Package 'clippda'

November 11, 2026

Title A package for the clinical proteomic profiling data analysis

Version 1.62.0

Date 2011-09-16

Author Stephen Nyangoma

Description Methods for the nalysis of data from clinical proteomic profiling studies. The focus is on the studies of human subjects, which are often observational case-control by design and have technical replicates. A method for sample size determination for planning these studies is proposed. It incorporates routines for adjusting for the expected heterogeneities and imbalances in the data and the within-sample replicate correlations.

Depends R (>= 2.13.1),limma, statmod, rgl, lattice, scatterplot3d, graphics, grDevices, stats, utils, Biobase, tools, methods

Maintainer Stephen Nyangoma <s.o.nyangoma@bham.ac.uk>

License GPL (>=2)

Collate checkNo.replicates.R sampleClusteredData.R sampleSize3DscatterPlots.R sampleSizeContourPlots.R mostSimilarTwo.R negativeIntensitiesCorrection.R phenoDataFrame.R preProcRepeatedPeakData.R spectrumFilter.R f.R ztwo.R ZvaluescasesVcontrolsPlots.R ZvaluesfrommultinomPlots.R betweensampleVariance.Generic.R fisherInformation.Generic.R proteomicsExprsData.Generic.R proteomicspData.Generic.R replicateCorrelations.Generic.R sampleSizeGeneric.R sampleSizeParameters.Generic.R sample_technicalVaraiance.Generic.R aclinicalProteomicData.class.R aclinicalProteomicData.methods.R betweensampleVariance.Method.R fisherInformation.Method.R proteomicsExprsData.Method.R proteomicspData.Method.R replicateCorrelations.Method.R sampleSizeMethod.R sampleSizeParameters.Method.R sample_technicalVariance.Method.R

LazyLoad yes

ZipData yes

URL http://www.cancerstudies.bham.ac.uk/crctu/CLIPPDA.shtml

biocViews Proteomics, OneChannel, Preprocessing, DifferentialExpression, MultipleComparison

git_url https://git.bioconductor.org/packages/clippda

git_branch RELEASE_3_23

git_last_commit 17d6f3c

git_last_commit_date 2026-04-28

Repository Bioconductor 3.23

Date/Publication 2026-11-11

Contents

clippda-package . . . 3
aclinicalProteomicsData-class . . . 6
aclinicalProteomicsData-methods . . . 7
betweensampleVariance . . . 8
betweensampleVariance-methods . . . 10
checkNo.replicates . . . 11
f . . . 12
fisherInformation . . . 13
fisherInformation-methods . . . 15
liverdata . . . 16
liverRawData . . . 18
liver_pheno . . . 20
mostSimilarTwo . . . 21
negativeIntensitiesCorrection . . . 22
phenoDataFrame . . . 23
pheno_urine . . . 24
preProcRepeatedPeakData . . . 25
proteomicsExprsData . . . 27
proteomicsExprsData-methods . . . 28
proteomicspData . . . 29
proteomicspData-methods . . . 30
replicateCorrelations . . . 30
replicateCorrelations-methods . . . 31
sampleClusterdData . . . 32
sampleSize . . . 33
sampleSize-methods . . . 35
sampleSize3DscatterPlots . . . 35
sampleSizeContourPlots . . . 37
sampleSizeParameters . . . 39
sampleSizeParameters-methods . . . 40
sample_technicalVariance . . . 41
sample_technicalVariance-methods . . . 42
show-methods . . . 42

spectrumFilter 43
ztwo 44
ZvaluescasesVcontrolsPlots 45
ZvaluesfrommultinomPlots 46

Index **50**

`clippda-package` *A package for clinical proteomics profiling data analysis*

Description

This package is still under development but it is intended to provide a range of tools for analysing clinical genomics, methylation and proteomics, data with the non-standard repeated expression measurments arising from technical replicates. Most of these studies are observational case-control by design and the results of analyses must be appropriately adjusted for confounding factors and imbalances in the data. This regression-type problem is different from the regression problem in `limma`, in which all the `covariates` are some kind of `contrasts` and are therefore important. Our method is specifically suitable for analysing single-channel microarrays and proteomics data with repeated probe, or peak measurements, especially in the case where there is no one-to-one correspondence between cases and controls and the data cannot be analysed as log-ratios. In the current version (version 0.1.0), we are more concerned with the problem of sample size calculations for these data sets. But some tools for pre-processing of the repeated peaks data, including tools for checking for the consistency in the number of replicates across samples, the consistency of the peak information between replicate spectra and tools for data formatting and averaging, are included. `clippda` also implements a routine for evaluating differential-expression between cases and controls, especially for data in which each sample is assayed more than once, and are obtained from studies which are observational, or those for which the data are heterogeneous (e.g. data for cancer studies in which controls are not directly sampled, but are obtained from samples from suspected cases that turn out to be benign disease, after an operation, for example. In this case there could be serious imbalances in demographics between the cases and controls). The test statistics considered are derived from the methods developed by Nyangoma et al. (2009). These new methods for evaluating differential-expression are compared with the empirical Bayes method in the `limma` package. To limit the number of false positive discoveries, we control the tail probability of the proportion of false positives, (TPPFP). Further details can be found in the package `vignette`.

Details

Package:	anRpackage
Type:	Package
Version:	0.1.0
Date:	2009-03-25
License:	GPL (>=2)
LazyLoad:	yes

This package provides a method for calculating the sample size required when planning proteomic profiling studies using repeated peak measurements. At the planning stage, an experimenter typically does not yet have information on the heterogeneity of the data expected. We provide a method which makes it possible to input, and adjust for the effect of, the expected heterogeneity in the sample size calculations. The code for calculating sample size is the `sampleSize` function. It requires the computation of the between- and within-sample variations, the differences in mean between cases and controls, the intraclass correlations between duplicate peak data, and the heterogeneity correction factor. These can be computed from pilot data using the functions: `betweensampleVariance`, `withinsampleVariance`, `replicateCorrelations` and `FisherInformation`, respectively. Before the data can be analysed using these functions, it must be adequately preprocessed, and this package provides a number of tools for doing this. We provide a grid of the clinically important differences versus protein variances (with superimposed sample size contours). On this grid, we have plotted sample sizes computed using parameters from several real-life proteomic data from a range of cancer-types, fluid-types, cancer stages and experimental protocols of SELDI and MALDI. These values provide sample size ranges which may be used to estimate the number of samples required. The example below takes you through some of the processes of sample size calculations using this package.

Author(s)

Stephen Nyangoma

Maintainer: S Nyangoma <s.o.nyangoma@bham.ac.uk>

References

Birkner, et al. Issues of processing and multiple testing of SELDI-TOF MS Proteomic Data. Stat Appl Genet Mol Biol 2006, 5.1

Nyangoma, Stephen O.; Collins, Stuart I.; Altman, Douglas G.; Johnson, Philip; and Billingham, Lucinda J. (2012) Sample Size Calculations for Designing Clinical Proteomic Profiling Studies Using Mass Spectrometry, Statistical Applications in Genetics and Molecular Biology: Vol. 11: Iss. 3, Article 2.

Nyangoma SO, et al. Multiple Testing Issues in Discriminating Compound-Related Peaks and Chromatograms from High Frequency Noise, Spikes and Solvent-Based Noise in LC - MS Data Sets. Stat Appl Genet Mol Biol 2007, 6, 1, Article 23

Smyth GK, et al.: Use of within-array replicate spots for assessing differential expression in microarray experiments. Bioinformatics 2005, 21, 2067 - 75

Smyth GK: Linear models and emperical Bayes methods for assessing differential expression in microarray experiments. Stat Appl Genet Mol Biol 2004, 3, 1, Article 3

Examples

```
##############################################################################################
# The routine for calculating sample size required when planning a clinical proteomic
# profiling study is provided in the sampleSize function. First, this functions performs
# computations for sample size parameters, that include: the biological variance, the
# tecnical variance, the differences to be estimated, the intraclass correlation
# (if unknown). These computations are done ase follows:
##############################################################################################
```

```
##############################################################################################
# biological variation, difference to be estimated, and the p-values for differential-
# expression are computed using the generic function: betweensampleVariance
# It requires data of a aclinicalProteomicsData class, as input
##########################################################################

##################################################
# Creating data of a aclinicalProteomicsData class
##################################################

data(liverdata)

data(liver_pheno)

OBJECT=new("aclinicalProteomicsData")

OBJECT@rawSELDIdata=as.matrix(liverdata)

OBJECT@covariates=c("tumor" , "sex")

OBJECT@phenotypicData=as.matrix(liver_pheno)

OBJECT@variableClass=c('numeric','factor','factor')

OBJECT@no.peaks=53

Data=OBJECT

##############################################################################
# Data manipulation carried out internally by the betweensampleVariance function
##############################################################################

rawData <- proteomicsExprsData(Data)

no.peaks <- Data@no.peaks

JUNK_DATA <- sampleClusteredData(rawData,no.peaks)

JUNK_DATA=negativeIntensitiesCorrection(JUNK_DATA)

# we use the log base 2 expression values

LOG_DATA <- log2(JUNK_DATA)

##############################################################################################
# compute biological variation, difference to be estimated, and the p-values
##############################################################################################

BiovarDiffSig <- betweensampleVariance(OBJECT)

BiovarDiffSig
```

```
####################
# technical variance
####################

sample_technicalVariance(Data)

#############################################################################################
# heterogeneity correction-factor is the second diagonal element of the output
# matrix from the fisherInformation function, i.e. from the expected Fisher Information
#############################################################################################

Z <- as.vector(fisherInformation(Data)[2,2])/2
Z

#############################################################################################
# The outputs of these functions are converted into statistics used in
# the sample size calculations using a wraper function sampleSizeParameters
# it gives the consensus parameter values. You must specify the p-value and the
# intraclass correation, cutoff. The description of how these parameters
# are chosen is given in Nyangoma, et al. (2009).
#############################################################################################

intraclasscorr  <-  0.60 #cut-off for intraclass correlation

signifcut <- 0.05      #significance cut-off

sampleparameters=sampleSizeParameters(Data,intraclasscorr,signifcut)

#############################################################################################
# SAMPLE SIZE CALCULATIONS
#The function sampleSize calculates the protein variance, difference to be estimated,
# the technical varaince. These parameters are computed from statistics of peaks with
# medium biological variation.
# It also gives sample sizes for beta=c(0.90,0.80,0.70) and alpha = c(0.001, 0.01,0.05)
#############################################################################################

samplesize <- sampleSize(OBJECT,intraclasscorr,signifcut)
samplesize
```

`aclinicalProteomicsData-class` *Class "aclinicalProteomicsData"*

Description

This is a class object for the mass spectrometry data sets, which are in the same format as the raw data from the Biomarkers wizard software. It has slots of matrices of raw mass spectrometry and phenotypic data sets, a character variable for the classes of all the covariates in the phenotypic data matrix, a character variable for the covariates of interest, and numeric value for the number of peaks of interest.

Objects from the Class

Objects can be created by calls of the form `new("aclinicalProteomicsData", ...)`.

Slots

`rawSELDIdata`**:** Object of class `"matrix"`

`phenotypicData`**:** Object of class `"matrix"`

`variableClass`**:** Object of class `"character"`

`covariates`**:** Object of class `"character"`

`no.peaks`**:** Object of class `"numeric"`

Methods

show: Display an `aclinicalProteomicsData` instance.

Author(s)

S Nyangoma

Examples

```
showClass("aclinicalProteomicsData")

data(liverdata)
data(liver_pheno)

OBJECT = new("aclinicalProteomicsData",
             rawSELDIdata=as.matrix(liverdata),
             phenotypicData=as.matrix(liver_pheno),
             variableClass=c('numeric','factor','factor'),
             no.peaks=53)

show(OBJECT)
```

---

aclinicalProteomicsData-methods

S4 method for the aclinicalProteomicsData class

Description

An S4 method for the object `aclinicalProteomicsData` class objects.

Examples

```
setClass("clinicalProteomicsData",representation(rawSELDIdata="matrix",phenotypicData="matrix",varInfo="charac
variableClass="character",no.peaks="numeric"),
prototype(rawSELDIdata=matrix(0),phenotypicData=matrix(0),varInfo=as.character(0),variableClass=as.character(0

slotNames("aclinicalProteomicsData")

setMethod("show","aclinicalProteomicsData",
function(object) {
cat("clinical proteomics data")
cat("Type   :", class(object), "\n")
cat("raw data  :", paste(object@rawSELDIdata), "\n")
cat("phenotypic data :", paste(object@phenotypicData), "\n")
cat("variable information :", paste(object@varInfo), "\n")
cat("variable class  :", paste(object@variableClass), "\n")
cat("number of peaks :", paste(object@no.peaks), "\n")
}
)

slotNames( new("clinicalProteomicsData"))

## library(clippda)
data(liverdata)
data(liver_pheno)

OBJECT=new("clinicalProteomicsData")
OBJECT@rawSELDIdata=as.matrix(liverdata)
OBJECT@varInfo=c("tumor" ,    "sex")
OBJECT@phenotypicData=as.matrix(liver_pheno)
OBJECT@variableClass=c('numeric','factor','factor')
OBJECT@no.peaks=53

show(OBJECT)
```

`betweensampleVariance` *A generic function for computing the biological variance and mean differences between cases and controls*

Description

This generic function fits a regression model to the averaged replicate data. The outputs are the between sample variance, and the differences in mean expression between cases and controls, adjusted for confounders.

Usage

```
betweensampleVariance(Data, ...)
```

Arguments

Data	An object of aclinicalProteomicsData class.
...	Some methods for this generic function may take additional, optional arguments. At present none do.

Value

It returns a list with the following components:

betweensamplevariance	A vector of the between-sample variance for each peak.
differences	A vector of the differences in mean expression values between the cases and controls, adjusted for confounders for each peak.
significance	A dataframe, or a vector of the differential-expression p-values for each peak.

Author(s)

Stephen Nyangoma

Examples

```
########################################
##### methods for the generic function
########################################

showMethods("betweensampleVariance")

##################################################
# Creating data of a aclinicalProteomicsData class
##################################################

data(liverdata)

data(liver_pheno)

OBJECT=new("aclinicalProteomicsData")

OBJECT@rawSELDIdata=as.matrix(liverdata)

OBJECT@covariates=c("tumor" , "sex")

OBJECT@phenotypicData=as.matrix(liver_pheno)

OBJECT@variableClass=c('numeric','factor','factor')

OBJECT@no.peaks=53

Data=OBJECT

#############################################################################
# Data manipulation carried out internally by the betweensampleVariance function
```

```
##############################################################################

rawData <- proteomicsExprsData(Data)

no.peaks <- Data@no.peaks

JUNK_DATA <- sampleClusteredData(rawData,no.peaks)

JUNK_DATA=negativeIntensitiesCorrection(JUNK_DATA)

# we use the log-basetwo2 expression values

LOG_DATA <- log2(JUNK_DATA)

##############################################################################################
# compute biological variation, difference to be estimated, and the p-values
##############################################################################################

BiovarDiffSig <- betweensampleVariance(OBJECT)

BiovarDiffSig
```

---

betweensampleVariance-methods
Methods for Function betweensampleVariance

Description

Methods for function betweensampleVariance are defined with class "aclinicalProteomicsData" in the signature.

Methods

Data = "aclinicalProteomicsData" This is a class object for the mass spectrometry data sets, which are in the same format as the raw data from the Biomarkers wizard software. It has slots of matrices of raw mass spectrometry and phenotypic data sets, a character variable for the classes of all the covariates in the phenotypic data matrix, a character variable for the covariates of interest, and numeric value for the number of peaks of interest. See also aclinicalProteomicsData-class.

checkNo.replicates *A function to detect disparity in the number of replicates across assays*

Description

Sometimes in a mass spectrometry experiment, it happens that a few samples have been mislabelled. Mislabelling means that some replicates are in the wrong sample group, and this results in some samples having more (or less) replicates than the number intended by the experimentalist. Apart from disparity in the number of replicates due to mislabelling, a few samples, e.g. the quality control (QC) samples, are often assayed several times. The aim is to analyze data with the same number of technical replicates (in this case, duplicates) for every sample. The function `checkNo.replicates` identifies samples with a disparate number of replicates. The identified samples are treated as follows:

(i) The QC samples can be independently analysed to ascertain the reproducibility of the data.

(ii) The samples with no replicates are discarded from further analysis.

(iii) The samples with more replicates than expected, due to mislabelling (or otherwise), are pre-processed using the function: `mostSimilarTwo` which detects and discards replicates which give conflicting peak information compared to the rest of the replicates. Here, the two most similar replicates are treated as the correct replicates for the sample in question.

Usage

```
checkNo.replicates(rawData, no.peaks, no.replicates)
```

Arguments

`rawData`	Duplicate data in the same format as the raw data from the Biomarker wizard software.
`no.peaks`	The number of peaks detected by the Biomarker wizard
`no.replicates`	The number of replicates intended by the biologist.

Value

It returns a vector whose elements are labels for samples with a disparate number peaks.

Author(s)

Stephen Nyangoma

Examples

```
data(liverRawData)

rawData <- liverRawData

no.peaks <- 53
```

```
no.replicates <- 2

checkNo.replicates(rawData,no.peaks,no.replicates)
```

---

f *A function to compute adjustments for the effct of covariates (Z values) for an experiment with a binary exposure and a binary confounder*

Description

A function to compute the Z values when planning an experiment with a binary exposure and a binary confounder. You input the probabilities of 3-cells of the resulting multinomial distribution.

Usage

```
f(x, y, z)
```

Arguments

x	Proportion of elements in cell 1 of a multinomial population with four cells.
y	Proportion of elements in cell 2 of a multinomial population with four cells.
z	Proportion of elements in cell 1 of a multinomial population with four cells. The z here is different from the Z which contains information on the effect of covariates and data imbalance on sample size.

Value

It returns a single real number (greater than or equal 2), representing Z.

Author(s)

Stephen Nyangoma

References

Nyangoma SO, Ferreira JA, Collins SI, Altman DG, Johnson PJ, and Billingham LJ: Sample size calculations for planning clinical proteomic profiling studies using mass spectrometry. (Working paper)

See Also

The function ZvaluesformultinomialPlots

Examples

```
# for a 1:1:1:1 experiment
x=.25;y=.25;z=.25

# compute Z
Z=f(x,y,z)
Z
## The function is currently defined as
function (x,y,z) {
Z=(1-x-z)*(x+y)/(2*(((1-x-z)*(1-x-y)*(1-y-z))-(1-x-y-z)^2))
Z
  }
```

`fisherInformation` *A generic function to compute the heterogeneity correction factor in sample size calculations*

Description

This generic function computes the inverse of the expected 'Fisher' information matrix, `I^-1/theta` (see the definition of theta in section 2.3 of Nyangoma `et al`., 2009). The second diagonal element of this matrix is the variance of the mean difference between cases and controls, having adjusted for the effect of confounders. The elements of `I` are sums of the proportions of samples having given attributes, or sums of proportions of class memberships of given conditional contingency tables obtained from the cross tabulated the attributes of the samples under study. Thus it contains information on the heterogeneity in the data due to imbalances in the proportions of samples having given attributes.

Usage

```
fisherInformation(Data, ...)
```

Arguments

`Data`	An object of `aclinicalProteomicsData` class. It extracts and uses a dataframe of the clinical information about the samples from a `slot` in the `Data`.
`...`	Some methods for this generic function may take additional, optional arguments. At present none do.

Details

Note that continuous variables must first be discretized, and the variable names must coincide with the column names of `PhenoInfo` extracted from the `object`. Currently this function only accepts a maximum of three binary variables. The existing methods (e.g. Diggle et al. 1997, page 31) for continuous repeated data consider only a single exposure variable. We recommend that some form of variable selection be used to determine which covariates to include in the analysis.

Value

This function returns a matrix: and its second diagonal element (divided by 2) is the quantity called Z (or the heterogeneity-correction factor) in the sample size calculation function, sampleSize.

Author(s)

Stephen Nyangoma

References

1. Nyangoma SO, Ferreira JA, Collins SI, Altman DG, Johnson PJ, and Billingham LJ (2009): Sample size calculations for planning clinical proteomic profiling studies using mass spectrometry. Bioinformatics (Submitted)

2. Diggle PJ, Heagerty P, Liang K.-Y and Zeger SL. (2002). Analysis of Longitudinal Data (second edition). Oxford: Oxford University Press

Examples

```
##############################################################################################
#The matrices of interest are of the form (see eq. 15, 18 and 22 Nyangoma et al. (2009))
##############################################################################################

#Examples are:

###################
# 1 binary variable
###################

data.frame(x1=c(1,'b'),x2=c('b','b'))

#####################
# 2 binary variables
#####################
data.frame(x1=c(1,'b','c'),x2=c('b','b','d'),x3=c('c','d','c'))

########################################
# 3 binary variables

data.frame(x1=c(1,'b','c','d'),x2=c('b','b','e','f'),x3=c('c','e','c','g'),x4=c('d','f','g','d'))

#############################################################################
#############################################################################
# Data # pheno_urine
# the phenotypic information of the urine cancer patients and normal controls.
#####
# I have discretized protein concentration
# concentration<=70 and concentration>70
########################################
########################################

#data(pheno_urine)
```

```
#PhenoInfo <- pheno_urine

#variables <- c('Tumor','Sex','Protein_concIndex')

#variables=c('Tumor','Sex')
#variables=c('Tumor')


# Tumor must contain characters "c" and "n"

#Protein_concIndex <- pheno_urine[!(pheno_urine$stage == 'late'),]$Protein_conc

#Protein_concIndex[Protein_concIndex<=70] <- 0
#Protein_concIndex[Protein_concIndex>70] <- 1
#Protein_concIndex=as.factor(Protein_concIndex)

#PhenoInfo <- data.frame(pheno_urine[!(pheno_urine$stage == 'late'),],Protein_concIndex)

#FisherInformation(PhenoInfo,variables)

data(liverdata)

data(liver_pheno)

OBJECT=new("aclinicalProteomicsData")

OBJECT@rawSELDIdata=as.matrix(liverdata)
OBJECT@covariates=c("tumor" ,    "sex")
OBJECT@phenotypicData=as.matrix(liver_pheno)
OBJECT@variableClass=c('numeric','factor','factor')
OBJECT@no.peaks=53

inversefisherinformation <- fisherInformation(OBJECT)

inversefisherinformation
```

`fisherInformation-methods`

Methods for Function fisherInformation

Description

Methods for function `fisherInformation` are defined with class "aclinicalProteomicsData" in the signature.

Methods

Data = "aclinicalProteomicsData" This is a class object for the mass spectrometry data sets, which are in the same format as the raw data from the Biomarkers wizard software. It has

slots of matrices of raw mass spectrometry and phenotypic data sets, a character variable for the classes of all the covariates in the phenotypic data matrix, a character variable for the covariates of interest, and numeric value for the number of peaks of interest. See also `aclinicalProteomicsData-class`.

`liverdata` *A dataframe of the protein expression data, peak information, and sample information*

Description

A dataframe of the duplicate protein expression data, peak information, sample information (e.g. sample ID, stage, gender, etc.). This is a pre-processed version of "raw `.csv`" file from the Biomarker wizard. The pre-processing involves filtering out samples with conflicting peak information, and detecting and discarding samples with no replicates.

Usage

```
data(liverdata)
```

Format

A data frame with 13886 observations on the following 6 variables.

`SampleTag` a numeric vector of sample ID.

`CancerType` a factor, with levels `c` and `n`, indicating cancer class

`Spectrum` a numeric vector, indicating the experimental run.

`Peak` a numeric vector identifying the peak.

`Intensity` a numeric vector of expression values.

`Substance.Mass` a numeric vector contaning the m/z (mass-to-charge ratio) value.

Source

Ward DG, Cheng Y, N'Kontchou G, Thar TT, Barget N, Wei W, Billingham LJ, Martin A, Beaugrand M, Johnson PJ: Changes in the serum proteome associated with the development of hepatocellular carcinoma in hepatitis C-related cirrhosis. Br J Cancer. 2006, 94(2):287-92.

References

Ward DG, Cheng Y, N'Kontchou G, Thar TT, Barget N, Wei W, Billingham LJ, Martin A, Beaugrand M, Johnson PJ: Changes in the serum proteome associated with the development of hepatocellular carcinoma in hepatitis C-related cirrhosis. Br J Cancer. 2006, 94(2):287-92.

Examples

```
#####################################################
#####################################################
## a pre-proceesed version of the raw .csv file from the
## Biomarker wizard.
#####################################################
#####################################################

data(liverdata)
data(liverRawData)
#############################################################################################
#############################################################################################
# liverdata is obtained by pre-processing of the raw .csv file from the Biomarker wizard
# as follows. These samples pre-processed to:
#  (i) discard the information on samples which have no replicate data, and
# (ii) for samples with more than 2 replicate expression data, only duplicates with most
#      similar peak information are retained for use in subsequent analyses.
# A wrapper function for executing these two pre-processing steps is preProcRepeatedPeakData
#############################################################################################
#############################################################################################

threshold <- 0.80
no.replicates <- 2
no.peaks <- 53
Data <- preProcRepeatedPeakData(liverRawData, no.peaks, no.replicates, threshold)

#############################################################################################
#############################################################################################
# Only sample with ID 250 has no replicates and has been omitted from the data to be used
# in subsequent analyses. This fact may varified by  using:
#############################################################################################
#############################################################################################

setdiff(unique(liverRawData$SampleTag),unique(liverdata$SampleTag))
setdiff(unique(Data$SampleTag),unique(liverdata$SampleTag))

#########################################################################
# Now filter out the samples with conflicting replicate peak information
# using the spectrumFilter function:
#########################################################################

TAGS <- spectrumFilter(Data,threshold,no.peaks)$SampleTag

NewRawData2 <- spectrumFilter(Data,threshold,no.peaks)
dim(Data)

dim(liverdata)

dim(NewRawData2)

#############################################################################################
#############################################################################################
```

```
# In the case of this data (the liver data), all technical replicates have coherent peak
# information, since no sample information has been discarded by spectra filter.
##############################################################################################
##############################################################################################

##############################################################################################
##############################################################################################
# Let us have a look at what the pre-processing does to samples with more than 2 replicate
# spectra. Both samples with IDs 25 and 40 have more than 2 replicates.
##############################################################################################
##############################################################################################

length(liverRawData[liverRawData$SampleTag == 25,]$Intensity)/no.peaks
length(liverRawData[liverRawData$SampleTag == 40,]$Intensity)/no.peaks

##############################################################################################
##############################################################################################
# Take correlations of the log-intensities to find which of the 2 replicates have the
# most coherent peak information.
##############################################################################################
##############################################################################################

Mat1 <- matrix(liverRawData[liverRawData$SampleTag == 25,]$Intensity,53,3)
Mat2 <-matrix(liverRawData[liverRawData$SampleTag == 40,]$Intensity,53,4)
cor(log2(Mat1))
cor(log2(Mat2))

#use mostSimilarTwo function to get duplicate spectra with most coherent peak information

Mat1 <- matrix(liverRawData[liverRawData$SampleTag == 25,]$Intensity,53,3)
Mat2 <-matrix(liverRawData[liverRawData$SampleTag == 40,]$Intensity,53,4)
sort(mostSimilarTwo(cor(log2(Mat1))))
sort(mostSimilarTwo(cor(log2(Mat2))))

##############################################################################################
##############################################################################################
#Next, check that the pre-processed data, \Robject{NewRawData2}, contains similar
# information to liverdata (the allready pre-processed data, included in the clippda).
##############################################################################################
##############################################################################################
names(NewRawData2)
dim(NewRawData2)
names(liverdata)
dim(liverdata)
setdiff(NewRawData2$SampleTag,liverdata$SampleTag)
setdiff(liverdata$SampleTag,NewRawData2$SampleTag)
summary(NewRawData2$Intensity)
summary(liverdata$Intensity)
```

`liverRawData` *A dataframe of the protein expression data, peak information and sample information*

Description

A dataframe of the protein expression data, peak information, sample information (e.g. sample ID, stage, gender, etc.). This is the "raw `.csv`" files from the Biomarker wizard.

Usage

```
data(liverRawData)
```

Format

A data frame with 14098 observations on the following 6 variables.

`SampleTag` a numeric vector of sample ID.

`CancerType` a factor, with levels `c` and `n`, indicating cancer class

`Spectrum` a numeric vector indicating the experimental run.

`Peak` a numeric vector identifying a peak.

`Intensity` a numeric vector of expression values.

`Substance.Mass` a numeric vector indicating the m/z value.

Source

Ward DG, Cheng Y, N'Kontchou G, Thar TT, Barget N, Wei W, Billingham LJ, Martin A, Beaugrand M, Johnson PJ: Changes in the serum proteome associated with the development of hepatocellular carcinoma in hepatitis C-related cirrhosis. Br J Cancer. 2006, 94(2):287-92.

References

Ward DG, Cheng Y, N'Kontchou G, Thar TT, Barget N, Wei W, Billingham LJ, Martin A, Beaugrand M, Johnson PJ: Changes in the serum proteome associated with the development of hepatocellular carcinoma in hepatitis C-related cirrhosis. Br J Cancer. 2006, 94(2):287-92.

Examples

```
data(liverRawData)
```

`liver_pheno` *A dataframe of phenotypic information*

Description

A dataframe containing the sample phenotypic information.

Usage

```
data(liver_pheno)
```

Format

A data frame with 131 observations on the following 3 variables.

`SampleTag` a character/numeric vector of sample ID.

`tumor` a factor, with levels `c` and `n`, describing the class of the samples.

`sex` a factor, with levels `F` and `M`, describing the gender of the persons from which the sample has been taken.

Source

Ward DG, Cheng Y, N'Kontchou G, Thar TT, Barget N, Wei W, Billingham LJ, Martin A, Beaugrand M, Johnson PJ: Changes in the serum proteome associated with the development of hepatocellular carcinoma in hepatitis C-related cirrhosis. Br J Cancer. 2006, 94(2):287-92.

References

Ward DG, Cheng Y, N'Kontchou G, Thar TT, Barget N, Wei W, Billingham LJ, Martin A, Beaugrand M, Johnson PJ: Changes in the serum proteome associated with the development of hepatocellular carcinoma in hepatitis C-related cirrhosis. Br J Cancer. 2006, 94(2):287-92.

Examples

```
data(liver_pheno)
```

`mostSimilarTwo` *A function which indentifies two columns of a matrix, or dataframe, with the highest pairwise positive correlations*

Description

A common practice in the analysis of repeated mass spectrometry data is to average the replicate expression values, a method which is only valid if there is some coherence in the peak information across replicates. The function `mostSimilarTwo` identifies the two columns of a matrix (or a dataframe) with the highest pairwise positive correlations. The most highly correlated replicates contain the most similar compounds. This function may also be used to reduce the number of spectra being analysed to two.

Usage

```
mostSimilarTwo(Mat)
```

Arguments

`Mat`	A dataframe, with the columns being the variables of interest, for example the spectra.

Details

The main application of this function is in the pre-processing of mass spectrometry data. In a mass spectrometry experiment, it often happens that there is mislabelling of samples, which results in some replicates being assigned to the wrong sample class. This function sifts through this data to identify the two spectra with the most coherent signal information between them. Thus, its function has the potential to help in reducing the number of false-positive discoveries. Its other application is in the reduction of the number of replicates to two, which are then analysed using tools for duplicate peak (or gene) expression data.

Value

It returns a vector with two elements, being the column indices for the two most correlated variables.

Author(s)

Stephen Nyangoma

References

Ward DG, Nyangoma S, Joy H, Hamilton E, Wei W, Tselepis C, Steven N, Wakelam MJ, Johnson PJ, Ismail T, Martin A: Proteomic profiling of urine for the detection of colon cancer. Proteome Sci. 2008, 16(6):19

Examples

```
n <- 10

Mat <- data.frame(x1=rnorm(n, mean = 0, sd = 1),x2=rnorm(n, mean = 0, sd = 3),x3=rnorm(n, mean = 1, sd = 1),x4=
rnorm(n,mean=2,sd=2))

mostSimilarTwo(Mat)
```

`negativeIntensitiesCorrection` *A function to correct the data for the negative intensities caused by the normalization and background correction procedures of mass spectrometry data*

Description

This function corrects the mass spectra data, which has been pre-processed using tools tools from the Biomarkers Wizard `PROcess` softwares, for the negative intensities caused by their normalization and background correction procedures.

Usage

```
negativeIntensitiesCorrection(Data)
```

Arguments

`Data`	is a dataframe , or a matrix, or a vector, of numerical values.

Value

A dataframe , or a matrix, or a vector (whichever is the input quantity), of nonnegative numerical values.

Author(s)

Stephen Nyangoma

Examples

```
data(liverdata)

no.peaks <- 53

JUNK_DATA <- sampleClusteredData(liverdata,no.peaks)

Data=JUNK_DATA
```

```
Data=JUNK_DATA
Data=Data+1
temp=negativeIntensitiesCorrection(Data)
temp[,1]
Data[,1]
```

---

phenoDataFrame *A generic function to set classes for the variables in the dataframe of phenotypic information.*

Description

A function to set classes (e.g. as numeric or factor) to the variables in the dataframe.

Usage

```
phenoDataFrame(PhenoData, variableClass)
```

Arguments

PhenoData	is the dataframe of phenotypic information extracted from an object of class aclinicalProteomicsData.
variableClass	a character vector of length equal to the number of columns of the dataframe of phenotypic information, giving classes of the variables studied.

Value

A dataframe of class-corrected phenotypic variables.

Author(s)

Stephen Nyangoma

Examples

```
data(liverdata)

data(liver_pheno)

OBJECT=new("aclinicalProteomicsData")

OBJECT@rawSELDIdata=as.matrix(liverdata)
OBJECT@covariates=c("tumor" ,    "sex")
OBJECT@phenotypicData=as.matrix(liver_pheno)
OBJECT@variableClass=c('numeric','factor','factor')
OBJECT@no.peaks=53

Data=OBJECT
variableClass =Data@variableClass
```

```
variables = c("SampleTag","tumor","sex")

PhenoInfo <- data.frame(Data@phenotypicData)

PhenoData <- data.frame(Data@phenotypicData)

pData=phenoDataFrame(PhenoData, variableClass)

class(pData$sex)
class(pData$SampleTag)
class(pData$tumor)
```

---

pheno_urine *A dataframe of phenotypic information*

Description

A dataframe containing the sample phenotypic information.

Usage

```
data(pheno_urine)
```

Format

A data frame with 167 observations on the following 6 variables.

`SpectrumTag` a character/numeric vector of sample ID.

`Tumor` a factor, with levels `c` and `n`, describing the sample class

`Age` a numeric vector of age.

`Sex` a factor, with levels `female` and `male`, indicating gender.

`stage` a factor, with levels `early`, `late` and `normal`, indicating tumor stage.

`Protein_conc` a numeric vector of urine concentration.

Source

Ward DG, Nyangoma S, Joy H, Hamilton E, Wei W, Tselepis C, Steven N, Wakelam MJ, Johnson PJ, Ismail T, Martin A: Proteomic profiling of urine for the detection of colon cancer. Proteome Sci. 2008, 16(6):19.

References

Ward DG, Nyangoma S, Joy H, Hamilton E, Wei W, Tselepis C, Steven N, Wakelam MJ, Johnson PJ, Ismail T, Martin A: Proteomic profiling of urine for the detection of colon cancer. Proteome Sci. 2008, 16(6):19.

Examples

```
data(pheno_urine)
```

`preProcRepeatedPeakData` *A function to pre-process repeated raw peak data*

Description

This function pre-processes repeated peak data from the Biomarker wizard. It identifies, and removes, samples which have no replicates. Then, for the samples with two or more replicates, it selects and returns data for the two replicates with most similar expression pattern. Then, the samples with conflicting peak information may be removed using the function: `spectrumFilter`. The output is similar to the `liverdata`, a pre-processed data which is included in this package.

Usage

```
preProcRepeatedPeakData(rawData, no.peaks, no.replicates,threshold)
```

Arguments

`rawData`	the raw data from the Biomarker wizard.
`no.peaks`	the number of peaks detected by the Biomarker wizard.
`no.replicates`	the intended number of replicates. The departures from this number could be due to mislabelling, quality control (QC) assays, or a few other samples which could have been assayed more times than the majority of samples.
`threshold`	The threshold for declaring expression patterns between duplicates as being "similar". It is especially needed in the function `spectrumFilter`, which is used by this function.

Value

It returns a dataframe of duplicate expression data for all peaks in the same format as the ".csv" data from the Biomarker wizard.

Author(s)

Stephen Nyangoma

References

Ward DG, Nyangoma S, Joy H, Hamilton E, Wei W, Tselepis C, Steven N, Wakelam MJ, Johnson PJ, Ismail T, Martin A: Proteomic profiling of urine for the detection of colon cancer. Proteome Sci. 2008, 16(6):19

Examples

```
#####################################################
#####################################################
# a pre-proceesed version of the raw .csv file from the
# Biomarker wizard.
#####################################################
#####################################################

data(liverdata) # allready pre-processed data
data(liverRawData) # raw version of liverdata
#############################################################################################
#############################################################################################
# These samples pre-processed to:
#  (i) discard the information on samples which have no replicate data, and
# (ii) for samples with more than 2 replicate expression data, only duplicates with most
#      similar peak information are retained for use in subsequent analyses.
# A wrapper function for executing these two pre-processing steps is preProcRepeatedPeakData
#############################################################################################
#############################################################################################

threshold <- 0.80
no.replicates <- 2
no.peaks <- 53
Data <- preProcRepeatedPeakData(liverRawData, no.peaks, no.replicates, threshold)

#############################################################################################
#############################################################################################
# Only sample with ID 250 has no replicates and has been omitted from the data to be used
# in subsequent analyses. This fact may varified by  using:
#############################################################################################
#############################################################################################

setdiff(unique(liverRawData$SampleTag),unique(liverdata$SampleTag))
setdiff(unique(Data$SampleTag),unique(liverdata$SampleTag))

#########################################################################
# Now filter out the samples with conflicting replicate peak information
# using the spectrumFilter function:
#########################################################################

#TAGS <- spectrumFilter(Data,threshold,no.peaks)$SampleTag

NewRawData2 <- spectrumFilter(Data,threshold,no.peaks)

dim(Data)
dim(liverdata)
dim(NewRawData2)

#############################################################################################
#############################################################################################
# In the case of this data (the liver data), all technical replicates have coherent peak
# information, since no sample information has been discarded by spectra filter.
```

```
##########################################################################################
##########################################################################################

##########################################################################################
##########################################################################################
# Let us have a look at what the pre-processing does to samples with more than 2 replicate
# spectra. Both samples with IDs 25 and 40 have more than 2 replicates.
##########################################################################################
##########################################################################################

length(liverRawData[liverRawData$SampleTag == 25,]$Intensity)/no.peaks
length(liverRawData[liverRawData$SampleTag == 40,]$Intensity)/no.peaks

##########################################################################################
##########################################################################################
# Take correlations of the log-intensities to find which of the 2 replicates have the
# most coherent peak information.
##########################################################################################
##########################################################################################

Mat1 <- matrix(liverRawData[liverRawData$SampleTag == 25,]$Intensity,53,3)
Mat2 <-matrix(liverRawData[liverRawData$SampleTag == 40,]$Intensity,53,4)
cor(log2(Mat1))
cor(log2(Mat2))

#use mostSimilarTwo function to get duplicate spectra with most coherent peak information

Mat1 <- matrix(liverRawData[liverRawData$SampleTag == 25,]$Intensity,53,3)
Mat2 <-matrix(liverRawData[liverRawData$SampleTag == 40,]$Intensity,53,4)
sort(mostSimilarTwo(cor(log2(Mat1))))
sort(mostSimilarTwo(cor(log2(Mat2))))

##########################################################################################
##########################################################################################
#Next, check that the pre-processed data, \Robject{NewRawData2}, contains similar
# information to liverdata (the allready pre-processed data, included in the clippda).
##########################################################################################
##########################################################################################
names(NewRawData2)
dim(NewRawData2)
names(liverdata)
dim(liverdata)
setdiff(NewRawData2$SampleTag,liverdata$SampleTag)
setdiff(liverdata$SampleTag,NewRawData2$SampleTag)
summary(NewRawData2$Intensity)
summary(liverdata$Intensity)
```

`proteomicsExprsData` *A generic fuction to extract duplicate SELDI data from an object of aclinicalProteomicsData class in the same format as the data from Biomarkers wizard*

Description

This generic function extracts a matrix of duplicate SELDI data from an object of aclinicalProteomicsData class. It then converts it into a dataframe which is in the same format as the data from Biomarkers wizard. In this dataframe, the intensities and the subject mass-to-charge ratio are represented as numeric variables, while the sample-tag is represented as a character variable.

Usage

```
proteomicsExprsData(Data, ...)
```

Arguments

Data	is an object of a aclinicalProteomicsData class.
...	means other defined arguments. Currently, we have not defined additional arguments.

Value

A dataframe of expression values, the substance mass, patient labels, and any other defined sample information.

Author(s)

S Nyangoma

Examples

```
data(liverdata)

data(liver_pheno)

OBJECT=new("aclinicalProteomicsData")

OBJECT@rawSELDIdata=as.matrix(liverdata)
OBJECT@covariates=c("tumor" ,    "sex")
OBJECT@phenotypicData=as.matrix(liver_pheno)
OBJECT@variableClass=c('numeric','factor','factor')
OBJECT@no.peaks=53

head(proteomicsExprsData(OBJECT))
```

---

proteomicsExprsData-methods

Methods for Function proteomicsExprsData

Description

Methods for function proteomicsExprsData are defined with class "aclinicalProteomicsData" in the signature.

Methods

Data = "aclinicalProteomicsData" This is a class object for the mass spectrometry data sets, which are in the same format as the raw data from the Biomarkers wizard software. It has slots of matrices of raw mass spectrometry and phenotypic data sets, a character variable for the classes of all the covariates in the phenotypic data matrix, a character variable for the covariates of interest, and numeric value for the number of peaks of interest. See also `aclinicalProteomicsData-class`.

---

proteomicspData *A function to extract a dataframe of phenotypic information from an object of aclinicalProteomicsData class*

Description

This function takes an object of `aclinicalProteomicsData` class, extracts a matrix of phenotypic data, and converts it to a dataframe with with variables having defined classes.

Usage

```
proteomicspData(Data, ...)
```

Arguments

`Data`	is an object of `aclinicalProteomicsData` class.
`...`	means other defined arguments. Currently, we have not defined additional arguments.

Value

Returns a dataframe with variables having defined classes.

Author(s)

S Nyangoma

Examples

```
########################################
##### the data
########################################


data(liverdata)

data(liver_pheno)

OBJECT=new("aclinicalProteomicsData")
```

```
OBJECT@rawSELDIdata=as.matrix(liverdata)
OBJECT@covariates=c("tumor" ,    "sex")
OBJECT@phenotypicData=as.matrix(liver_pheno)
OBJECT@variableClass=c('numeric','factor','factor')
OBJECT@no.peaks=53

head(proteomicspData(OBJECT))
```

`proteomicspData-methods`

Methods for Function proteomicspData

Description

Methods for function `proteomicspData` are defined with class "aclinicalProteomicsData" in the signature.

Methods

Data = "aclinicalProteomicsData" This is a class object for the mass spectrometry data sets, which are in the same format as the raw data from the Biomarkers wizard software. It has slots of matrices of raw mass spectrometry and phenotypic data sets, a character variable for the classes of all the covariates in the phenotypic data matrix, a character variable for the covariates of interest, and numeric value for the number of peaks of interest. See also `aclinicalProteomicsData-class`.

`replicateCorrelations` *A generic function to compute intraclass correlations*

Description

This generic function computes intraclass correlations for duplicate peak data.

Usage

```
replicateCorrelations(Data, ...)
```

Arguments

`Data`	An object of `aclinicalProteomicsData` class.
`...`	Some methods for this generic function may take additional, optional arguments. At present none do.

Value

`consensus:` consensus intraclass correlation.

`correlations:` intraclass correlations for each peak.

Author(s)

Stephen Nyangoma

References

Nyangoma SO, Ferreira JA, Collins SI, Altman DG, Johnson PJ, and Billingham LJ: Sample size calculations for planning clinical proteomic profiling studies using mass spectrometry. Bioinformatics (Submitted)

Smyth GK, et al.: Use of within-array replicate spots for assessing differential expression in microarray experiments. Bioinformatics 2005, 21, 2067 - 75

Smyth GK: Linear models and emperical Bayes methods for assessing differential expression in microarray experiments. Stat Appl Genet Mol Biol 2004, 3, 1, Article 3

Examples

```
data(liverdata)

data(liver_pheno)

OBJECT=new("aclinicalProteomicsData")

OBJECT@rawSELDIdata=as.matrix(liverdata)
OBJECT@covariates=c("tumor" ,    "sex")
OBJECT@phenotypicData=as.matrix(liver_pheno)
OBJECT@variableClass=c('numeric','factor','factor')
OBJECT@no.peaks=53

replicateCorrelations(OBJECT)
```

---

replicateCorrelations-methods

Methods for Function replicateCorrelations

Description

Methods for function `replicateCorrelations` are defined with class "aclinicalProteomicsData" in the signature.

Methods

Data = "aclinicalProteomicsData" This is a class object for the mass spectrometry data sets, which are in the same format as the raw data from the Biomarkers wizard software. It has slots of matrices of raw mass spectrometry and phenotypic data sets, a character variable for the classes of all the covariates in the phenotypic data matrix, a character variable for the covariates of interest, and numeric value for the number of peaks of interest. See also `aclinicalProteomicsData-class`.

`sampleClusterdData` *A function to arrange the data in sample-wise pairs*

Description

This function arranges the duplicate data, grouped by samples, in a form which can be averaged using the `limma` function `avedups`.

Usage

```
sampleClusteredData(Data, no.peaks)
```

Arguments

`Data`	A data frame of duplicate data from the biomarker wizard.
`no.peaks`	The number of peaks detected by the biomarker wizard.

Details

The output is a dataframe of repeated sample expression values, clustered by samples. This data is used as an input to the `betweensampleVariance` function, and is then averaged and used to compute the biological variance as well as the mean difference in the expression values between cancer and noncancer controls.

Value

The output is a dataframe of repeated sample expression values, clustered by samples.

Author(s)

Stephen Nyangoma

Examples

```
# arrange the data in a form which can be averaged by the limma function dupcor
# use the function called limmaData

data(liverdata)

no.peaks <- 53

Data <- liverdata

JUNK_DATA <- sampleClusteredData(Data,no.peaks)
```

`sampleSize` *A function for sample size calculations*

Description

This generic function `sampleSize` calculates the protein variance and the sample size required to estimate the clinically important differences (`DIFF`). The input data are the consensus parameters of peaks with medium biological variation.

Usage

```
sampleSize(Data,intraclasscorr,signifcut, ...)
```

Arguments

`Data`	An object of `aclinicalProteomicsData` class.
`intraclasscorr`	An object of `numeric` class. It is a known value of the intraclass correlation, or an estimate from a pilot data.
`signifcut`	An object of `numeric` class. It is significance threshold (usually, taken to be 0.05 in the analysis of the protein profiling studies).
`...`	Some methods for this generic function may take additional, optional arguments. At present none do.

Details

The sample sizes are computed for various combinations of the power with values `beta=c(0.90,0.80,0.70)` and the significance values, `alpha = c(0.001, 0.01,0.05)`. Note that here we use `beta` for power rather than the conventional `1-beta`.

Value

`protein_variance`	consensus protein variance
`replicate_correlation`	consensus intraclass correlation
`sample_size`	the sample size required

Author(s)

Stephen Nyangoma

References

Nyangoma SO, Ferreira JA, Collins SI, Altman DG, Johnson PJ, and Billingham LJ: Sample size calculations for planning clinical proteomic profiling studies using mass spectrometry. Bioinformatics (Submitted)

Smyth GK, et al.: Use of within-array replicate spots for assessing differential expression in microarray experiments. Bioinformatics 2005, 21, 2067 - 75

Smyth GK: Linear models and emperical Bayes methods for assessing differential expression in microarray experiments. Stat Appl Genet Mol Biol 2004, 3, 1, Article 3

Examples

```
#########################################################################
## SAMPLE SIZE
#########################################################################
#The function sampleSize calculates the biological variance, differences.
#These are the consensus values of peaks with median biological variation
# It also gives sample sizes for beta=c(0.90,0.80,0.70) and alpha = c(0.001, 0.01,0.05)

#########################################################################
#########################################################################
#########################################################################

intraclasscorr  <-  0.60 #cut-off for intraclass correlation

signifcut <- 0.05      #significance cut-off

data(liverdata)

data(liver_pheno)

OBJECT=new("aclinicalProteomicsData")

OBJECT@rawSELDIdata=as.matrix(liverdata)
OBJECT@covariates=c("tumor" ,    "sex")
OBJECT@phenotypicData=as.matrix(liver_pheno)
OBJECT@variableClass=c('numeric','factor','factor')
OBJECT@no.peaks=53


sampleSize(OBJECT,intraclasscorr,signifcut)

#########################################################################
#########################################################################
#########################################################################
```

sampleSize-methods *~~ Methods for Function sampleSize*

Description

Methods for function sampleSize are defined with: the class object, "aclinicalProteomicsData", intraclasscorr, and signifcut, in the signature.

Methods

Data = "aclinicalProteomicsData", intraclasscorr = "numeric", signifcut = "numeric" aclinicalProteomicsData is a class object for the mass spectrometry data sets, which are in the same format as the raw data from the Biomarkers wizard software. It has slots of matrices of raw mass spectrometry and phenotypic data sets, a character variable for the classes of all the covariates in the phenotypic data matrix, a character variable for the covariates of interest, and numeric value for the number of peaks of interest. See also aclinicalProteomicsData-class. intraclasscorr is a numeric variable for intraclass correlation, and signifcut is the desired significance value for declearing a protein to be differentially expressed.

sampleSize3DscatterPlots *A function for 3D display of sample size in a multi parameter space*

Description

Displays the sample sizes computed using the clinically important parameters. This plot complements the contours plot.

Usage

```
sampleSize3DscatterPlots(Z,m,DIFF,VAR,beta,alpha,observedDIFF,observedVAR,observedSampleSize,Angle,
```

Arguments

Z	the heterogeneity correction factor.
m	the number of replicates
DIFF	the clinically important difference.
VAR	the protein variance.
beta	the power to estimated the clinically important difference.
alpha	the significance level.
observedDIFF	the clinically important difference from your pilot data.
observedVAR	the clinically important variance from your pilot data.

`observedSampleSize`
: the sample size estimated from your pilot data.

`Angle`
: the angle for setting the orientation of the 3D-scatterplot.

`Indicator`
: An indicator variable for controlling items to include in the plot. If it takes the value 1, then the parameters and sample size of previous proteomic profiling studies together with the results from your pilot study are plotted as points on the sample size calculation grid. If it is set to 0, then only the latter will be plotted.

Value

It returns a 3D plot of sample size against the variance versus differences.

Author(s)

Stephen Nyangoma

References

1. Nyangoma SO, Ferreira JA, Collins SI, Altman DG, Johnson PJ, and Billingham LJ: Sample size calculations for planning clinical proteomic profiling studies using mass spectrometry. Bioinformatics, 2009, Submitted

2. Nyangoma SO, Collins SI, Douglas GW, Altman DG, Johnson PJ, and Billingham LJ: Issues in sample size calculations for designing cancer proteomic profiling studies. BMC Bioinformatics, 2009, Submitted

See Also

sampleSizeContourPlots

Examples

```
# the plot will be saved in your working directory

Z <- 2.460018

m <- 2

######

DIFF <- seq(0.1,0.50,0.01)
VAR <- seq(0.2,4,0.1)
beta <- c(0.90,0.80,0.70)
alpha  <-  1 - c(0.001, 0.01,0.05)/2

####

observedDIFF  <-  0.4
observedVAR  <-  1.0
observedSampleSize  <-  80
```

```
#########
# indicator for including results of previous studies on the 3D plot.

Indicator  <-  1

# sets the orientation of the 3D plot.

Angle  <-  60

sampleSize3DscatterPlots(Z,m,DIFF,VAR,beta,alpha,observedDIFF,observedVAR,observedSampleSize,Angle,Indicator)
```

`sampleSizeContourPlots` *A function to construct a grid with contours for calculating sample size in multi dimensional parameter space*

Description

This function draws a grid for calculating the sample size based on the clinically important values of variances versus differences. Based on the analysis of data from past proteomic profiling studies of cancer, we define the clinically important parameters as the summary statistics of the intensities of the peaks with medium biological variation. On the grid, you may display the parameter values from a wide range of real-life data from past proteomic profiling studies, including: data from urine and serum samples of early- and late-stage colorectal cancer patients; serum samples of colorectal cancer patients assayed on four SELDI chip-types (IMAC, H50, Q10 and CM10); plasma samples from Limanda limanda fish; and urine samples of colorectal cancer patients analysed using both SELDI and MALDI sample processing protocols. These values may be used as guidelines for choosing the sample size calculation parameters. If your study involves profiling samples from late-stage disease or sera assayed on the IMAC chip, then the sample size is probably a value close to that of the outer left contour. The urine profiling studies require more samples to detect differences and the value of the contours to the right of grid may be used as bounds. You may also display parameters and sample size from your pilot study in this grid by inputting a vector (`observedPara`) of consensus values of the variance and the corresponding difference, or `rbind` several vectors of such parameters into a matrix/dataframe if you have multiple pilots.

Usage

```
sampleSizeContourPlots(Z,m,DIFF,VAR,beta,alpha,observedPara,Indicator)
```

Arguments

`Z`	the heterogeneity correction factor.
`m`	the number of replicates.
`DIFF`	the clinically important difference.
`VAR`	the protein variance.
`beta`	the power to estimate the clinically important difference.

alpha	the significance level.
observedPara	a vector or a matrix/dataframe (if there is more than one pilot study) containing the variance(s) and the clinically important difference(s) observed from your pilot data. The first element (column) of the vector (matrix) contains the observed variances, while the second contains the information on the clinically important difference(s).
Indicator	an indicator variable. If it is set to 1, then the results of previous proteomic profiling studies together with the results of your pilot study are included in the plot. If it is set to 0, it leads to a plot of only the latter.

Value

Plots of grids of variance versus the clinically important differences with sample size contours superimposed on it.

Author(s)

Stephen Nyangoma

References

1. Nyangoma SO, Ferreira JA, Collins SI, Altman DG, Johnson PJ, and Billingham LJ: Sample size calculations for planning clinical proteomic profiling studies using mass spectrometry. Bioinformatics, 2009, Submitted

2. Nyangoma SO, Collins SI, Douglas GW, Altman DG, Johnson PJ, and Billingham LJ: Issues in sample size calculations for designing cancer proteomic profiling studies. BMC Bioinformatics, 2009, Submitted

Examples

```
# The plot will be saved in your working directory.
# On the grid, we have plotted a number of sample sizes we computed from real life data.
#From these values you can gauge how many samples you may need.
# Fewer samples than 50, will not result in any meaningful estimation of differences.
# For late-stage cancer you  need the fewest samples, even from a very variable sample such as urine.
# You need more samples, over 200, to estimate differences between early stage cancer and
#noncancer controls.
#etc.
m <- 2

DIFF <- seq(0.1,0.50,0.01) # 0.01

VAR <- seq(0.2,4,0.1)

beta <- c(0.90,0.80,0.70)

alpha  <-  1 - c(0.001, 0.01,0.05)/2

Corr <- c(0.70,0.90) #intraclass correlation also fixed
```

```
Z <- 2.6   # fix at 2.6 or use FisherInformation(???)

# You may input parameters from your pilot study. Suppose they are:

#observedPara=c(1,0.4) #the variance you computed from pilot data

observedPara <- data.frame(var=c(0.7,0.5,1.5),DIFF=c(0.37,0.33,0.43))

# you may set these values to 0, if you do not have pilot data
#observedVAR=0
#observedDIFF=0
# in this case the values computed from my pilot studies (dotted on the plot)
# may be used as guidelines.

Indicator <- 0 #1

sampleSizeContourPlots(Z,m,DIFF,VAR,beta,alpha,observedPara,Indicator)
```

`sampleSizeParameters` *A generic function to calculate sample size parameters*

Description

This generic function computes input parameters for the sample size calculation function.

Usage

```
sampleSizeParameters(Data,intraclasscorr,signifcut, ...)
```

Arguments

`Data`	An object of `aclinicalProteomicsData` class.
`intraclasscorr`	An object of `numeric` class. It is a known value of the intraclass correlation, or an estimate from a pilot data.
`signifcut`	An object of `numeric` class. It is significance threshold (usually, taken to be 0.05 in the analysis of the protein profiling studies).
`...`	Some methods for this generic function may take additional, optional arguments. At present none do.

Value

A list of parameters:

`Corr`	the intraclass correlation from your pilot data.
`techVar`	the technical variance from your pilot data.
`bioVar`	the biological variance from your pilot data.
`DIFF`	the clinically important difference from your pilot data.
`no.peaks`	the number of peaks detected by the Biomarker wizard.

Author(s)

Stephen Nyangoma

References

Nyangoma SO, Ferreira JA, Collins SI, Altman DG, Johnson PJ, and Billingham LJ: Sample size calculations for planning clinical proteomic profiling studies using mass spectrometry. Bioinformatics, 2009, Submitted

Smyth GK, et al.: Use of within-array replicate spots for assessing differential expression in microarray experiments. Bioinformatics 2005, 21, 2067 - 75

Smyth GK: Linear models and emperical Bayes methods for assessing differential expression in microarray experiments. Stat Appl Genet Mol Biol 2004, 3, 1, Article 3

Examples

```
intraclasscorr  <-  0.60 #cut-off for intraclass correlation

signifcut <- 0.05      #significance cut-off

data(liverdata)

data(liver_pheno)

OBJECT=new("aclinicalProteomicsData")

OBJECT@rawSELDIdata=as.matrix(liverdata)
OBJECT@covariates=c("tumor" ,    "sex")
OBJECT@phenotypicData=as.matrix(liver_pheno)
OBJECT@variableClass=c('numeric','factor','factor')
OBJECT@no.peaks=53


sampleSizeParameters(OBJECT,intraclasscorr,signifcut)
```

---

sampleSizeParameters-methods

~~ Methods for Function sampleSizeParameters

Description

Methods for function `sampleSize` are defined with: the class object, "aclinicalProteomicsData", intraclasscorr, and signifcut, in the signature.

Methods

Data = "aclinicalProteomicsData", intraclasscorr = "numeric", signifcut = "numeric" `aclinicalProteomicsData` is a class object for the mass spectrometry data sets, which are in the same format as the raw data from the Biomarkers wizard software. It has slots of matrices of raw mass spectrometry and phenotypic data sets, a character variable for the classes of all the covariates in the phenotypic data matrix, a character variable for the covariates of interest, and numeric value for the number of peaks of interest. See also `aclinicalProteomicsData-class`. `intraclasscorr` is a numeric variable for intraclass correlation, and `signifcut` is the desired significance value for declearing a protein to be differentially expressed.

---

sample_technicalVariance

A generic function for computing the technical variance

Description

This generic function computes the within-sample (technical) variance. It fits a simple regression model for repeated measures using the `mixedModel2` function in the `statmod` package. The technical variance is the block component of the `varcomp` output.

Usage

```
sample_technicalVariance(Data, ...)
```

Arguments

`Data`	An object of `aclinicalProteomicsData` class.
`...`	Some methods for this generic function may take additional, optional arguments. At present none do.

Value

It returns a vector of the within-sample variances, one for each peak.

Author(s)

Stephen Nyangoma

Examples

```
#arrange the data in a form that can be averaged by limma function dupcor
# use the function called limmaData
data(liverdata)

data(liver_pheno)

OBJECT=new("aclinicalProteomicsData")
```

```
OBJECT@rawSELDIdata=as.matrix(liverdata)
OBJECT@covariates=c("tumor" , "sex")
OBJECT@phenotypicData=as.matrix(liver_pheno)
OBJECT@variableClass=c('numeric','factor','factor')
OBJECT@no.peaks=53

sample_technicalVariance(OBJECT)
```

`sample_technicalVariance-methods` *Methods for Function sample_technicalVariance*

Description

Methods for function `sample_technicalVariance` are defined with class "aclinicalProteomicsData" in the signature.

Methods

Data = "aclinicalProteomicsData" This is a class object for the mass spectrometry data sets, which are in the same format as the raw data from the Biomarkers wizard software. It has slots of matrices of raw mass spectrometry and phenotypic data sets, a character variable for the classes of all the covariates in the phenotypic data matrix, a character variable for the covariates of interest, and numeric value for the number of peaks of interest. See also `aclinicalProteomicsData-class`.

`show-methods` *Methods for Function show in Package 'methods'*

Description

Methods for function `show` in Package 'methods'.

Methods

object = "aclinicalProteomicsData" This is a class object for the mass spectrometry data sets, which are in the same format as the raw data from the Biomarkers wizard software. It has slots of matrices of raw mass spectrometry and phenotypic data sets, a character variable for the classes of all the covariates in the phenotypic data matrix, a character variable for the covariates of interest, and numeric value for the number of peaks of interest.

`spectrumFilter` *A function to filter out samples with conflicting pair-wise compound information*

Description

This function filters out samples whose spectra have poor pairwise correlations. These samples give conflicting TIC information.

Usage

```
spectrumFilter(Data,threshold,no.peaks)
```

Arguments

`Data`	a data frame of duplicate peak data, in the format of "raw .csv" data from the Biomarker wizard. The original data frame to have the following columns: `SampleTag`, `CancerType`, `Spectrum.`, `Peak`. `Intensity` and `Substance.Mass`.
`threshold`	indicates the threshold for rejecting samples whose spectra contain conflicting signal information.
`no.peaks`	the number of peaks.

Value

A data frame with two columns: the first contains the IDs of the samples meeting the threshold and the second contains the corresponding correlations (i.e. similarities in peak information across spectra).

Author(s)

Stephen Nyangoma

References

Ward DG, Nyangoma S, Joy H, Hamilton E, Wei W, Tselepis C, Steven N, Wakelam MJ, Johnson PJ, Ismail T, Martin A: Proteomic profiling of urine for the detection of colon cancer. Proteome Sci. 2008, 16(6):19

Examples

```
#####################################################
#####################################################

data(liverRawData) # raw version of liverdata
#############################################################################################
#############################################################################################
# These samples pre-processed to:
#  (i) discard the information on samples which have no replicate data, and
```

```
# (ii) for samples with more than 2 replicate expression data, only duplicates with most
#      similar peak information are retained for use in subsequent analyses.
# A wrapper function for executing these two pre-processing steps is preProcRepeatedPeakData
#################################################################################################
#################################################################################################

threshold <- 0.80
no.replicates <- 2
no.peaks <- 53
Data <- preProcRepeatedPeakData(liverRawData, no.peaks, no.replicates, threshold)

head(spectrumFilter(Data,threshold,no.peaks))
```

`ztwo` *A function to compute Z values when there are no covariates other than the cancer class*

Description

This function computes the effects of covariates (Z values) needed when designing a study in simple cases where the only covariate available is the cancer class. The Z values computed may, however, be used in more complex setups, when additional covariates are expected.

Usage

```
ztwo(x, y)
```

Arguments

`x`	expected proportion of cases
`y`	expected proportion of controls

Value

produces the Z value

Author(s)

Stephen Nyangoma

References

Nyangoma SO, Ferreira JA, Collins SI, Altman DG, Johnson PJ, and Billingham LJ: Sample size calculations for planning clinical proteomic profiling studies using mass spectrometry. (Working paper)

See Also

function ZvaluescasesVcontrolsPlots

Examples

```
x=1/3;y=1-x
ztwo(x,y)
```

`ZvaluescasesVcontrolsPlots`

A function for ploting the odds of being a case vs control and their effects on adjustments for confounders

---

Description

A function for ploting the odds of being a case vs control and their effects on adjustments for confounders. It may be useful in cases when it is envisaged that no confounders are expected. It automatically plots the values of Z for the common experimental designs (e.g. 1:1, 1:3 and 1:4). You input alarge number of hypothetical ratios of proportions of cases to controls. It uses another function (ztwo), which computes the Z values.

Usage

```
ZvaluescasesVcontrolsPlots(probs)
```

Arguments

`probs`	A vector of a large number of hypothetical ratios of proportions of cases to controls.

Value

It returns a pdf plot of the odds of being a case vs control and their effects on adjustments for confounders.

Author(s)

Stephen Nyangoma

References

Nyangoma SO, Ferreira JA, Collins SI, Altman DG, Johnson PJ, and Billingham LJ: Sample size calculations for planning clinical proteomic profiling studies using mass spectrometry. (Working paper)

See Also

function ztwo

Examples

```
 # provide hypothetical proportions of cases vs controls
probs=seq(0,1,0.01)
ZvaluescasesVcontrolsPlots(probs)
```

`ZvaluesfrommultinomPlots`

A generic functon to plot Density of Z values from a simulation from a multinomial population using the balanced and unbalanced studies and a 3D representaion of the Z values

---

Description

A functon to plot Density of Z values from a simulation from a multinomial population using the balanced and unbalanced studies and a 3D representaion of the Z values. These plots are useful as visual tools for the confounding effects. The median values are indicated on these plots and these can be used as the consesus values of the effects of covariates in sample size calculations.

Usage

```
ZvaluesfrommultinomPlots(nsim,nobs,proposeddesign,balanceddesign,...)
```

Arguments

`nsim`	number of simulations to be done
`nobs`	number of multinomial observation
`proposeddesign`	a numeric vector with four elements indicating the design weights
`balanceddesign`	a numeric vector with all the four elements being one, indicating equal weights
`...`	other arguments

Details

This function generates saples from a given four cell multinomial populaton then uses the resulting multinomial probabilities in calculating the effect of covariates (Z values). Currently we implement a design arising from a proteomics study in which there is a binary confounder and a binary exposure. The cross-tabulation of the categories of these covariates results into a 4-cell multinomial categories.

Value

`density plot`	Density plot of the Z values
`3D plot of Z values`	3D plot of the Z values against a two dimensional subspace of the 3-D space of multinomial probabilities

Author(s)

Stephen Nyangoma

References

Nyangoma SO, Ferreira JA, Collins SI, Altman DG, Johnson PJ, and Billingham LJ: Sample size calculations for planning clinical proteomic profiling studies using mass spectrometry. (Working paper)

See Also

Also see the function f.

Examples

```
#density plots

nsim=10000;nobs=300;proposeddesign=c(1,2,1,7);balanceddesign=c(1,1,1,1)

f=function (x,y,z) {
Z=(1-x-z)*(x+y)/(2*(((1-x-z)*(1-x-y)*(1-y-z))-(1-x-y-z)^2))
Z
}

mul_1=rmultinom(nsim, nobs, prob = proposeddesign)/nobs
mul_1=t(mul_1)

mul_1=data.frame(mul_1)



names(mul_1)=c('x','y','z')


x=mul_1$x
y=mul_1$y

z=mul_1$z

# compute Z values (see Nyangoma et al. 2009)

Z=f(x,y,z)


x1=x
y1=y

z1=Z


########################################
#####  pr=c(1,1,1,1)# balanced design
```

```
########################################

mul_2=rmultinom(nsim, nobs, prob = balanceddesign)/nobs
mul_2=t(mul_2)


mul_2=data.frame(mul_2)
names(mul_2)=c('x','y','z')

x=mul_2$x
y=mul_2$y

z=mul_2$z

Zb=f(x,y,z)

x2=x
y2=y

z2=Zb

####################################
####################################
#summary(Zb)
pdf('ZvaluesDensityPlots.pdf')
densityZ=density(Z,bw=0.1)
densityZb=density(Zb,bw=0.1)

plot(density(Zb,bw=0.1),xlim=c(min(c(densityZ$x,densityZb$x)),max(c(densityZ$x,densityZb$x))),
ylim=c(min(c(densityZ$y,densityZb$y)),max(c(densityZ$y,densityZb$y))),col='blue',lwd=2,lty=1,xlab='confounding


lines(density(Z,bw=0.1),lwd=2,xlab='',main='')

abline(v=median(Z),lty=2,col='red')
abline(v=median(Zb),lty=2,col='green')

legend(max(c(densityZ$x,densityZb$x)) - 2, max(c(densityZ$y,densityZb$y)) - 0.2, legend=c("blanced","unblanced")
dev.off()
####################################
####################################

library(lattice)
library(rgl)
 library(scatterplot3d)
a=c(x1,x2)
 b=c(y1,y2)
 Z1=c(z1,z2)

group=c(rep(1,10000),rep(2,10000))

Data=data.frame(a,b,Z=Z1,group)
```

```
Data$group=as.factor(Data$group)

Plot3D=cloud(b ~ a*Z,scales = list(arrows = FALSE), data=Data, group=group,screen = list(x = 30, y = -60),ylim=c(0,

Plot3D
nsim=10000;nobs=300;proposeddesign=c(1,2,1,7);balanceddesign=c(1,1,1,1)
ZvaluesfrommultinomPlots(nsim,nobs,proposeddesign,balanceddesign)
```

Index

∗ **Z values**
ztwo, 44
∗ **classes**
aclinicalProteomicsData-class, 6
∗ **confounders**
ztwo, 44
∗ **datasets**
liverdata, 16
∗ **generic**
ZvaluesfrommultinomPlots, 46
∗ **methods**
betweensampleVariance-methods, 10
fisherInformation-methods, 15
proteomicsExprsData-methods, 28
proteomicspData-methods, 30
replicateCorrelations-methods, 31
sample_technicalVariance-methods, 42
sampleSize-methods, 35
sampleSizeParameters-methods, 40
show-methods, 42
∗ **package**
clippda-package, 3
∗ **plot**
ZvaluescasesVcontrolsPlots, 45

aclinicalProteomicsData *(aclinicalProteomicsData-class)*, 6
aclinicalProteomicsData-class, 6
aclinicalProteomicsData-methods, 7

betweensampleVariance, 8
betweensampleVariance,aclinicalProteomicsData-method *(betweensampleVariance-methods)*, 10
betweensampleVariance-methods, 10
biologicalVariance *(betweensampleVariance)*, 8

checkNo.replicates, 11
clippda *(clippda-package)*, 3
clippda-package, 3

f, 12
fisherInformation, 13
fisherInformation,aclinicalProteomicsData-method *(fisherInformation-methods)*, 15
fisherInformation-methods, 15

intraclassCorrelations *(replicateCorrelations)*, 30

liver_pheno, 20
liverdata, 16
liverRawData, 18

mostSimilarTwo, 21

negativeIntensitiesCorrection, 22

pheno_urine, 24
phenoDataFrame, 23
preProcRepeatedPeakData, 25
proteomicsExprsData, 27
proteomicsExprsData,aclinicalProteomicsData-method *(proteomicsExprsData-methods)*, 28
proteomicsExprsData-methods, 28
proteomicspData, 29
proteomicspData,aclinicalProteomicsData-method *(proteomicspData-methods)*, 30
proteomicspData-methods, 30

replicateCorrelations, 30
replicateCorrelations,aclinicalProteomicsData-method *(replicateCorrelations-methods)*, 31
replicateCorrelations-methods, 31

sample_technicalVariance, 41

sample_technicalVariance,aclinicalProteomicsData-method
(sample_technicalVariance-methods), 42
sample_technicalVariance-methods, 42
sampleClusterdData, 32
sampleClusteredData
(sampleClusterdData), 32
sampleSize, 33
sampleSize,aclinicalProteomicsData,numeric,numeric-method
(sampleSize-methods), 35
sampleSize-methods, 35
sampleSize3DscatterPlots, 35
sampleSizeContourPlots, 37
sampleSizeParameters, 39
sampleSizeParameters,aclinicalProteomicsData,numeric,numeric-method
(sampleSizeParameters-methods), 40
sampleSizeParameters-methods, 40
show,aclinicalProteomicsData-method
(show-methods), 42
show-methods, 42
spectrumFilter, 43

withinsampleVariance
(sample_technicalVariance), 41

ztwo, 44
ZvaluescasesVcontrolsPlots, 45
ZvaluesfrommultinomPlots, 46