"A simple testing procedure for multiseq"

Heejung Shim

February 13, 2016

Contents

1 Overview

- After making changes in a software package multiseq, we run a set of tests to check if the changes in multiseq affect output as we planned (e.g., changes shouldn't affect other outputs).
- We simulate four sets of data - typical data and data with no reads with/without library read depth.

2 Simulation of data set

2.1 Collect information to simulate data

I'll copy phenotype data (from which we will sample data) and signals for group 1 and group 2. See compareWaveQTLandmultiseq_dsQTL_final.org for detailed description of how to obtain this information.

```
cd ~/multiseq/
mkdir data
```

```
cd data/
mkdir simulation
cd simulation/
mkdir dsQTL
cp ~/multiscale_analysis/analysis/simulation/sample_size/simulation_QTLfinal_v2
    /data/pheno.dat dsQTL
cp ~/multiscale_analysis/analysis/simulation/sample_size/simulation_QTLfinal_v2
    /data/alt.sig0 dsQTL
cp ~/multiscale_analysis/analysis/simulation/sample_size/simulation_QTLfinal_v2
    /data/alt.sig1 dsQTL
```

I'll copy library read depth for this DNase-seq data (you can download from WaveQTL repo) to this directory.

```
cp ~/WaveQTL/data/dsQTL/library.read.depth.dat ~/multiseq/data/simulation/dsQTL
    /
```

Information have been save in:

```
cd ~/multiseq/data/simulation/dsQTL
alt.sig0 alt.sig1 library.read.depth.dat pheno.dat
```

2.2 Simulate data

The function 'simulate.data' is a modification of the script in multiscale_analysis repo. This script uses functions in 'my.utils.R' in multiscale_analysis repo.

```
setwd("~/multiseq/data/simulation/dsQTL/")
source("~/multiscale_analysis/src/R/my.utils.R")
##' 'simulate.data' simulate data sets by resampling reads from real data (real
    .read.counts) for given signals (sig0, sig1).
##'
##' @param seed seed number to set up
##' @param numGroup0 number of samples for Group0
##' @param numGroup1 number of samples for Group1
##' @param real.read.counts a vector of size T (e.g., 1024); t-th element
    contains number of reads at t-th position; from which we will resample
    reads for simulation;
```

```
##' @param sig0 a vector of size T (e.g., 1024); t-th element contains
    probability of sampling read at t-th position in real.read.counts for
    Group0
##' @param sig1 a vector of size T (e.g., 1024); t-th element contains
    probability of sampling read at t-th position in real.read.counts for
    Group1
##' @param real.library.read.depth a vector of size M (>1); from whch we will
    sample library read depth
##' @param over.dispersion parameter used in sample.from.Binomial.with.
    Overdispersion; see 'my.utils.R' for details.
##' @return a list of data, group, library.read.depth; data contains simulated
    data; matrix of (numGroup0+numGroup1) by T; group contains group indicator
    for simulated data; a vector of size (numGroup0+numGroup1); library.read.
    depth contains simulated library read depth; a vector of size (numGroup0+
    numGroup1)
simulate.data <- function(seed = 1, numGroup0, numGroup1, sig0, sig1, real.read
    .counts, real.library.read.depth = NULL, over.dispersion=NULL){

  genoD = c(rep(0, numGroup0), rep(1, numGroup1))
  numSam = length(genoD)
  numBPs = length(sig0)

  ## phenotype data
  phenoD = matrix(data=NA, nr= length(genoD), nc = numBPs)

  ## let's sample!!!
  set.seed(seed)

  ## upper and lower bound!
  trunc.fun = function(x){
    x = max(0, x)
    return(min(1,x))
  }
  p.sig0 = sapply(sig0, trunc.fun)
  p.sig1 = sapply(sig1, trunc.fun)

  ## geno = 0
  wh0 = which(genoD == 0)
  if(length(wh0) > 0){
    phenoD[wh0,] = sample.from.Binomial.with.Overdispersion(num.sam = length(
```

```
      wh0), total.count = real.read.counts, mu.sig = p.sig0, over.dispersion =
       over.dispersion)
  }
  ## geno = 1
  wh1 = which(genoD == 1)
  if(length(wh1) > 0){
    phenoD[wh1,] = sample.from.Binomial.with.Overdispersion(num.sam = length(
      wh1), total.count = real.read.counts, mu.sig = p.sig1, over.dispersion =
       over.dispersion)
  }
  if(is.null(real.library.read.depth)){
    library.read.depth=NULL
  }else{
    library.read.depth = sample(real.library.read.depth, numSam, replace = TRUE
      )
  }
  return(list(data = phenoD, group = genoD, library.read.depth = library.read.
    depth))
}

seed = 1
numGroup0 = 10
numGroup1 = 10
sig0 = scan("~/multiseq/data/simulation/dsQTL/alt.sig0", what=double())
sig1 = scan("~/multiseq/data/simulation/dsQTL/alt.sig1", what=double())
real.DNase.dat = read.table("~/multiseq/data/simulation/dsQTL/pheno.dat", as.is
    = TRUE)
real.read.counts = ceiling(as.numeric(apply(real.DNase.dat, 2, sum)))
real.library.read.depth = scan("~/multiseq/data/simulation/dsQTL/library.read.
   depth.dat", what=double())

## data with sample size 20 with library read depth
res = simulate.data(seed = seed, numGroup0 = numGroup0, numGroup1 = numGroup1,
   sig0 = sig0, sig1= sig1, real.read.counts = real.read.counts, real.library.
   read.depth = real.library.read.depth, over.dispersion=NULL)
str(res)
## List of 3
## $ data : int [1:20, 1:1024] 0 0 0 0 0 0 0 0 0 0 ...
## $ group : num [1:20] 0 0 0 0 0 0 0 0 0 0 ...
## $ library.read.depth: num [1:20] 44257311 37331440 30843655 36823292
```

```
    50966695 ...
apply(res$data,1,sum)
## [1] 65 60 63 62 81 75 57 65 57 66 39 40 49 46 44 48 53 43 37 51

## data with sample size 20 with library read depth, but all samples in group 2
     have zero read count.
res = simulate.data(seed = seed, numGroup0 = numGroup0, numGroup1 = numGroup1,
    sig0 = sig0, sig1= rep(0, length(sig1)), real.read.counts = real.read.
    counts, real.library.read.depth = real.library.read.depth, over.dispersion=
    NULL)
apply(res$data,1, sum)
## [1] 65 60 63 62 81 75 57 65 57 66 0 0 0 0 0 0 0 0 0 0

## data with sample size 20 without library read depth
res = simulate.data(seed = seed, numGroup0 = numGroup0, numGroup1 = numGroup1,
    sig0 = sig0, sig1= sig1, real.read.counts = real.read.counts, over.
    dispersion=NULL)
apply(res$data,1,sum)
## [1] 65 60 63 62 81 75 57 65 57 66 39 40 49 46 44 48 53 43 37 51
res$library.read.depth
## NULL

## data with sample size 20 without library read depth, but all samples in
    group 2 have zero read count.
res = simulate.data(seed = seed, numGroup0 = numGroup0, numGroup1 = numGroup1,
    sig0 = sig0, sig1= rep(0, length(sig1)), real.read.counts = real.read.
    counts, over.dispersion=NULL)
apply(res$data,1,sum)
## [1] 65 60 63 62 81 75 57 65 57 66 0 0 0 0 0 0 0 0 0 0
res$library.read.depth
##NULL
```