Package 'ChAMP'

September 10, 2026

Type Package

Title Chip Analysis Methylation Pipeline for Illumina HumanMethylation450 and EPIC

Version 2.43.0

Date 2020-11-2

Description The package includes quality control metrics, a selection of normalization methods and novel methods to identify differentially methylated regions and to highlight copy number alterations.

License GPL-3

VignetteBuilder knitr

Depends R (>= 3.3), minfi, ChAMPdata (>= 2.6.0),DMRcate, Illumina450ProbeVariants.db,IlluminaHumanMethylationEPICmanifest, DT, RPMM

Imports prettydoc,Hmisc,globaltest,sva,illuminaio,rmarkdown,IlluminaHumanMethylation450kmanifest,IlluminaHumanMeth
limma, DNAcopy, preprocessCore,impute, marray, wateRmelon, plyr,goseq,missMethyl,ggplot2, GenomicRanges,qvalue,isva,doParallel,bumphunter,quadprog,shiny,shinythemes,plotly (>= 4.5.6),RColorBrewer,dendextend, matrixStats,combinat

biocViews Microarray, MethylationArray, Normalization, TwoChannel, CopyNumber, DNAMethylation

Suggests knitr,rmarkdown

Author Yuan Tian [cre,aut], Tiffany Morris [ctb], Lee Stirling [ctb], Andrew Feber [ctb], Andrew Teschendorff [ctb], Ankur Chakravarthy [ctb]

Maintainer Yuan Tian <champ450k@gmail.com>

NeedsCompilation no

LazyData true

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

git_branch devel

git_last_commit a448898

git_last_commit_date 2026-04-28

Repository Bioconductor 3.24

Date/Publication 2026-09-10

Contents

ChAMP-package 2
Block.GUI 3
champ.Block 5
champ.CNA 6
champ.DMP 8
champ.DMR 10
champ.filter 14
champ.GSEA 16
champ.import 18
champ.impute 19
champ.load 20
champ.norm 22
champ.process 24
champ.QC 26
champ.refbase 27
champ.runCombat 28
champ.SVD 30
CpG.GUI 31
DMP.GUI 32
DMR.GUI 34
QC.GUI 35

Index **37**

`ChAMP-package` *ChAMP-Chip Analysis Methylation Pipeline*

Description

A pipeline that enables pre-processing of 450K or EPIC data, a selection of normalization methods and a bundle of analysis method including SVD checking, Batch effect correction, DMP, DMR, Block detection, Cell proportion detection, GSEA pathway detection, EpiMod module detection, and copy number variance detection. ChAMP provided a very comprehensive analysis pipeline for EPIC or 450K data set.

Details

Package:	ChAMP
Type:	Package
Version:	2.8.6
Date:	2017-07-19
License:	GPL-3

The full analysis pipeline can be run with all defaults using champ.process() Alternatively, it can be run in steps using all functions separately.

Author(s)

Yuan Tian, Tiffany Morris, Lee Stirling, Andy Feber, Andrew Teschendorff, Ankur Chakravarthy, Stephen Beck
Maintainer: Yuan Tian <champ450k@gmail.com>

Examples

```
directory=system.file('extdata',package='ChAMPdata')
champ.process(directory=directory)
   ### run champ functions separately.
   myLoad <- champ.load(directory)
   myImpute <- champ.impute()
   champ.QC()
   myNorm <- champ.norm()
   champ.SVD()
   myCombat <- champ.runCombat()
   myDMP <- champ.DMP()
   myDMR <- champ.DMR()
   myBlock <- champ.Block()
   myGSEA <- champ.GSEA()
   myCNA <- champ.CNA()
   myRefbase <- champ.refbase() ### for blood sample only

   CpG.GUI()
   QC.GUI()
   DMP.GUI()
   DMR.GUI()
   Block.GUI()
```

`Block.GUI` *Generate interactive plot for the result of champ.Block() function,*

Description

A Shiny, Plotly and Web Brower based analysis interface. Block.GUI() is aimed to provide a comprehensive interactive analysis platform for the result of champ.Block(). The left panel indicate parameters user may be used to select significant Block, here I only provided minium number of clusters and p value as two threshold cutoff. After opening this web page, user may select their cutoff, then press submit, the webpage would calculate the result automatically. User could check the Blocktable in first tab easily, users can rank and select certain genes in the table, the content of the table might be changed based on the cutoff you selected in left panel. The second tab provide the mapping information from CpGs to Blocks, which will makes your easier to find connection between CpGs to clusters then Blocks. The third tab is the plot of Block and the clusters' differential methylation information, you may search the Block you want to check by left panel, note that if there is only one significant cluster in the Block you selected, the plot might not be show properly.

Usage

```
Block.GUI(Block=myBlock,
          beta=myNorm,
          pheno=myLoad$pd$Sample_Group,
          runDMP=TRUE,
          compare.group=NULL,
          arraytype="450K")
```

Arguments

Block	The result from champ.Block(). (default = myBlock)
beta	A matrix of values representing the methylation scores for each sample (M or B). Better to be imputed and normalized data. (default = myNorm)
pheno	This is a categorical vector representing phenotype of factor wish to be analysed, for example "Cancer", "Normal"... Tow or even more phenotypes are allowed. (default = myLoadpdSample_Group)
runDMP	If DMP result sould be calculated and combined into the result of CpGs annotation.
compare.group	compare.group is a parameter to assign which two phenotypes you wish to analysis, if your pheno contains only 2 phenotyes you can leave it as NULL, but if your pheno contains multiple phenotypes, you MUST specify compare.group. (default = NULL)
arraytype	Choose microarray type is 450K or EPIC. (default = "450K")

Value

Totally three tabs would be generated on opened webpage.

Blocktable	The Block list of all significant Blocks selected by cutoff in left panel.
CpGtable	Information of all significant CpGs selected by cutoff in left panel. More importantly, it also contains mapping information each between CpG ID, Cluster ID and Block ID.
BlockPlot	Dots and lines of all clusters involved in one Block, the xaix is based on real Map information of clusters. Above the plot, is the differential methylation information of clusters contained in this Block.

Note

Please make sure you are running R locally or connected with local graph software(X11) remotely.

Author(s)

Yuan Tian

Examples

```
## Not run:
    myLoad <- champ.load(directory=system.file("extdata",package="ChAMPdata"))
    myNorm <- champ.norm()
    myBlock <- champ.Block()
    Block.GUI()

## End(Not run)
```

`champ.Block` *Identify Differential Methylation Blocks in Illumina HumanMethylation450 or HumanMethylationEPIC data.*

Description

This function would detect all methylation Blocks exist in your dataset, methylation Block should be calculated based on the average value of clusters across whole genome. Firstly champ.Block would calculate all clusters in the dataset with clustermaker() function provided by Bumphunter package. Then, only OpenSea Clusters would be picked out to calculate Block. Block can be seen as "large clusters" generated from all small OpenSea Clusters. The algrorithms is similar to the normal DMR-detection one. We will firstly collapse all OpenSea Clusters (or to say regions) into one dot on genome, using average beta value to represent their beta value, and using average position to represent their position. Then we do clustering on these collapsed regions with Bumphunter algrorithms but bigger ranges.

Usage

```
champ.Block(beta=myNorm,
            pheno=myLoad$pd$Sample_Group,
            arraytype="450K",
            maxClusterGap=250000,
            B=500,
            bpSpan=250000,
            minNum=10,
            cores=3)
```

Arguments

`beta`	A matrix of values representing the methylation scores for each sample (M or B). Better to be imputed and normalized data. (default = myNorm)
`pheno`	This is a categorical vector representing phenotype of factor wish to be analysed, for example "Cancer", "Normal"... Tow or even more phenotypes are allowed. (default = myLoadpdSample_Group)
`arraytype`	Choose microarray type is 450K or EPIC. (default = "450K")
`maxClusterGap`	Max gap between clusters when calculating region at first step. (default = 250000)
`B`	An integer denoting the number of resamples to use when computing null distributions. If `permutations` is supplied that defines the number of permutations/bootstraps and `B` is ignored. (default = 250)
`bpSpan`	The maximum length for a Block should be detected, regions longer then this would be discarded. (default = 250000)
`minNum`	Threshold to filtering Blocks with too few probes in it. After region detection, champ.Block will only select Blocks contain more than minNum clusters(OpenSea Regions) to continue the program. (default = 10)
`cores`	The embeded DMR detection function, bumphunter, could automatically use more parallel to accelerate the program. User may assgin number of cores could be used on users's computer. User may use `detectCore()` function to detect number of cores in total. (default = 3)

Value

Block	A data.frame contains all detected Blocks, with colnames as chr, start, end, value, area, cluster, indexStart, indexEnd, L, clusterL, p.value, fwer, p.valueArea, fwerArea. The result format is actually the same as Bumphunter, you may refer to Bumphunter packages to get more explaination about the result.
clusterInfo	When champ.Block() detection significant Blocks, a group of candidate Blocks would be detected out at first, this is the data frame of all candidate Blocks. The "TRUE" Blocks in above value are located in these candidate Blocks.
allCLID.v	The first step of detectiong methylation Blocks is to get each probes into a cluster(region). This value is the clustering result of each probes.
avbetaCL.m	The beta matrix for each cluster. The value is calculated by taking mean value of all probes located in each cluster.
posCL.m	Position of each cluster, which is calculated by average all probes' position in each cluster.

Note

The internal structure of the result of champ.Block() function should not be modified if it's not necessary caused it would be assigned as inpute for some other functions like Block.GUI(). You can try to use Block.GUI() to do interactively analysis on the result of champ.Block().

Author(s)

Yuan Tian

References

Hansen KD, Timp W, Bravo HC, et al. Increased methylation variation in epigenetic domains across cancer types. Nat Genet. 2011;43(8):768-775.

Examples

```
## Not run:
    myLoad <- champ.load(directory=system.file("extdata",package="ChAMPdata"))
    myNorm <- champ.norm()
    myBlock <- champ.Block()
    Block.GUI()

## End(Not run)
```

---

champ.CNA *Inference of Copy Number Abberrations from intensity values.*

Description

This function enables CNA profiles to be built using methylation data from Illumina HumanMethylation450K and HumanMethylationEPIC BeadChips. This function provide options to find Copy Number Abberrations between two phenotype (.e.g. Cancer & Normal), or the function would take the average value of your dataset as control and detect if some value are out of average status. For user want to detect abberrations between phenotypes, they can specify controlGroup in parameter, or they can simply used packaged dataset as control. Two kinds of plot would be returned,

the abberrations of each sample, and the abberrations of each phenotype. The older version of ChAMP provide batchcorrect for intensity dataset, but it's nolonger provided here, user may use champ.runCombat() function to correct batch effect just like they correct beta matrix.

Usage

```
champ.CNA(intensity=myLoad$intensity,
          pheno=myLoad$pd$Sample_Group,
          control=TRUE,
          controlGroup="champCtls",
          sampleCNA=TRUE,
          groupFreqPlots=TRUE,
          Rplot=FALSE,
          PDFplot=TRUE,
          freqThreshold=0.3,
          resultsDir="./CHAMP_CNA",
          arraytype="450K")
```

Arguments

intensity	A matrix of intensity values for each sample. (default = myLoad$intensity)
pheno	This is a categorical vector representing phenotype of factor wish to be analysed, for example "Cancer", "Normal"...Tow or even more phenotypes are allowed. (default = myLoadpdSample_Group)
control	If champ.CNA() should calculate the difference between groups(controls and case) of not(with average). (default = TRUE)
controlGroup	which phenotype in your pheno parameter shall be treated as control type is you want to comparision between two groups. If this value was missing or invalid, the function would automatically use packaged Blood sample(champCtls) as control. (default = "champCtls")
sampleCNA	If sampleCNA=TRUE, then each sample's Copy Number Abberrations would be calculated and plotted. (default = TRUE)
groupFreqPlots	If groupFreqPlots=TRUE, then each group's Copy Number Abberrations Frequence would be calculated and plotted. (default = TRUE)
freqThreshold	If groupFreqPlots=T, then freqThreshold will be used as the cutoff for calling a gain or loss. (default = 0.3)
PDFplot	If PDFplot would be generated and save in resultsDir. (default = TRUE)
Rplot	If Rplot would be generated and save in resultsDir. Note if you are doing analysis on a server remotely, please make sure the server could connect your local graph applications. (For example X11 for linux.) (default = TRUE)
arraytype	Choose microarray type is 450K or EPIC.
resultsDir	The directory where PDF files would be saved. (default = "./CHAMP_CNA/")

Value

sampleResult	The Copy Number Abberrations result calculated and ploted for each Sample.
groupResult	The Copy Number Abberrations result calculated and ploted for each Group.

Author(s)

Feber, A
adapted by Yuan Tian

References

Feber, A et. al. (2014). CNA profiling using high density DNA methylation arrays. Genome Biology.

Examples

```
## Not run:
    myLoad <- champ.load(directory=system.file("extdata",package="ChAMPdata"))
    myCNA <- champ.CNA()

## End(Not run)
```

---

champ.DMP *Identify Differential Methylation Positions (DMP) and Numeric Covariate related CpGs in Illumina HumanMethylation450 or HumanMethylationEPIC data.*

Description

New modification: champ.DMP() can now find numeric variable related CpGs, and do pairwise comparision between more than 2 phenotypes' covariate. This function would use limma package to calculate differential methylation probes between two phenotypes. Or use linear regression model to calcuate CpGs related with certain variables. Now in new version champ.DMP() we still have compare.group parameter, but if compare.group is NULL, and user's pheno variable contains more than 2 phenotypes, champ.DMP() would calculate pairwise DMP between each pair of them. Note that the result of champ.DMP() would be used as inpute of champ.GSEA() and DMP.GUI() function, thus we suggest user not change the internal structure of the result of champ.DMP() function.

Usage

```
champ.DMP(beta = myNorm,
          pheno = myLoad$pd$Sample_Group,
          compare.group = NULL,
          adjPVal = 0.05,
          adjust.method = "BH",
          arraytype = "450K")
```

Arguments

beta	A matrix of values representing the methylation scores for each sample (M or B). Better to be imputed and normalized data. (default = myNorm)
pheno	Covariate that you want to do analysis, it might be a categorical vector representing phenotype of factor wish to be analysed, for example "Cancer", "Normal"... Tow or even more phenotypes are allowed. Or it can be a numeric variable like age. (default = myLoadpdSample_Group)

`compare.group` If your pheno is categorical variable, you may specify this parameter to ask champ.DMP() only compare certain two phenotypes. If your pheno contains more than 2 phenotypes, and compare.group is NULL, pairwise comparision would be done between each two phenotypes. You may set the value as compare.group=c("C","T"), it must be a vector contains only two charactor element. (default = NULL)

`adjPVal` The minimum threshold of significance for probes to be considered an DMP. (default = 0.05)

`adjust.method` The p-value adjustment method to be used for the limma analyis, (default= BH (Benjamini-Hochberg))

`arraytype` Choose microarray type is 450K or EPIC. (default = "450K")

Value

`DMP` A list DMP results. Each element in the list is a data frame of all probes with an adjusted p-value for significance of differential methylation containing columns for logFC, AveExpr, t, P.Value, adj.P.Val, B, C_AVG, T_AVG, deltaBeta, CHR, MAPINFO, Strand, Type, gene, feature, cgi, feat.cgi, UCSC_CpG_Islands_Name, DHS, Enhancer, Phantom, Probe_SNPs, Probe_SNPs_10. These values are directly calculated from limma package, user may read limma manual for more information. deltaBeta is the same as logFC, we kept it here cause maybe old users would stil using it. XXX_AVG is mean value of XXX pheno type in your pheno parameter. Note for numeric variables, the returned result will be named as "NumericVariable", it contains most features as output for categorical covariates except for XXX_AVG and deltaBeta

Note

The internal structure of the result of champ.DMP() function should not be modified if it's not necessary caused it would be assigned as inpute for some other functions like DMP.GUI(), champ.DMR() or champ.GSEA(). You can try to use DMP.GUI() to do interactively analysis on the result of champ.DMP().

Author(s)

Yuan Tian

References

Ritchie, ME, Phipson, B, Wu, D, Hu, Y, Law, CW, Shi, W, and Smyth, GK (2015). limma powers differential expression analyses for RNA-sequencing and microarray studies. Nucleic Acids Research 43(7), e47

Phipson, B, Lee, S, Majewski, IJ, Alexander, WS, and Smyth, GK (2016). Robust hyperparameter estimation protects against hypervariable genes and improves power to detect differential expression. Annals of Applied Statistics 10(2), 946-963.

Examples

```
## Not run:
    myLoad <- champ.load(directory=system.file("extdata",package="ChAMPdata"))
    myNorm <- champ.norm()
    myDMP <- champ.DMP()
    DMP.GUI()
```

```
## End(Not run)
```

---

champ.DMR *Applying Bumphunter, DMRcate or ProbeLasso Algorithms to detect Different Methylation Regions in a beta valued Methylation Dataset.*

Description

Applying Bumphunter, DMRcate or ProbeLasso Algorithms to Estimate regions for which a genomic profile deviates from its baseline value. Originally implemented to detect differentially methylated genomic regions between two populations. By default, we recommend user do champ.DMR on normalized beta value on two populations, like case to control. The function will return detected DMR and estimated p value. The three algorithms specified in this function is different, while Bumphunter and DMRcate calcuated averaged candidate bumps methylation value between case and control. Thus parameters is different for three algorithms. Note that the result of champ.DMR() would be used as inpute of champ.GSEA() function, thus we suggest user not change the internal structure of the result of champ.DMR() function.

Usage

```
champ.DMR(beta=myNorm,
          pheno=myLoad$pd$Sample_Group,
          compare.group=NULL,
          arraytype="450K",
          method = "Bumphunter",
          minProbes=7,
          adjPvalDmr=0.05,
          cores=3,
          ## following parameters are specifically for Bumphunter method.
          maxGap=300,
          cutoff=NULL,
          pickCutoff=TRUE,
          smooth=TRUE,
          smoothFunction=loessByCluster,
          useWeights=FALSE,
          permutations=NULL,
          B=250,
          nullMethod="bootstrap",
     ## following parameters are specifically for probe ProbeLasso     method.
          meanLassoRadius=375,
          minDmrSep=1000,
          minDmrSize=50,
          adjPvalProbe=0.05,
          Rplot=T,
          PDFplot=T,
          resultsDir="./CHAMP_ProbeLasso/",
          ## following parameters are specifically for DMRcate method.
          rmSNPCH=T,
          fdr=0.05,
          dist=2,
```

```
                mafcut=0.05,
                lambda=1000,
                C=2)
```

Arguments

Since there are three methods incoporated to detect DMRs, user may specify which function to do DMR detection, Bumphunter DMRcate or ProbeLasso. All three methods are available for both 450K and EPIC beadarray. But they are controled by different parameters, thus users shall be careful when they specify parameters for corresponding algorithm. Parameters shared by three algorithms:

`beta`	Methylation beta valueed dataset user want to detect DMR. We recommend to use normalized beta value. In Bumphunter method, beta value will be transformed to M value. NA value is NOT allowed into this function, thus user may need to do some imputation work beforehead. This parameter is essential for both two algorithms. (default = myNorm)
`pheno`	This is a categorical vector representing phenotype of factor wish to be analysed, for example "Cancer", "Normal"... Tow or even more phenotypes are allowed. (default = myLoadpdSample_Group)
`compare.group`	ProbeLasso Method does not allow pheno contains more than 2 phenotypes, so if your want use ProbeLasso method, but pheno parameter contains more than 2 phenotypes, you MUST specify compare.group as "compare.group=c("A","B")" to make sure ProbeLasso only works on ONLY two phenotypes. If your pheno parameter contains only 2 phenotypes, you can leave it as NULL. (default=NULL)
`arraytype`	Choose microarray type is 450K or EPIC. (default = "450K")
`method`	Specify the method users want to use to do DMR detection. There are three options: "Bumphunter", "DMRcate" or "ProbeLasso". (default = "Bumphunter").
`minProbes`	Threshold to filtering clusters with too few probes in it. After region detection, champ.DMR will only select DMRs contain more than minProbes to continue the program. (default = 7)
`adjPvalDmr`	This is the significance threshold for including DMRs in the final DMR list. (default = 0.05)
`cores`	The embeded DMR detection function, bumphunter and DMRcate, could automatically use more parallel to accelerate the program. User may assgin number of cores could be used on users's computer. User may use `detectCore()` function to detect number of cores in total. (default = 3)

Parameters specific for Bumphunter algorithm:

`maxGap`	The maximum length for a DMR should be detected, regions longer then this would be discarded. (default = 300)
`cutoff`	A numeric value. Values of the estimate of the genomic profile above the cutoff or below the negative of the cutoff will be used as candidate regions. It is possible to give two separate values (upper and lower bounds). If one value is given, the lower bound is minus the value. (default = NULL)
`pickCutoff`	A bool value to indicate if bumphunter algorithm will automatically select the threshold of DMRs. If the value is TRUE, bumphunter will automatically generated 0.99 cutoff from permutation. If user think this threshold is not suitable, user may set their own cutoff here. (default = TRUE)
`smooth`	A logical value. If TRUE the estimated profile will be smoothed with the smoother defined by `smoothFunction`. (default = TRUE)

smoothFunction A function to be used for smoothing the estimate of the genomic profile. Two functions are provided by the package: loessByCluster and runmedByCluster. (default = loessByCluster)

useWeights A logical value. If TRUE then the standard errors of the point-wise estimates of the profile function will be used as weights in the loess smoother loessByCluster. If the runmedByCluster smoother is used this argument is ignored. (default = FALSE)

permutations is a matrix with columns providing indexes to be used to scramble the data and create a null distribution when nullMethod is set to permutations. If the bootstrap approach is used this argument is ignored. If this matrix is not supplied and B>0 then these indexes are created using the function sample. (default = NULL)

B An integer denoting the number of resamples to use when computing null distributions. If permutations is supplied that defines the number of permutations/bootstraps and B is ignored. (default = 250)

nullMethod Method used to generate null candidate regions, must be one of 'bootstrap' or 'permutation' (defaults to 'permutation'). However, if covariates in addition to the outcome of interest are included in the design matrix (ncol(design)>2), the 'permutation' approach is not recommended. See vignette and original paper for more information. (default = "bootstrap")

Parameters specific for ProbeLasso algorithm:

meanLassoRadius
Radius around each DMP to detect DMR. (default = 375)

minDmrSep The minimum seperation (bp) between neighbouring DMRs. (default = 1000.)

minDmrSize The minimum DMR size (bp). (default = 50)

adjPvalProbe The minimum threshold of significance for probes to be includede in DMRs. (default = 0.05)

PDFplot If PDFplot would be generated and save in resultsDir. (default = TRUE)

Rplot If Rplot would be generated and save in resultsDir. Note if you are doing analysis on a server remotely, please make sure the server could connect your local graph applications. (For example X11 for linux.) (default = TRUE)

resultsDir The directory where PDF files would be saved. (default = "./CHAMP_ProbeLasso/")

Parameters specific for Dmrcate algorithm:

rmSNPCH Filters a matrix of M-values (or beta values) by distance to SNP. Also (optionally) removes crosshybridising probes and sex-chromosome probes. (default = TRUE)

fdr FDR cutoff (Benjamini-Hochberg) for which CpG sites are individually called as significant. Used to index default thresholding in dmrcate(). Highly recommended as the primary thresholding parameter for calling DMRs.

dist Maximum distance (from CpG to SNP) of probes to be filtered out. See details for when Illumina occasionally lists a CpG-to-SNP distance as being < 0. (default = 2)

mafcut Minimum minor allele frequency of probes to be filtered out. (default = 0.05)

lambda Gaussian kernel bandwidth for smoothed-function estimation. Also informs DMR bookend definition; gaps >= lambda between significant CpG sites will be in separate DMRs. Support is truncated at 5*lambda. See DMRcate package for further info. (default = 1000)

C — Scaling factor for bandwidth. Gaussian kernel is calculated where lambda/C = sigma. Empirical testing shows that when lambda=1000, near-optimal prediction of sequencing-derived DMRs is obtained when C is approximately 2, i.e. 1 standard deviation of Gaussian kernel = 500 base pairs. Cannot be < 0.2. (default = 2)

Value

myDmrs — A data.frame in a list contains Different Methylation Regions detected by champ.DMR. For different algorithms, myDmrs would be in different structure and named as "BumphunterDMR", "DMRcateDMR" and "ProbeLassoDMR". They may contain some different informations, caused by their method. However all three kinds of result are already suitable for champ.GSEA() analysis, so please don't modify the stucture if it's not necessary.

Note

The internal structure of the result of champ.DMR() function should not be modified if it's not necessary caused it would be assigned as inpute for some other functions like champ.GSEA(). You can try to use DMR.GUI() to do interactively analysis on the result of champ.DMR().

Note

The internal structure of the result of champ.DMR() function should not be modified if it's not necessary caused it would be assigned as inpute for some other functions like DMR.GUI() and champ.GSEA(). You can try to use DMR.GUI() to do interactively analysis on the result of champ.DMR().

Author(s)

Butcher, L,Aryee MJ, Irizarry RA, Andrew Teschendorff, Yuan Tian

References

Jaffe AE et a. Bump hunting to identify differentially methylated regions in epigenetic epidemiology studies. Int J Epidemiol. 2012;41(1):200-209.

Butcher LM, Beck S. Probe lasso: A novel method to rope in differentially methylated regions with 450K dna methylation data. Methods. 2015;72:21-28.

Peters TJ, Buckley MJ, Statham AL, et al. De novo identification of differentially methylated regions in the human genome. Epigenetics & Chromatin. 2015;8(1):1-16.

Examples

```
    ## Not run:
        myLoad <- champ.load(directory=system.file("extdata",package="ChAMPdata"))
        myNorm <- champ.norm()
        myDMR <- champ.DMR()
        DMR.GUI()

## End(Not run)
```

champ.filter *Do filtering on beta, M, intensity, Meth and UnMeth matrix.*

Description

Do filtering on beta, M, intensity, Meth and UnMeth matrix. So that user who have no IDAT file can also do filtering. This function has been totally recoded, firstly it is not take result from champ.import() as input and do filtering on that. So that user can use champ.import() + champ.filter() to generate data set. The other purpose of doing that is user can do any filtering on any of above 5 matrix, as long as they have a single matrix. Not that some accessory data sets are required for some methods, like you MUST provide detect P value matrix so that champ.filtering can do filtering on detect P value. Also you have to provide beadcount information so that champ.filtering can do filtering on beadcount. Also remember that, if you want to keep pd file in accord with your data matrix, you can surely input pd, but make sure Sample_Name of your pd file is EXACTLY the same as your data matrix's colnames. Also keep in mind that, if you want to do filtering on multiple data matrix, you MUST make sure they have EXACTLY the same rownames and colnames. The function would filtering all matrix at the same time, so keeping the two names same make sure champ.filter() is not doing wrong filtering on different data sets.

Usage

```
champ.filter(beta=myImport$beta,
             M=NULL,
             pd=myImport$pd,
             intensity=NULL,
             Meth=NULL,
             UnMeth=NULL,
             detP=NULL,
             beadcount=NULL,
             autoimpute=TRUE,
             filterDetP=TRUE,
             ProbeCutoff=0,
             SampleCutoff=0.1,
             detPcut=0.01,
             filterBeads=TRUE,
             beadCutoff=0.05,
             filterNoCG = TRUE,
             filterSNPs = TRUE,
             population = NULL,
             filterMultiHit = TRUE,
             filterXY = TRUE,
             fixOutlier = TRUE,
             arraytype = "450K")
```

Arguments

beta	One single beta matrix to do filtering. (default = myImport$beta).
M	One single M matrix to do filtering. (default = NULL).
pd	pd file related to this beta matrix, suggest provided, because maybe filtering would be on pd file. (default = myImport$pd)

intensity	intensity matrix. (default = NULL).
Meth	Methylated matrix. (default = NULL).
UnMeth	UnMethylated matrix. (default = NULL).
detP	Detected P value matrix for corresponding beta matrix, it MUST be 100% corresponding, which can be ignored if you don't have.(default = NULL)
beadcount	Beadcount information for Green and Red Channal, need for filterBeads.(default = NULL)
autoimpute	If after detect P filtering, some NA are still exist in your data set (Only beta or M matrix), should imputation be done one them. Should only be done on big data set. Before do imputation, checking process would be done ahead to make sure Detect P, ProbeCutoff, beta or M valule are exist. (default = TRUE)
filterDetP	If filter = TRUE, then probes above the detPcut will be filtered out.(default = TRUE)
SampleCutoff	The detection p value threshhold for samples. Samples with above proportion of failed p value will be removed. (default = 0.1)
ProbeCutoff	The detection p value threshhold for Probe. After removing failed Samples(controled by SampleCutoff parameter), probes with above proportion of failed p value will be removed.(default = 0)
detPcut	The detection p-value threshhold. Probes about this cutoff will be filtered out. (default = 0.01)
filterBeads	probes with less then 3 beads would be set NA. If for one probe, number of NAs above certian ratio, filtering would be conducter on that probe. (default = TRUE)
beadCutoff	Ratio threshhold that a probe should be removed for failed in beadcount check (default = 0.05).
filterNoCG	If filterNoCG=TRUE, non-cg probes are removed.(default = TRUE)
filterSNPs	If filterSNPs=TRUE, probes in which the probed CpG falls near a SNP as defined in Nordlund et al are removed.(default = TRUE)
population	If you want to do filtering on specifical populations you may assign this parameter as one of "AFR","EAS"... The full list of population is in http://www.internationalgenome.org/cate (default = TRUE)
filterMultiHit	If filterMultiHit=TRUE, probes in which the probe aligns to multiple locations with bwa as defined in Nordlund et al are removed.(default = TRUE)
filterXY	If filterXY=TRUE, probes from X and Y chromosomes are removed.(default = TRUE)
fixOutlier	If fixOutlier=TRUE, in beta matrix only, value below 0 would be replaced as minium positive value, would value above 1 would be replaced as maxium value below 1.(default = TRUE)
arraytype	Choose microarray type is "450K" or "EPIC".(default = "450K")

Value

Objects	A list of data sets you want to filtering and inputted into this function.

Author(s)

Yuan Tian

References

Zhou W, Laird PW and Shen H: Comprehensive characterization, annotation and innovative use of Infinium DNA Methylation BeadChip probes. Nucleic Acids Research 2016

Examples

```
## Not run:
    myimport <- champ.import(directory=system.file("extdata",package="ChAMPdata"))
 myfilter <- champ.filter(beta=myImport$beta,pd=myImport$pd,detP=myImport$detP,beadcount=myImport$beadc

## End(Not run)
```

`champ.GSEA` *Do GSEA for DMP, DMR and other methylation data related results.*

Description

This function would do GSEA on the results of champ functions like DMP and DMR. However users may also add individual CpGs and genes in it. There are three method are incoporated into champ.GSEA function here. One is old Fisher Exact Test method, which will used information downloaded from MSigDB and do fisher exact test to calculated the enrichment status for each pathways.

Usage

```
champ.GSEA(beta=myNorm,
           DMP=myDMP[[1]],
           DMR=myDMR,
           CpGlist=NULL,
           Genelist=NULL,
           pheno=myLoad$pd$Sample_Group,
           method="fisher",
           arraytype="450K",
           Rplot=TRUE,
           adjPval=0.05,
           cores=1)
```

Arguments

`beta`	A matrix of values representing the methylation scores for each sample (M or B). Better to be imputed and normalized data. (default = myNorm)
`DMP`	Results from champ.DMP() function. (default = myDMP)
`DMR`	Results from champ.DMR() function. (default = myDMR)
`CpGlist`	Apart from previous parameters, if you have any other CpGs list want to do GSEA, you can input them here as a list. (default = NULL)
`Genelist`	Apart from previous parameters, if you have any other Gene list want to do GSEA. you can inpute them here as a list. (default = NULL)
`pheno`	If use ebayes method, user needs to provide phenotype information to conduct global test. (default = myLoadpdSample_Group)

method	Which method would be used to do GSEA?"gometh","fisher". (default = "fisher")
arraytype	Which kind of array your data set is? (default = "450K")
Rplot	If gometh method was chosen, should Probability Weight plot will be plotted. More information please check gometh package. (default = TRUE)
adjPval	Adjusted p value cutoff for all calculated GSEA result. (default = 0.05)
cores	Number of parallel threads/cores used in ebayes method. (default = 1)

Value

For fisher Method:

Genelist	List of pathway we get by enriching genes onto annotation database.
nOVLAP	Number of genes overlapped in your significant gene list and annotated pathways.
OR	Odds Ratio calculated for each enrichment.
P-value	Significance calculated from fisher exact test.
adjPval	Adjusted P value from "BH" method.
Genes	Name of genes enriched in each pathway.

For gometh method, the returned value are:

category	GO pathway's index.
over_represented_pvalue	The p value for genes' over representing in this pathway.
under_represented_pvalue	The p value for genes' under representing in this pathway.(Not likely to be used)
numDEInCat	Numbers of Different Methylation Genes in this pathway.
numInCat	Numbers of all genes related to this pathway.
term	The short explaination for this pathway.
ontology over_represented_adjPvalue	The ajusted over representing p value with "BH" method. User may used this one to select qualitied Pathways.

Author(s)

Yuan Tian, Danyue Dong

Examples

```
    ## Not run:
        myLoad <- champ.load(directory=system.file("extdata",package="ChAMPdata"))
        myNorm <- champ.norm()
        myDMP <- champ.DMP()
        myDMR <- champ.DMR()
        myGSEA <- champ.GSEA()

## End(Not run)
```

`champ.import` *Read data from IDAT file.*

Description

Function provided by ChAMP to extract value from IDAT file, and mapping between CpGs and Probes on Chip. The older version of ChAMP used minfi to load data, this is a version provided by ChAMP. The function would read data from one directory, which contains IDAT files and phenotype data csv files. Then champ.import() would firstly read the csv file, mapping between each sample and IDAT file. Then champ.import() would read IDAT file for each sample. After reading Green and Red Channal, Meth Matrix, UnMeth Matrixn beta value, intensity, detect P value, bead count would be calculated. Above are matrix would be used in champ.filter(). Note that, champ.import() would NOT do batch correction. And data read by champ.import() can not be used for SWAN normalization and FunctionNormalization in champ.norm() function. If user want to use SWAN, you may still consider champ.load() function, but remember to set "method" parameter as "minfi", which is "ChAMP" in default.

Usage

```
champ.import(directory = getwd(),
             offset=100,
             arraytype="450K")
```

Arguments

`directory`	Location of IDAT files, default is current working directory.(default = getwd())
`offset`	offset is set to make sure no inf value would be returned.(default = 100)
`arraytype`	Choose microarray type is "450K" or "EPIC".(default = "450K")

Value

`beta`	A matrix of beta methylation scores for all probes and all samples (No filtering has been don).
`M`	A matrix of M methylation scores for all probes and all samples (No filtering has been done).
`pd`	pd file of all sample information from Sample Sheet, which would be very frequently by following functions as DEFAULT input, thus it's not very necessarily, please don't modify it.
`intensity`	A matrix of intensity values for all probes and all samples, the information would be used in champ.CNA() function. It has not been filtered. Actually, intensity are the sum of Meth Matrix and UnMeth Matrix.
`detP`	A matrix of detection p-values for all probes and all samples.
`beadcount`	A matrix beads for each probe on each sample. Value less then 3 has been set NA.
`Meth`	Methylated Matrix for all probe and all samples.
`UnMeth`	UnMethylated Matrix for all probe and all samples.

Author(s)

Yuan Tian

Examples

```
    ## Not run:
        myimport <- champ.import(directory=system.file("extdata",package="ChAMPdata"))

## End(Not run)
```

`champ.impute` *Conduct imputation for NA value on beta matrix and corresponding pd(Sample_sheet.csv) file.*

Description

champ.impute will conduct imputation on beta matrix contains missing value. This function can be used for any beta dataset, along with their corresponding pd files. If you loaded this file with champ.load(), champ.impute() function will automatically loaded myLoad$beta as inputted beta matrix, while take myLoad$pd as pd input. There are totally three method provided in champ.impute() function. "Delete" is simply remove all NA related CpGs and Samples contain certian proportion of missing value, which is suitable for Small DataSets. "KNN" method use impute.knn() function from "impute" to do imputation on all missing value, which is rather popular but would cause trouble if DataSets contains few samples, no CpGs or samples woule be deleted. "Combine" method would remove all Samples and CpGs with certian proportions of missing value, then do KNN imputation for the rest (Default).

Usage

```
champ.impute(beta=myLoad$beta,
             pd=myLoad$pd,
             method="Combine",
             k=5,
             ProbeCutoff=0.2,
             SampleCutoff=0.1)
```

Arguments

`beta`	Data matrix want to be imputed, user can input M matrix or intensity matrix even. (default = myLoad$beta)
`pd`	Phenotype file for your data set. It's optional for this function, but if during imputation some samples contain too many NA values dicarded, your old pd file might not be able to work for imputed data properly any more. (default = myLoad$pd)
`method`	Imputation method optional, only "Combine","KNN","Delete" are feasible. (default = "Combine").
`k`	Number of neighbors to be used in the imputation (default = 5)
`ProbeCutoff`	Proportion of for probes shall be removed. Any probes with NA value proportion above this parameter will be removed. (default = 0.2)
`SampleCutoff`	Proportion of for Sample shall be removed. Any Sample with NA value proportion above this parameter will be removed. (default = 0.1)

Value

beta	The matrix get imputed
pd	The pd file corresponding to imputed matrix, if provided.

Author(s)

Yuan Tian

Examples

```
    ## Not run:
        myLoad <- champ.load(directory=system.file("extdata",package="ChAMPdata"))
        myImpute <- champ.impute()

## End(Not run)
```

---

champ.load *Upload of raw HumanMethylation450K or HumanMethylationEPIC data from IDAT files.*

Description

Function that loads data from IDAT files to calculate intensity. Some kinds of filtering will be conducted as well such as unqualied CpGs, SNP, multihit sites, and XY chromosomes related CpGs. In new version champ.load() function, we provided a new loading method, which is coded by ChAMP group. User may set "method" parameter as "minfi" to use old minfi way. Note that new "ChAMP" would NOT return rgSet and mset as "minfi" object, only pain matrix or data frame would be returned, which makes it easier to intepret the result, but it also means current ChAMP can not works on "SWAN" and "FunctionNormalization" method in champ.norm(), you can still use "BMIQ" and "PBC" method though.

Usage

```
champ.load(directory = getwd(),
           method="ChAMP",
           methValue="B",
           autoimpute=TRUE,
           filterDetP=TRUE,
           ProbeCutoff=0,
           SampleCutoff=0.1,
           detPcut=0.01,
           filterBeads=TRUE,
           beadCutoff=0.05,
           filterNoCG=TRUE,
           filterSNPs=TRUE,
           population=NULL,
           filterMultiHit=TRUE,
           filterXY=TRUE,
           force=FALSE,
           arraytype="450K")
```

Arguments

directory	Location of IDAT files, default is current working directory.(default = getwd())
method	Method to load data, "ChAMP" method is newly provided by ChAMP group, while "minfi" is old minfi way.(default = "ChAMP")
methValue	Indicates whether you prefer m-values M or beta-values B. (default = "B")
autoimpute	If after filtering (or not do filtering) there are NA values in it, should impute.knn(k=3) should be done for the rest NA?
filterDetP	If filter = TRUE, then probes above the detPcut will be filtered out.(default = TRUE)
ProbeCutoff	The NA ratio threshhold for probes. Probes with above proportion of NA will be removed.
SampleCutoff	The failed p value (or NA) threshhold for samples. Samples with above proportion of failed p value (NA) will be removed.
detPcut	The detection p-value threshhold. Probes about this cutoff will be filtered out. (default = 0.01)
filterBeads	If filterBeads=TRUE, probes with a beadcount less than 3 will be removed depending on the beadCutoff value.(default = TRUE)
beadCutoff	The beadCutoff represents the fraction of samples that must have a beadcount less than 3 before the probe is removed.(default = 0.05)
filterNoCG	If filterNoCG=TRUE, non-cg probes are removed.(default = TRUE)
filterSNPs	If filterSNPs=TRUE, probes in which the probed CpG falls near a SNP as defined in Nordlund et al are removed.(default = TRUE)
population	If you want to do filtering on specifical populations you may assign this parameter as one of "AFR","EAS"... The full list of population is in http://www.internationalgenome.org/cate (default = TRUE)
filterMultiHit	If filterMultiHit=TRUE, probes in which the probe aligns to multiple locations with bwa as defined in Nordlund et al are removed.(default = TRUE)
filterXY	If filterXY=TRUE, probes from X and Y chromosomes are removed.(default = TRUE)
force	A parameter in minfi's read.metharray.exp function, if your arrays are not coming from same batch, force parameter would allow you to select their common probes and do analysis on them.(default = FALSE)
arraytype	Choose microarray type is "450K" or "EPIC".(default = "450K")

Value

mset	mset object from minfi package, with filtering CpGs discarded.
rgSet	rgset object from minfi package function read.metharray.exp(), contains all information of a .idat methylation dataset. If you want to do more analysis than functions provided by ChAMP, you can take this as a start point.
pd	pd file of all sample information from Sample Sheet, which would be very frequently by following functions as DEFAULT input, thus it's not very necessarily, please don't modify it.
intensity	A matrix of intensity values for all probes and all samples, the information would be used in champ.CNA() function. CpGs has been filtered as well.
beta	A matrix of methylation scores (M or beta values) for all probes and all samples.
detP	A matrix of detection p-values for all probes and all samples.

Author(s)

Yuan Tian

References

Aryee MJ, Jaffe AE, Corrada-Bravo H, Ladd-Acosta C, Feinberg AP, Hansen KD and Irizarry RA (2014). Minfi: A flexible and comprehensive Bioconductor package for the analysis of Infinium DNA Methylation microarrays. Bioinformatics, 30(10), pp. 1363-1369. doi: 10.1093/bioinformatics/btu049.

Jean-Philippe Fortin, Timothy Triche, Kasper Hansen. Preprocessing, normalization and integration of the Illumina HumanMethylationEPIC array. bioRxiv 065490; doi: https://doi.org/10.1101/065490

Zhou W, Laird PW and Shen H: Comprehensive characterization, annotation and innovative use of Infinium DNA Methylation BeadChip probes. Nucleic Acids Research 2016

Examples

```
## Not run:
    myLoad <- champ.load(directory=system.file("extdata",package="ChAMPdata"))

## End(Not run)
```

`champ.norm` *Normalization for HumanMethylation450 or HumanMethylationEPIC data*

Description

Option to normalize data with a selection of normalization methods. There are four functions could be selected: "PBC","BMIQ","SWAN" and "FunctionalNormalize". SWAN method call for BOTH rgSet and mset input, FunctionNormalization call for rgset only , while PBC and BMIQ only needs beta value. Please set parameter correctly. BMIQ method is the default function, which would also return normalised density plots in PDF format in results Dir. FunctionalNormalize is provided in minfi package, which ONLY support 450K data yet. Not that BMIQ function might fail if you sample's beta value distribution is not beta distribution, which occationally happen when too many CpGs are deleted while loading .idat files with champ.load() function. Also multi-cores parallel is conductable for BMIQ function, if your server or computer is good enought with more than one cores, you may assign more cores like 10 to accelerate the process. No matter what method you selected, they all will return the same result: Normalize beta matrix with effect of Type-I and Type-II probes corrected.

Usage

```
champ.norm(beta=myLoad$beta,
           rgSet=myLoad$rgSet,
           mset=myLoad$mset,
           resultsDir="./CHAMP_Normalization/",
           method="BMIQ",
           plotBMIQ=FALSE,
           arraytype="450K",
           cores=3)
```

Arguments

beta	Original beta matrix waiting to be normalized. NA value are not recommended, thus you may want to use champ.impute to impute data first. colname of each sample MUST be marked. (default = myLoad$beta)
rgSet	Original full information matrix from champ.load(), which is required by "SWAN" and "FunctionNormalization" method. (default = myLoad$rgSet)
mset	mset object from minfi package, with filtering CpGs discarded, which is required by "SWAN" method. (default = myLoad$mset)
resultsDir	The folder where champ.norm()'s PDF file should be saved. (default = "./CHAMP_Normalization/")
method	Method to do normalization: "PBC","BMIQ","SWAN" and "FunctionalNormalize". (default = "BMIQ")
plotBMIQ	If "BMIQ" method is choosen, should champ.norm() plot normalized plot in PDF and save it in resultsDir. (default = FALSE)
arraytype	Choose microarray type is "450K" or "EPIC".(default = "450K")
cores	If "BMIQ" method is choosen, how many cores shall be used to run parallel. (default = 3)

Value

beta.p	A matrix of normalised methylation scores (M or beta values) for all probes and all samples.

Author(s)

Yuan Tian wrote the wrappers

References

Teschendorff AE, Marabita F, Lechner M, Bartlett T, Tegner J, Gomez-Cabrero D, Beck S. A beta-mixture quantile normalization method for correcting probe design bias in Illumina Infinium 450k DNA methylation data. Bioinformatics. 2013 Jan 15;29(2):189-96.

Dedeurwaerder S, Defrance M, Calonne E, Denis H, Sotiriou C, Fuks F.Evaluation of the Infinium Methylation 450K technology. Epigenomics. 2011,Dec;3(6):771-84.

Touleimat N, Tost J. Complete pipeline for Infinium Human Methylation 450K BeadChip data processing using subset quantile normalization for accurate DNA methylation estimation. Epigenomics. 2012 Jun;4(3):325-41.

Fortin J. P. et al. Functional normalization of 450k methylation array data improves replication in large cancer studies. Genome Biol. 15, 503 (2014).

Examples

```
    ## Not run:
        myLoad <- champ.load(directory=system.file("extdata",package="ChAMPdata"))
        myNorm <- champ.norm()

## End(Not run)
```

champ.process *Process function to run all methods in ChAMP pipeline.*

Description

This function allows the user to run the entire pipeline in one function. Arguments allow user to select functions if desired. Note that it maybe run during champ.process() if there is any problem during the process, thus run ChAMP functions one by one is actually recommended.

Usage

```
champ.process(runload=TRUE,
              directory = getwd(),
              filters=c("XY","DetP","Beads","NoCG","SNP","MultiHit"),
              #---champ.impute parameters below---#
              runimpute=TRUE,
              imputemethod="Combine",
              #---champ.QC parameters below---#
              runQC=TRUE,
              QCplots=c("mdsPlot","densityPlot","dendrogram"),
              #---champ.norm parameters below---#
              runnorm=TRUE,
              normalizationmethod="BMIQ",
              #---champ.SVD parameters below---#
              runSVD=TRUE,
              RGEffect=FALSE,
              #---champ.runCombat parameters below---#
              runCombat=TRUE,
              batchname=c("Slide"),
              #---champ.DMP parameters below---#
              runDMP=TRUE,
              #---champ.DMR parameters below---#
              runDMR=TRUE,
              DMRmethod="Bumphunter",
              #---champ.Block parameters below---#
              runBlock=TRUE,
              #---champ.GSEA parameters below---#
              runGSEA=TRUE,
              #---champ.EpiMod parameters below---#
              #---champ.CNA parameters below---#
              runCNA=TRUE,
              control=TRUE,
              controlGroup="champCtls",
              #---champ.refbase parameters below---#
              runRefBase=FALSE,
              #---universal settings---#
              compare.group=NULL,
              adjPVal=0.05,
              resultsDir="./CHAMP_RESULT/",
              arraytype="450K",
              PDFplot=TRUE,
```

```
                    Rplot=TRUE,
                    cores=3,
                    saveStepresults=TRUE)
```

Arguments

runload	If champ.load() should be run? (default = TRUE)
directory	The folder directory of .idat files. (default = getwd())
filters	A character vector indicates filters should be done if load data from .idat files. You can remove some of the filters in it if you don't need that much. (default = c("XY","DetP","Beads","NoCG","SNP","MultiHit"))
runimpute	If champ.impute() should be run? Note that if your data contains too many NA, champ.impute() may remove not only CpGs, but also samples. (default = TRUE)
imputemethod	Which imputation method should be applied into champ.impute().
runQC	If champ.QC() should be run? (default = TRUE)
QCplots	A character vector indicates plots should be drawn by champ.QC(). You can remove some plots in it if you don't need them. (default = c("mdsPlot","densityPlot","dendrogram"))
runnorm	If champ.norm() should be run? (default = TRUE)
normalizationmethod	Which normalization method should be selected by champ.norm().
runSVD	If champ.SVD() should be run? (default = TRUE)
RGEffect	If Red Gree color Effect should be calculated in champ.SVD(). (default = FALSE)
runCombat	If champ.runCombat() should be run? (default = TRUE)
batchname	A character vector indicates what factors should be corrected by champ.runCombat(). (default = c("Slide"))
runDMP	If champ.DMP() should be run? (default = TRUE)
runDMR	If champ.DMR() should be run? (default = TRUE)
DMRmethod	Which DMR method should be applied by champ.DMR()? (default = TRUE)
runBlock	If champ.Block() should be run? (default = TRUE)
runGSEA	If champ.GSEA() should be run? (default = TRUE)
runCNA	If champ.CNA() should be run? (default = TRUE)
control	If champ.CNA() should be calculate copy number variance between case and control? (The other option for champ.CNA() is calculate copy number variance for each sample to the averaged value). (default = TRUE)
controlGroup	Which pheno should be treated as control group while running champ.CNA().(default = "champCtls")
runRefBase	If champ.refbase() should be run? (default = TRUE)
compare.group	Which two phenos should be compared in champ.DMP()?
adjPVal	The adjusted p value for each function's significant cutoff.
resultsDir	The directory where result should be stored. (default = "./CHAMP_RESULT/")
arraytype	If the data set under analysis is "450K" or "EPIC"? (default = "450K")
PDFplot	If PDF files should be plotted during running? (default = TRUE)

Rplot	If R plots should be plotted during running? (default = TRUE)
cores	How many cores should be used for parallel running during champ.process()? (default = 3)
saveStepresults	If result of each steps should be saved as .rd file into resultsDir folder? (default = TRUE)

Value

CHAMP_RESULT	A list contains all results from each champ.method.

Author(s)

Yuan Tian

Examples

```
    ## Not run:
        directory=system.file("extdata",package="ChAMPdata")
        champ.process(directory=directory)

## End(Not run)
```

---

champ.QC *Plot quality control plot, mdsplot, densityPlot, dendrogram for a data set.*

Description

champ.QC() function would plot some summary plot for a dataset, including mdsplot, densityPlot, dendrogram. You may use QC.GUI() function to see even more plot interactively, like heatmap, Type-I and Type-II probes plot. Note that the dendrogram would do it's best to modify plot size automatically, but if you have too many samples like 1000+, the speed would be slow and the plot might be hard to read.

Usage

```
champ.QC(beta = myLoad$beta,
         pheno=myLoad$pd$Sample_Group,
         mdsPlot=TRUE,
         densityPlot=TRUE,
         dendrogram=TRUE,
         PDFplot=TRUE,
         Rplot=TRUE,
         Feature.sel="None",
         resultsDir="./CHAMP_QCimages/")
```

Arguments

beta	beta matrix want to be analysed. NA value are not recommended, thus you may want to use champ.impute to impute data first. colname of each sample MUST be marked. (default = myLoad$beta)
pheno	one Phenotype categorical vector for your dataset. NO list or dataframe or numeric. (default = myLoadpdSample_Group)
mdsPlot	If mdsPlot would be plotted. (default = TRUE)
densityPlot	If densityPlot would be plotted. (default = TRUE)
dendrogram	If dendrogram would be plotted. (default = TRUE)
PDFplot	If PDFplot would be generated and save in resultsDir. (default = TRUE)
Rplot	If Rplot would be generated and save in resultsDir. Note if you are doing analysis on a server remotely, please make sure the server could connect your local graph applications. (For example X11 for linux.) (default = TRUE)
Feature.sel	Featrue Selection method when champ.QC() calculate dendrogram. Two options are provided, "None" means no featrue selection would be done, all probes would be used to calculate distance between each sample. "SVD" method means champ.QC() would firstly do SVD deconvolution on beta dataset, then use Random Theory Matrix mathod in "isva" package to calculated numbers of latent variable, and the "distance" between samples would be calcuated by top components of SVD result (similar to PCA). (default = "None")
resultsDir	The directory where PDF files would be saved. (default = "./CHAMP_QCimages/")

Note

You can try to use QC.GUI() to do similar but interactively analysis.

Author(s)

Yuan Tian

Examples

```
## Not run:
    myLoad <- champ.load(directory=system.file("extdata",package="ChAMPdata"))
    champ.QC()

## End(Not run)
```

---

champ.refbase *Applying References-Base Methold to beta valued methylation data.*

Description

Applying References-Based Methold to correct cell-proportion in a methylation dataset. Reference-based method use purified whole blood cell-type specific methylation value to correct beta value dataset. Cell Proportions for each cell-type will be detected, and lm function will be used to correct beta value for 5 largest cell types. Cell type with smallest cell proportion will not be corrected.

Usage

```
champ.refbase(beta=myNorm,
              arraytype="450K")
```

Arguments

beta	whole blood beta methylation dataset user want to correct. (default = myNorm)
arraytype	There are two types of purified cell-type specific references can be chosen, "450K" and "27K". By default, 450K value will be used, but user may choose 27K as well. (default = myNorm)

Value

CorrectedBea	A beta valued matrix, with all value get corrected with RefBaseEWAS method. Be aware, champ.refbase will only correct top 5 cell types with largest mean cell proportions, and leave the cell with smallest mean cell proportion. User may check CellFraction result to find out which cell types are get corrected.
CellFraction	Proportion for each cell type.

Author(s)

Houseman EA, Yuan Tian, Andrew Teschendorff

References

Houseman EA, Accomando WP, Koestler DC, Christensen BC, Marsit CJ, et al. (2012) DNA methylation arrays as surrogate measures of cell mixture distribution. BMC Bioinformatics 13: 86. doi: 10.1186/1471-2105-13-86. pmid:22568884

Examples

```
## Not run:
    myLoad <- champ.load(directory=system.file("extdata",package="ChAMPdata"))
    myNorm <- champ.norm()
    myRefbase <- champ.refbase()

## End(Not run)
```

---

champ.runCombat — *Function that uses ComBat to correct for batch effects. Multiple batch effect correction is allowed.*

Description

This function formats data to run through ComBat batch correction. If beta values are used the data is first logit transformed. Them Combat function from "sva" package would be used to do batch correction. Note that multi-batch correction is supported, user just need to assign name of batch need to be corrected. Note Combat function is a little bit critical to dataset, thus you have futher question or higher lever of application of Combat, you may turn to "sva" pacakge for help. After inputing pd file, champ.runCombat() would automatically detect all correctable factors and list them below, if your assigned batchname is correct, champ.runCombat() would start to do batch correction. Note that in new version champ.runCombat() function, we will check if user's variable and batch confound with each other.

Usage

```
champ.runCombat(beta=myNorm,
                pd=myLoad$pd,
                variablename="Sample_Group",
                batchname=c("Slide"),
                logitTrans=TRUE)
```

Arguments

beta	A matrix of values representing the methylation scores for each sample (M or B). (default = myNorm).
pd	This data.frame includes the information from the sample sheet. (default = myLoad$pd).
variablename	Variable name which batch should be corrected for, in previous version of ChAMP, variablename was "Sample_Group". (default = "Sample_Group").
batchname	A character vector of name indicates which batch factors shall be corrected. (default = c("Slide"))
logitTrans	If logitTrans=T then your data will be logit transformed before the Combat correction and inverse logit transformed after correction. This is T by default for Beta values but if you have selected M values, it should be FALSE. It is also FALSE when used with CNA as those are intensity values that don't need to be transformed.

Value

beta	The matrix of values represeting the methylation scores for each sample after ComBat batch correction.

Author(s)

Yuan Tian

References

Johnson WE et a. Adjusting batch effects in microarray expression data using empirical bayes methods. Biostatistics. 2007;8(1):118-127.

Examples

```
## Not run:
    myLoad <- champ.load(directory=system.file("extdata",package="ChAMPdata"))
    myNorm <- champ.norm()
    champ.SVD()
    myCombat <- champ.runCombat()

## End(Not run)
```

champ.SVD *Singular Value Decomposition analysis for batch effects prediciton in HumanMethylation450 or HumanMethylationEPIC data*

Description

New modification: We have added a new plot scree plot (proposed by rasmus.rydbirk@regionh.dk), to help user to judge the importance of deconvoluted components. After SVD deconvolution, each components would "explain" part of variances existing in origin data matrix, in other word, your beta matrix. Thus we hope to see some top components (normally 3-5) would have captured most variances existing in your original data. Thus, after champ.SVD(), you may check the PDF file, and see how many components needs to be considered in following anlaysis. For example, if component 1 has captured 80 percent of variance, and it is highly correlated with the phenotype you want to research, you may ignore following components' batch effect. Runs Singular Value Decomposition on a dataset to estimate the impact of batch effects. This function would run SVD deconvolution on beta matrix, get components explain most variance in original data set. Then use Random Matrix Theory to estimate numbers of latent variables. Then each significant components would be correlated with each phenotype, to see if this phenotype show significant correlation with this component. All suitable factors in your pd(Sample_Sheet.csv) file will be analysed. After champ.SVD(), used would get a heatmap indicating effect of factors on original data set. And decide if some batch effect shall be corrected before future analysis. Not all factors in your pd file woule be analysis though, name information like Sample_Name, Pool_ID... would be discarded, covariates contain less then 2 variances shall be discarded as well. Note that numeric covariates like age would be calculated with linear regression, while factors and character covariates like Sample_Group would be calculated with Krustal Test. Thus please check your input pd file carefully as well. We have added legend on plot. In the plot generated by champ.SVD(), color indicates different levels of significance. The darker the color is, the more significant your deconvoluted components are correlated with your phenotype. Also, we modified the number of x axis (number of component) as dimentions of latent variables detected by EstDimRMT() function from "isva" package, however if this function estimated too many components, say more than 20 components, champ.SVD() would automatically selected only top 20 components.

Usage

```
champ.SVD(beta = myNorm,
          rgSet=NULL,
          pd=myLoad$pd,
          RGEffect=FALSE,
          PDFplot=TRUE,
          Rplot=TRUE,
          resultsDir="./CHAMP_SVDimages/")
```

Arguments

beta	beta matrix waiting to be analysed, better to be one get Probe-Type normalized and imputed. (default = myNorm)
rgSet	An rgSet object that was created when data was loaded the data from the .idat files, which contains green and red color information of original data set, might be used if RGEffect set TRUE. (default = myLoad$rgSet)
pd	This data.frame includes the information from the sample sheet. (default = myLoad$pd)

RGEffect	If Green and Red color control probes would be calculated. (default = FALSE)
PDFplot	If PDFplot would be generated and save in resultsDir. (default = TRUE)
Rplot	If Rplot would be generated and save in resultsDir. Note if you are doing analysis on a server remotely, please make sure the server could connect your local graph applications. (For example X11 for linux.) (default = TRUE)
Rplot	If Splot is true, generates Scree plot (elbow plot). If PDFPlot is also true, would be generated and save in resultsDir. (default = TRUE)
resultsDir	The directory where PDF files would be saved. (default = "./CHAMP_SVDimages/")

Author(s)

Teschendorff, A
adapted by Yuan Tian

References

Teschendorff, A. E., Menon, U., Gentry-Maharaj, A., Ramus, S. J., Gayther, S. A., Apostolidou, S., Jones, A., Lechner, M., Beck, S., Jacobs, I. J., and Widschwendter, M. (2009). An epigenetic signature in peripheral blood predicts active ovarian cancer. PLoS One, 4(12), e8274

Examples

```
## Not run:
    myLoad <- champ.load(directory=system.file("extdata",package="ChAMPdata"))
    myNorm <- champ.norm()
    champ.SVD()

## End(Not run)
```

CpG.GUI *Generate interactive plot for summary information of a list of CpGs*

Description

A Shiny, Plotly and Web Brower based analysis interface. CpG.GUI() is aimed to generate summary of a list of CpGs. Feature distribution, CpG island distribution .e.g. It's call for X11 similar graph software locally if you are doing analysis on server. Also the RAM memory might be large if you have a very big dataset. This function can be used anytime you have a list of CpGs from any analysis, you simply need to imput the CpGs and specify the array type, a web brower interactive interface would be generated automatically. The plots are interactive thus you can make easier and better analysis on your data, and also download them at any size (jpg only).

Usage

```
CpG.GUI(CpG=rownames(myLoad$beta),
        arraytype="450K")
```

Arguments

CpG	A list of CpG you want to do plot summary. MUST be a vector with CpG ID. (default = rownames(myLoad$beta))
arraytype	Choose microarray type is 450K or EPIC. (default = "450K")

Value

Totally four plots would be generated on opened webpage.

`chromosome_barplot`
: A chromosome barplot for the CpG list

`feature_barplot`
: A feature barplot for the CpG list

`cgi_barplot`
: A cgi barplot for the CpG list

`type_barplot`
: A type-I and type-II barplot for the CpG list

Note

Please make sure you are running R locally or connected with local graph software(X11) remotely.

Author(s)

Yuan Tian

Examples

```
## Not run:
    myLoad <- champ.load(directory=system.file("extdata",package="ChAMPdata"))
    CpG.GUI()

## End(Not run)
```

`DMP.GUI` *Generate interactive plot for the result of champ.DMP() function,*

Description

New Modification: We now makes DMP.GUI() would detect numeric variables, which means, if you use champ.DMP() calculated some numeric variable related CpGs, you can continuely use this DMP.GUI() function to draw nice plot for these CpGs and genes even. For CpG plot, while for categorical variables, boxplot would be plotted, we will plot scatter plot for numeric variable (like age) now. For gene plot, we will firstly devide your covariates into couple groups (default is 4), then treat it as a categorical variable. By doing this, you may see that your CpGs should significantly difference lines for difference phenotypes. Also, since now champ.DMP() would calculate pairwise comparision for covariate contains more than 2 phenotypes. All result of DMP would be stored into a list, no longer directly myDMP again, so if you have multiple result from champ.DMP(), please inpute each of them here into DMP.GUI(), like DMP.GUI(myDMP[[1]]...), DMP.GUI(myDMP[[2]]...), DMP.GUI(myDMP[[3]]...) A Shiny, Plotly and Web Brower based analysis interface. DMP.GUI() is aimed to provide a comprehensive interactive analysis platform for the result of champ.DMP(). The left panel indicate parameters user may be used to select significant CpGs, here I only provided abslogFC and p value as two threshold cutoff. After opening this web page, user may select their cutoff, then press submit, the webpage would calculate the result automatically. User could check the DMPtable in first tab easily, users can rank and select certain genes in the table, the content of the table might be changed based on the cutoff you selected in left panel. The second tab provide the heatmap of all significant CpGs you selected, be careful that if there are too many CpGs, the memory consumption might be large. The third tab provide barplots of proportions of feature and CpGs in for your selected CpGs. The fourth tab is the plot of gene and

the wikigene information of certain gene, you may search the gene you want to check by left panel, note that if there is only one significant CpG in the gene you selected, the plot might not be show properly. The last panel provide a boxplot of CpGs and a gene enrichment plot, you may use this gene enrichment plot to find interesting genes.

Usage

```
DMP.GUI(DMP=myDMP[[1]],
        beta=myNorm,
        pheno=myLoad$pd$Sample_Group,
        cutgroupnumber=4)
```

Arguments

DMP	The result from champ.DMP(). (default = myDMP)
beta	A matrix of values representing the methylation scores for each sample (M or B). Better to be imputed and normalized data. (default = myNorm)
pheno	This is a categorical vector representing phenotype of factor wish to be analysed, for example "Cancer", "Normal"... Tow or even more phenotypes are allowed. In our new upgrading work, DMP.GUI() also accept numeric variables. (default = myLoadpdSample_Group)
cutgroupnumber	This parameters only works if your pheno parameter is a numeric variable, when DMP.GUI() plot gene plot, we will automatically devide your phenotype into couple groups, then treat it as a categorical variable. You may modify this parameter here to tell DMP.GUI how many groups should be devide. Note that this parameter should be setted based on number of value in your pheno parameter. (default = 4)

Value

Totally five tabs would be generated on opened webpage.

DMPtable	The DMP list of all significant CpGs selected by cutoff in left panel.
Heatmap	Heatmap of all significant CpGs selected by cutoff in left panel.
Feature&CpG	Barplot of feature and Cgi information for all significant CpGs selected by cutoff in left panel.
Gene	Dots and lines of all significant CpGs involved in one gene, the distance between CpGs are equal, and the feature and Cgi information are marked down the plot. Below the plot, is the wikigene information extracted from website.
CpG	Boxplot for CpGs you want to check, you can search CpGs based on the left panel. Below is the gene enrichment plot, hyper CpGs and hyper CpGs are separated.

Note

Please make sure you are running R locally or connected with local graph software(X11) remotely.

Author(s)

Yuan Tian

Examples

```
## Not run:
    myLoad <- champ.load(directory=system.file("extdata",package="ChAMPdata"))
    myNorm <- champ.norm()
    myDMP <- champ.DMP()
    DMP.GUI()

## End(Not run)
```

DMR.GUI *Generate interactive plot for the result of champ.DMR() function,*

Description

A Shiny, Plotly and Web Brower based analysis interface. DMR.GUI() is aimed to provide a comprehensive interactive analysis platform for the result of champ.DMR(). The last panel indicate parameters user may be used to select significant DMRs, here I only provided minprobes and p value as two threshold cutoff. After opening this web page, user may select their cutoff, then press submit, the webpage would calculate the result automatically. User could check the DMRtable in first tab easily, users can rank and select certain genes in the table, the content of the table might be changed based on the cutoff you selected in left panel. The second tab is the CpGtable, which extract all CpGs involved in selected CpGs. Note that maybe not all CpGs are DMPs. The thrid tab provide the plot of the DMR, just like gene plot in DMP.GUI(). Above the plot are CpGs information involved in this DMR. The fourth panel provide a heatmap of all CpGs involved in significant DMRs, and a gene enrichment plot. Both plot maybe not very clear to look, but user may zoom in for these two plots. Again be careful if you have a very big dataset. Note that the runDMP parameters will indicate if DMR.GUI() shall calculated DMP for all CpGs, which may cause slight different in the CpG table and the gene enrichment plot. And though there are three ways to calculate DMR, all three results from champ.DMR() are applicatable for this function. The title would changed automatically for different result.

Usage

```
DMR.GUI(DMR=myDMR,
        beta=myNorm,
        pheno=myLoad$pd$Sample_Group,
        runDMP=TRUE,
        compare.group=NULL,
        arraytype="450K")
```

Arguments

DMR	The result from champ.DMR(), all three DMR methods' result are supported. (default = myDMR)
beta	A matrix of values representing the methylation scores for each sample (M or B). Better to be imputed and normalized data. (default = myNorm)
pheno	This is a categorical vector representing phenotype of factor wish to be analysed, for example "Cancer", "Normal"... Tow or even more phenotypes are allowed. (default = myLoadpdSample_Group)
runDMP	If DMP result sould be calculated and combined into the result of CpGs annotation.

compare.group	compare.group is a parameter to assign which two phenotypes you wish to analysis, if your pheno contains only 2 phenotyes you can leave it as NULL, but if your pheno contains multiple phenotypes, you MUST specify compare.group. (default = NULL)
arraytype	Choose microarray type is 450K or EPIC. (default = "450K")

Value

Totally four tabs would be generated on opened webpage.

DMRtable	The DMR list of all significant DMR you selected by cutoff in left panel.
CpGtable	A CpGs annotation (with p value and t value if runDMP=TRUE) of all CpGs related with selected DMRs in tab 1.
DMRPlot	Dots and lines of all significant CpGs involved in one DMR, the distance between CpGs are equal, and the feature and Cgi information are marked down the plot. Above the plot, is the CpGs list involved in this DMR.
Summary	CpG enrichment gene barplot, hyper CpGs and hyper CpGs may be marked if runDMP=TRUE. Below is the heatmap for all significant DMRs related CpGs. Both plots maybe not that clear but zoomable.

Note

Please make sure you are running R locally or connected with local graph software(X11) remotely.

Author(s)

Yuan Tian

Examples

```
    ## Not run:
        myLoad <- champ.load(directory=system.file("extdata",package="ChAMPdata"))
        myNorm <- champ.norm()
        myDMR <- champ.DMR() # All three methods supported.
        DMR.GUI()

## End(Not run)
```

QC.GUI *Generate interactive plot for Quality Control of a dataset*

Description

A Shiny, Plotly and Web Brower based analysis interface. QC.GUI() is aimed to provide mdsplot, densityPlot, Type-I&Type-II densityplot, dendrogram(no interactable) and heatmap for top 1000 variale CpGs. In the first tab,mdsplot are plotted based on the distance calculated by top 1000 variable CpGs. For dendrogram, if there are only less than 10 samples, the distance between samples are calculated by all CpGs, if there are more than 10 samples, QC.GUI() would apply SVD doconvolution on the dataset first then extract top significant components as latent variabls and calculate distance between samples. For the heatmap, if your dataset contains less than 1000 CpGs, all CpGs would be plotted, but if your dataset contains more than 1000 CpGs, the top 1000 variable CpGs would be selected and plot.

Usage

```
QC.GUI(beta=myLoad$beta,
       pheno=myLoad$pd$Sample_Group,
       arraytype="450K")
```

Arguments

beta	A matrix of values representing the methylation scores for each sample (M or B). Better to be imputed and normalized data. (default = myNorm)
pheno	This is a categorical vector representing phenotype of factor wish to be analysed, for example "Cancer", "Normal"... Tow or even more phenotypes are allowed. (default = myLoadpdSample_Group)
arraytype	Choose microarray type is 450K or EPIC. (default = "450K")

Value

Totally five tabs would be generated on opened webpage.

mdsplot	A mdsplot used to see the clustering result and similarity between sampels.
TypeDensity	A two-line density Plot indicate Type-I CpGs and Type-II CpGs.
QCplot	Beta distribution of each sample. You may use it to check samples with low qualities.
Dendrogram	Dendrogram of all samples. If there are only less than 10 samples, the distance between samples are calculated by all CpGs, if there are more than 10 samples, QC.GUI() would apply SVD doconvolution on the dataset first then extract top significant components as latent variabls and calculate distance between samples.
heatmap	Heatmap for top 1000 variale CpGs.

Note

Please make sure you are running R locally or connected with local graph software(X11) remotely.

Author(s)

Yuan Tian

Examples

```
## Not run:
    myLoad <- champ.load(directory=system.file("extdata",package="ChAMPdata"))
    QC.GUI()

## End(Not run)
```

Index

∗ **450k**
ChAMP-package, 2
∗ **Beadchip**
ChAMP-package, 2
∗ **Block**
champ.Block, 5
∗ **ComBat**
champ.runCombat, 28
∗ **DMR**
champ.DMR, 10
∗ **DNAMethylation**
ChAMP-package, 2
∗ **EPIC**
ChAMP-package, 2
∗ **GSEA**
champ.GSEA, 16
∗ **HumanMethylation450**
ChAMP-package, 2
∗ **RefbaseEWAS**
champ.refbase, 27
∗ **array**
ChAMP-package, 2
∗ **batch effects**
champ.SVD, 30
∗ **copynumber**
champ.CNA, 6
∗ **limma**
champ.DMP, 8
∗ **methylation**
ChAMP-package, 2
∗ **normalization**
champ.norm, 22
∗ **package**
ChAMP-package, 2
∗ **plotly**
Block.GUI, 3
CpG.GUI, 31
DMP.GUI, 32
DMR.GUI, 34
QC.GUI, 35
∗ **shiny**
Block.GUI, 3
CpG.GUI, 31
DMP.GUI, 32
DMR.GUI, 34
QC.GUI, 35

Block.GUI, 3

ChAMP *(*ChAMP-package*)*, 2
ChAMP-package, 2
champ.Block, 5
champ.CNA, 6
champ.DMP, 8
champ.DMR, 10
champ.filter, 14
champ.GSEA, 16
champ.import, 18
champ.impute, 19
champ.load, 20
champ.norm, 22
champ.process, 24
champ.QC, 26
champ.refbase, 27
champ.runCombat, 28
champ.SVD, 30
CpG.GUI, 31

DMP.GUI, 32
DMR.GUI, 34

QC.GUI, 35