Package ‘ssize’

February 1, 2017

Title Estimate Microarray Sample Size
Version 1.48.0
Date 2012-06-07
Author Gregory R. Warnes, Peng Liu, and Fasheng Li
Description Functions for computing and displaying sample size information for gene expression arrays.
Maintainer Gregory R. Warnes <greg@random-technologies-llc.com>
Depends gdata, xtable
License LGPL
biocViews Microarray, DifferentialExpression
NeedsCompilation no

R topics documented:

exp.sd ................................................................. 1
pow ................................................................. 2
power.t.test.FDR ................................................ 4

Index

exp.sd Example baseline variability for gene expression experiment

Description

Example baseline variability for gene expression experiment

Usage

data(exp.sd)

Format

Vector of 12,625 standard deviations of gene expression data normalized via the RMA method (ie on log2 scale) with names from Affymetrix probe set IDs.
Examples

data(exp.sd)

hist(exp.sd, prob=TRUE)
lines(density(exp.sd), col="red", lwd=2)

pow

Compute and plot power, required sample-size, or detectible effect size for gene expression experiment

Description

Compute and plot power, required sample-size, or detectible effect size for gene expression experiment

Usage

pow(sd, n, delta, sig.level, alpha.correct = "Bonferonni")
pow.plot(x, xlab = "Power", ylab = "Proportion of Genes with Power >= x", marks = c(0.7, 0.8, 0.9), ...)

ssize(sd, delta, sig.level, power, alpha.correct = "Bonferonni")
ssize.plot(x, xlab = "Sample Size (per group)", ylab = "Proportion of Genes Needing Sample Size <= n", marks = c(2, 3, 4, 5, 6, 8, 10, 20), ...)

delta(sd, n, power, sig.level, alpha.correct = "Bonferonni")
delta.plot(x, xlab = "Fold Change", ylab = "Proportion of Genes with Power >= 80% at Fold Change=delta", marks = c(1.5, 2, 2.5, 3, 4, 6, 10), ...)

Arguments

sd Vector of standard deviations for control samples, *on the log2 scale*

n Number of observations (per group)

delta Hypothetical True difference in expression, on the log2 scale.
sig.level Significance level (Type I error probability)

power Power

alpha.correct Type of correction for multiple comparison. One of "Bonferonni" or "None".

x Vector of powers generated by pow

xlab, ylab x and y axis labels

marks Powers at which percent of genes achieving the specified cutoff is annotated on the plot.

... Additional graphical parameters
The `pow` function computes power for each element of a gene expression experiment using an vector of estimated standard deviations. The power is computed separately for each gene, with an optional correction to the significance level for multiple comparison. The `power.plot` function generates a cumulative power plot illustrating the fraction and number of genes achieve a given power for the specified sample size, significance level, and delta. Periods are printed for every 10 calculations so that the user can see that the computation is proceeding.

**Value**

`pow` returns a vector containing the power for each standard deviation.

**Note**

This code was intended to be used with data are on the log2 scale, in which case the delta can be set to becomes log2(fold-change).

**Author(s)**

Gregory R. Warnes <greg@warnes.net>

**References**


**See Also**

`ssize, ssize.plot, delta, delta.plot`

**Examples**

```r
library(gdata) # for nobs()

data(exp.sd)

# Histogram of the standard deviations

hist(exp.sd,n=20, col="cyan", border="blue", main="", 
  xlab="Standard Deviation (for data on the log scale)"
)
dens <- density(exp.sd)
lines(dens$x, dens$y*par("usr")[4]/max(dens$y),col="red",lwd=2)

title("Histogram of Standard Deviations")

# 1) What is the power if using 6 patients 3 measurements assuming 
# Delta=1.0, Alpha=0.05 and Observed SDs?
```
n=6; fold.change=2.0; power=0.8; sig.level=0.05;
#
all.power <- pow(sd=exp.sd, n=n, delta=log2(fold.change),
sig.level=sig.level)

title("Power to Detect 2-Fold Change")

# 2) What is necessary sample size for 80% power using 3 measurements/patient
# assuming Delta=1.0, Alpha=0.05 and Observed SDs?
#
all.size <- ssize(sd=exp.sd, delta=log2(fold.change),
sig.level=sig.level, power=power)

title("Sample Size to Detect 2-Fold Change")

# 3) What is necessary fold change to achieve 80% power using 3
# measurements/patient assuming n=6, Delta=1.0, Alpha=0.05 and Observed
# SDs?
#
all.delta <- delta(sd=exp.sd, power=power, n=n, sig.level=sig.level)

title("Fold Change to Achieve 80% Power")

power.t.test.FDR

---

**Description**

Compute power of test, or determine parameters to obtain target power.
Usage

power.t.test.FDR(sd=1, n=NULL, delta=NULL, FDR.level=0.05, pi0, power=NULL, type=c("two.sample", "one.sample", "paired"), alternative=c("two.sided", "one.sided") )

Arguments

sd Standard deviation

n Number of observations (per group)

delta True difference in means

FDR.level False Discovery Rate (expected ratio of false discoveries among all discoveries)

pi0 Proportion of true null hypotheses (fraction of tests that with no difference)

power Power of test (1 minus Type II error probability)

type Type of t test

alternative One- or two-sided test

Details

Exactly one of the parameters n, delta, power, sd, and FDR.level must be passed as NULL, and that parameter is determined from the others. Notice that the last two have non-NULL defaults so NULL must be explicitly passed if you want to compute them.

Value

Object of class "power.htest", a list of the arguments (including the computed one) augmented with method and note elements.

Note

uniroot is used to solve power equation for unknowns, so you may see errors from it, notably about inability to bracket the root when invalid arguments are given.

Author(s)

Peng Liu, based on power.t.test code by Peter Dalgaard, which in turn is based on previous work by Claus Ekström

See Also

t.test, uniroot

Examples

## Compute power given sd, n, delta, FDR & pi.0
power.t.test.FDR(sd=1, n=5, delta=2, FDR.level=0.05, pi0=0.95, power=NULL, type="two.sample", alternative="two.sided")

## Compute power
power.t.test.FDR(n=20, delta=1, FDR=0.05, pi0=0.75)
power.t.test.FDR(n=29, delta=1, FDR=0.05, pi0=0.75)

## compute n
power.t.test.FDR(n=NULL, sd=1, power=.90, delta=1, FDR=0.05, pi0=0.975)
power.t.test.FDR(n=NULL, sd=1, power=.90, delta=1, FDR=0.05, pi0=0.975,
  alt="one.sided")

## compute sd
power.t.test.FDR(sd=NULL, n=29, power=.90, delta=1, FDR=0.05, pi0=0.975)

## compute FDR level
power.t.test.FDR(sd=1, n=29, power=.90, delta=1, FDR=NULL, pi0=0.975)
Index

*Topic **datasets**
  exp.sd, 1

*Topic **design**
  pow, 2

*Topic **htest**
  pow, 2
  power.t.test.FDR, 4

delta, 3
delta(pow), 2
delta.plot, 3

exp.sd, 1

pow, 2
power.plot(pow), 2
power.t.test.FDR, 4

ssize, 3
ssize(pow), 2
ssize.plot, 3

t.test, 5

unirroot, 5