats1, result_data_stats0[, tg_id])
    colnames(result_data_stats1)[ncol(result_data_stats1)] <- tg_id
    result_data_stats0[which(result_data_stats0[, tg_id] <= 1), tg_id] <- p.adjust(result_data_stats0[wh
      tg_id] <= 1), tg_id], method = "fdr")
```

```
    result_data_stats1 <- cbind(result_data_stats1, result_data_stats0[, tg_id])
    colnames(result_data_stats1)[ncol(result_data_stats1)] <- tg_id2
    # hist(result_data_stats0[,tg_id],main=paste(tg_id,'Data Test\nReliable FDR'),xlab='FDR, 2 for NA t
  }
  return(result_data_stats1)
}

adjust_result_indi_stats <- function(result_indi_stats, Reliable_Data_test_stats) {
  # par(mfcol=c(2,3))
  result_indi_stats0 <- result_indi_stats
  result_indi_stats1 <- c()
  for (i in 1:ncol(Reliable_Data_test_stats)) {
    tg_id <- paste("Pvalue", colnames(Reliable_Data_test_stats)[i], sep = ".")
    tg_id1 <- paste("Sign", colnames(Reliable_Data_test_stats)[i], sep = ".")
    tg_id2 <- paste("FDR", colnames(Reliable_Data_test_stats)[i], sep = ".")
    # hist(result_indi_stats[,tg_id],main=paste(tg_id,'Data Test\nOriginal P'),xlab='p',col='lightblue'
    result_indi_stats0[, tg_id1] <- sign(result_indi_stats0[, tg_id1])
    result_indi_stats1 <- cbind(result_indi_stats1, result_indi_stats0[, tg_id1])
    colnames(result_indi_stats1)[ncol(result_indi_stats1)] <- tg_id1
    result_indi_stats1 <- cbind(result_indi_stats1, result_indi_stats0[, tg_id])
    colnames(result_indi_stats1)[ncol(result_indi_stats1)] <- tg_id
    result_indi_stats0[which(result_indi_stats0[, tg_id] <= 1), tg_id] <- p.adjust(result_indi_stats0[wh
      tg_id] <= 1), tg_id], method = "fdr")
    result_indi_stats1 <- cbind(result_indi_stats1, result_indi_stats0[, tg_id])
    colnames(result_indi_stats1)[ncol(result_indi_stats1)] <- tg_id2
    # hist(result_indi_stats0[,tg_id],main=paste(tg_id,'Data Test\nReliable FDR'),xlab='FDR, 2 for NA t
  }
  return(result_indi_stats1)
}
```

```
calculate_prob_sep_Zcut <- function(data1, Zcut, a, u, sig) {
  cc <- matrix(0, length(a), length(data1))
  colnames(cc) <- names(data1)
  for (i in 1:length(a)) {
    c <- a[i] / sig[i] * exp(-(data1 - u[i]) ^ 2 / (2 * sig[i] ^ 2))
    cc[i, ] <- c
  }
  cut_p <- rep(0, length(a))
  for (i in 1:length(a)) {
    cut_p[i] <- a[i] * pnorm(Zcut, u[i], sig[i])
  }
  for (i in 1:ncol(cc)) {
    if (data1[i] < Zcut) {
      cc[, i] <- cut_p
    }
  }
  cc[which(is.na(cc) == 1)] <- 0
  return(cc)
}
```

```
UB <- max(log(Data_0)) + 1
LB <- min(log(Data_0[which(Data_0 > 0)])) - 1
```

```r
M <- length(Data_list_new)
genes <- head(tg_genes)
results <- list()
for (gene in genes) {
  Zcut_c <- c()
  xx <- vector(mode = "list", length = M)
  for (j in 1:M) {
    data <- Data_list_new[[j]][gene, ]
    xx[[j]] <- log(data)
    ddd <- data[which(data != 0)]
    if(length(ddd) > 0) {
      Zcut_c <- c(Zcut_c, min(ddd))
    }
  }
  Zcut0 <- log(max(Zcut_c))
  if (max(sapply(xx, function (x) sum(x > Zcut0))) > 5) {
    results[[gene]] <- LTMGSCA::SeparateKRpkmNewLR(xx, 2500, Zcut0, 10, M = UB, m = LB)
  } else {
    warning(sprintf("The total number of longest elements after the cutoff in %s is %d, too small, skip
  }
}
```

```
## [1] 161
## [1] 153
## [1] 1390
## [1] 1094
## [1] 758
## [1] 2500
```

```r
print(results)
```

```
## $ENSG00000000003
## $ENSG00000000003[[1]]
##              p      mean       sd
## [1,] 0.3738087 -3.497825 3.228957
## [2,] 0.6261913  2.672815 1.326875
##
## $ENSG00000000003[[2]]
##              p      mean        sd
## [1,] 0.4357877 -3.497825 3.2289568
## [2,] 0.5642123  3.109634 0.9831824
##
## $ENSG00000000003[[3]]
##              p      mean        sd
## [1,] 0.4342959 -3.497825 3.2289568
## [2,] 0.5657041  3.456682 0.6227757
##
## $ENSG00000000003[[4]]
##              p      mean        sd
## [1,] 0.8578782 -3.497825 3.2289568
## [2,] 0.1421218  3.842686 0.7764387
##
##
## $ENSG00000000419
```

```
## $ENSG00000000419[[1]]
##              p     mean        sd
## [1,] 0.3425207 1.944359 1.1393016
## [2,] 0.6574793 4.515045 0.5492899
##
## $ENSG00000000419[[2]]
##               p     mean        sd
## [1,] 0.09317591 1.944359 1.1393016
## [2,] 0.90682409 4.664900 0.8492887
##
## $ENSG00000000419[[3]]
##      p     mean        sd
## [1,] 0 1.944359 1.1393016
## [2,] 1 4.150752 0.9511056
##
## $ENSG00000000419[[4]]
##              p     mean       sd
## [1,] 0.5230824 1.944359 1.139302
## [2,] 0.4769176 4.747897 0.689633
##
##
## $ENSG00000000457
## $ENSG00000000457[[1]]
##              p      mean       sd
## [1,] 0.5644109 -3.938694 1.283288
## [2,] 0.4355891  1.413680 1.586458
##
## $ENSG00000000457[[2]]
##              p      mean       sd
## [1,] 0.6119077 -3.938694 1.283288
## [2,] 0.3880923  1.912846 1.272533
##
## $ENSG00000000457[[3]]
##              p       mean       sd
## [1,] 0.7251829 -3.9386935 1.283288
## [2,] 0.2748171  0.4639882 1.916704
##
## $ENSG00000000457[[4]]
##               p      mean        sd
## [1,] 0.92511088 -3.938694 1.2832881
## [2,] 0.07488912  1.678747 0.5887797
##
##
## $ENSG00000000460
## $ENSG00000000460[[1]]
##              p      mean       sd
## [1,] 0.6553811 -3.213198 1.040950
## [2,] 0.3446189  2.113502 1.118085
##
## $ENSG00000000460[[2]]
##              p      mean       sd
## [1,] 0.6886989 -3.213198 1.040950
## [2,] 0.3113011  1.650884 1.451824
##
```

```
## $ENSG00000000460[[3]]
##              p      mean       sd
## [1,] 0.7898557 -3.213198 1.040950
## [2,] 0.2101443  2.033720 1.027467
##
## $ENSG00000000460[[4]]
##              p      mean        sd
## [1,] 0.8252106 -3.213198 1.0409495
## [2,] 0.1747894  1.134048 0.4399927
##
##
## $ENSG00000001036
## $ENSG00000001036[[1]]
##              p      mean       sd
## [1,] 0.4776226 -1.831094 1.107425
## [2,] 0.5223774  2.275612 1.053380
##
## $ENSG00000001036[[2]]
##              p      mean        sd
## [1,] 0.6788949 -1.831094 1.1074251
## [2,] 0.3211051  2.929435 0.6927851
##
## $ENSG00000001036[[3]]
##              p      mean       sd
## [1,] 0.4393235 -1.831094 1.107425
## [2,] 0.5606765  2.043461 1.199349
##
## $ENSG00000001036[[4]]
##              p      mean        sd
## [1,] 0.7743784 -1.831094 1.1074251
## [2,] 0.2256216  3.185454 0.9484531
##
##
## $ENSG00000001084
## $ENSG00000001084[[1]]
##              p        mean         sd
## [1,] 0.4976996 -0.09891725 0.07972853
## [2,] 0.5023004  1.16576865 1.43612951
##
## $ENSG00000001084[[2]]
##              p        mean         sd
## [1,] 0.5238312 -0.09891725 0.07972853
## [2,] 0.4761688  2.40522875 1.15081210
##
## $ENSG00000001084[[3]]
##             p        mean         sd
## [1,] 0.624143 -0.09891725 0.07972853
## [2,] 0.375857  2.30985457 0.78911564
##
## $ENSG00000001084[[4]]
##              p        mean         sd
## [1,] 0.8188561 -0.09891725 0.07972853
## [2,] 0.1811439  2.74820565 1.38366547
```

```r
save(results, file = "deg.head.RData")
```

```r
load("deg.head.RData")
LTMG_2LR_test_results <- LTMG2LR_DEG_test_new(Data_conditions = Data_list_new, Stat_list = results, Cond
```

```
## [1] "General Statistics Setup: Done!"
## [1] "LTMR2LR DEG test: Start! Progress per 500 genes:"
## [1] 1
## [1] "Test Done!\nResults Adjustment."
```

```r
LTMG_2LR_test_results
```

```
## $`Bimodal test Result`
##                  Sign.Si    Pvalue.Si       FDR.Si Sign.H
## ENSG00000000003      -1 0.5488206054 0.7106359695     -1
## ENSG00000000419       1 0.0294258661 0.0882775984      1
## ENSG00000000457      -1 0.5921966413 0.7106359695     -1
## ENSG00000000460      -1 0.8159206489 0.8159206489     -1
## ENSG00000001036      -1 0.0957977857 0.1915955715      1
## ENSG00000001084      -1 0.0001129087 0.0006774521      1
##                    Pvalue.H     FDR.H Sign.Si__H
## ENSG00000000003 0.56171732 0.9911513         -1
## ENSG00000000419 0.99026578 0.9911513         -1
## ENSG00000000457 0.07834325 0.4018299         -1
## ENSG00000000460 0.13394330 0.4018299         -1
## ENSG00000001036 0.69179037 0.9911513         -1
## ENSG00000001084 0.99115128 0.9911513         -1
##                  Pvalue.Si__H  FDR.Si__H
## ENSG00000000003   0.01115951 0.06695708
## ENSG00000000419   0.98907441 0.99310793
## ENSG00000000457   0.13822437 0.41467311
## ENSG00000000460   0.89678874 0.99310793
## ENSG00000001036   0.37709759 0.75419518
## ENSG00000001084   0.99310793 0.99310793
##
## $`Expression level test Result`
##                  Sign.Si    Pvalue.Si       FDR.Si Sign.H
## ENSG00000000003       1 2.229298e-01 0.3974151619      1
## ENSG00000000419       1 4.888317e-01 0.4888316602     -1
## ENSG00000000457       1 3.311793e-01 0.3974151619     -1
## ENSG00000000460      -1 3.125128e-01 0.3974151619     -1
## ENSG00000001036       1 1.345680e-01 0.3974151619     -1
## ENSG00000001084       1 2.602198e-05 0.0001561319      1
##                     Pvalue.H      FDR.H Sign.Si__H
## ENSG00000000003 0.006825854 0.04095513          0
## ENSG00000000419 0.049976972 0.14993092          1
## ENSG00000000457 0.242886378 0.48577276          1
## ENSG00000000460 0.943734547 0.94991082         -1
## ENSG00000001036 0.446700711 0.67005107          1
## ENSG00000001084 0.949910819 0.94991082         -1
##                  Pvalue.Si__H    FDR.Si__H
## ENSG00000000003 8.223174e-01 8.223174e-01
## ENSG00000000419 1.867969e-01 4.669922e-01
## ENSG00000000457 2.000000e+00 2.000000e+00
## ENSG00000000460 5.568584e-01 6.960730e-01
```

```
## ENSG00000001036 4.277317e-01 6.960730e-01
## ENSG00000001084 6.284695e-06 3.142348e-05
```

```
M <- length(Data_list_new)
genes <- head(tg_genes)
results <- list()
for (gene in c("ENSG00000138698", "ENSG00000124243", "ENSG00000067606", "ENSG00000064490")) {
  Zcut_c <- c()
  xx <- vector(mode = "list", length = M)
  for (j in 1:M) {
    data <- Data_list_new[[j]][gene, ]
    xx[[j]] <- log(data)
    ddd <- data[which(data != 0)]
    if(length(ddd) > 0) {
      Zcut_c <- c(Zcut_c, min(ddd))
    }
  }
  Zcut0 <- log(max(Zcut_c))
  if (max(sapply(xx, function (x) sum(x > Zcut0))) > 5) {
    results[[gene]] <- LTMGSCA::SeparateKRpkmNewLR(xx, 2500, Zcut0, 10, M = UB, m = LB)
  } else {
    warning(sprintf("The total number of longest elements after the cutoff in %s is %d, too small, skip
  }
}
```

```
## [1] 78
## [1] 2500
## [1] 2500
## [1] 794
```

```
print(results)
```

```
## $ENSG00000138698
## $ENSG00000138698[[1]]
##              p      mean        sd
## [1,] 0.1023794 -1.269086 2.6845542
## [2,] 0.8976206  3.960338 0.6939969
##
## $ENSG00000138698[[2]]
##              p      mean        sd
## [1,] 0.1273337 -1.269086 2.6845542
## [2,] 0.8726663  3.942644 0.8822589
##
## $ENSG00000138698[[3]]
##               p      mean       sd
## [1,] 0.07737872 -1.269086 2.684554
## [2,] 0.92262128  4.056912 0.638288
##
## $ENSG00000138698[[4]]
##               p      mean        sd
## [1,] 0.06025366 -1.269086 2.6845542
## [2,] 0.93974634  4.184985 0.7144058
##
##
## $ENSG00000124243
```

```
## $ENSG00000124243[[1]]
##      p       mean        sd
## [1,] 0 -31.264611 9.2765074
## [2,] 1   1.223014 0.8828217
##
## $ENSG00000124243[[2]]
##             p       mean        sd
## [1,] 0.129454 -31.264611 9.2765074
## [2,] 0.870546   1.273955 0.9521927
##
## $ENSG00000124243[[3]]
##             p       mean        sd
## [1,] 0.7078486 -31.264611 9.2765074
## [2,] 0.2921514   1.303262 0.5464833
##
## $ENSG00000124243[[4]]
##             p       mean        sd
## [1,] 0.7980034 -31.264611 9.2765074
## [2,] 0.2019966   1.638236 0.8193597
##
##
## $ENSG00000067606
## $ENSG00000067606[[1]]
##               p      mean      sd
## [1,] 0.91304102 -1.453921 0.36346
## [2,] 0.08695898  2.133417 0.67148
##
## $ENSG00000067606[[2]]
##             p       mean       sd
## [1,] 0.7625359 -1.4539205 0.363460
## [2,] 0.2374641  0.3027226 0.616358
##
## $ENSG00000067606[[3]]
##               p      mean      sd
## [1,] 0.97916667 -1.453921 0.36346
## [2,] 0.02083333  1.806124 0.05000
##
## $ENSG00000067606[[4]]
##         p      mean      sd
## [1,] 0.95 -1.453921 0.36346
## [2,] 0.05  1.738299 0.05000
##
##
## $ENSG00000064490
## $ENSG00000064490[[1]]
##             p      mean        sd
## [1,] 0.8331979 0.2250071 0.8208021
## [2,] 0.1668021 2.8356994 0.3335933
##
## $ENSG00000064490[[2]]
##               p      mean        sd
## [1,] 0.93093911 0.2250071 0.8208021
## [2,] 0.06906089 2.2936973 0.0500000
##
```

```
## $ENSG00000064490[[3]]
##              p      mean        sd
## [1,] 0.7742321 0.2250071 0.8208021
## [2,] 0.2257679 3.1363247 0.4164883
##
## $ENSG00000064490[[4]]
##               p      mean         sd
## [1,] 0.95065417 0.2250071 0.82080211
## [2,] 0.04934583 2.9667406 0.08577783
```

```
if (file.exists("deg.RData")){
  load("deg.RData")
} else {
  M <- length(Data_list_new)
  genes <- head(tg_genes)
  results <- list()
  library(doParallel)
  registerDoParallel(cores = 63)
  system.time(results <- foreach (gene = 1:length(Data_list_new[[1]][,1])) %dopar% {
    Zcut_c <- c()
    xx <- vector(mode = "list", length = M)
    for (j in 1:M) {
      data <- Data_list_new[[j]][gene, ]
      xx[[j]] <- log(data)
      ddd <- data[which(data != 0)]
      if(length(ddd) > 0) {
        Zcut_c <- c(Zcut_c, min(ddd))
      }
    }
    Zcut0 <- log(max(Zcut_c))
    if (max(sapply(xx, function (x) sum(x > Zcut0))) > 5) {
      result <- LTMGSCA::SeparateKRpkmNewLR(xx, 2500, Zcut0, 10, M = UB, m = LB)
    } else {
      warning(sprintf("The total number of longest elements after the cutoff in %s is %d, too small, ski
      NA
    }
  })
  names(results) <- row.names(Data_list_new[[1]])
  save(results, file = "deg.RData")
}
```

```
if (file.exists("LTMG_2LR_test_results.RData")) {
  load("LTMG_2LR_test_results.RData")
} else {
  LTMG_2LR_test_results <- LTMG2LR_DEG_test_new(Data_conditions = Data_list_new, Stat_list = results, C
  save(LTMG_2LR_test_results, file = "LTMG_2LR_test_results.RData")
}
head(LTMG_2LR_test_results[[1]])
```

```
##                  Sign.Si    Pvalue.Si      FDR.Si Sign.H
## ENSG00000000003      -1 0.7373114893 1.000000000     -1
## ENSG00000000419       1 0.0212521943 0.110001220      1
## ENSG00000000457      -1 0.7230569531 0.992222965     -1
## ENSG00000000460      -1 0.8159206489 1.000000000     -1
## ENSG00000001036      -1 0.0957977857 0.300654953      1
```

```
## ENSG00000001084      -1 0.0001129087 0.002269553      1
##                   Pvalue.H     FDR.H Sign.Si__H Pvalue.Si__H
## ENSG00000000003 0.5944713 0.9026139         -1  0.008951548
## ENSG00000000419 0.9902658 1.0000000         -1  0.989075964
## ENSG00000000457 0.1013051 0.2948821         -1  0.100710344
## ENSG00000000460 0.1339433 0.3573691         -1  0.896788737
## ENSG00000001036 0.6917904 0.9699528         -1  0.344736871
## ENSG00000001084 0.9911513 1.0000000         -1  0.993107931
##                 FDR.Si__H
## ENSG00000000003 0.1141071
## ENSG00000000419 1.0000000
## ENSG00000000457 0.4594235
## ENSG00000000460 1.0000000
## ENSG00000001036 0.8567342
## ENSG00000001084 1.0000000
```