Package 'canprot'

May 11, 2020

Date 2020-05-11

Version 1.0.0

Title Compositional Analysis of Differentially Expressed Proteins in Cancer

Maintainer Jeffrey Dick <j3ffdick@gmail.com>

Depends R (>= 3.1.0)

Imports CHNOSZ (>= 1.3.0), xtable, MASS, knitr, rmarkdown

Suggests KernSmooth

Description Compositional analysis of differentially expressed proteins in cancer and cell culture proteomics experiments. The data include lists of up- and down-regulated proteins in different cancer types (breast, colorectal, liver, lung, pancreatic, prostate) and laboratory conditions (hypoxia, hyperosmotic stress, high glucose, 3D cell culture, and proteins secreted in hypoxia), together with amino acid compositions computed for protein sequences obtained from UniProt. Functions are provided to calculate compositional metrics including protein length, carbon oxidation state, and stoichiometric hydration state. In addition, phylostrata (evolutionary ages) of protein-coding genes are compiled using data from Liebeskind et al. (2016) <doi:10.1093/gbe/evw113> or Trigos et al. (2017) <doi:10.1073/pnas.1617743114>. The vignettes contain plots of compositional differences, phylostrata for human proteins, and references for all datasets.

Encoding UTF-8

License GPL (>= 2)

BuildResaveData no

VignetteBuilder knitr

URL http://github.com/jedick/canprot

NeedsCompilation no

Author Jeffrey Dick [aut, cre] (<https://orcid.org/0000-0002-0687-5890>), Ben Bolker [ctb] (<https://orcid.org/0000-0002-2127-0443>)

Repository CRAN

Date/Publication 2020-05-11 11:10:09 UTC

R topics documented:

`canprot-package` *Compositional Analysis of Differentially Expressed Proteins*

Description

canprot is a package for analysis of the chemical compositions of proteins from their amino acid compositions. The package compiles datasets for differentially expressed proteins in cancer and cell culture conditions from over 250 studies. It also has tools for the thermodynamic analysis of differentially expressed proteins using the **CHNOSZ** package.

Overview

This package includes datasets for differential expression of proteins in six cancer types (breast, colorectal, liver, lung, pancreatic, prostate), and four cell culture conditions (hypoxia, hyperosmotic stress, secreted proteins in hypoxia, and 3D compared to 2D growth conditions). The hyperosmotic stress data are divided into bacteria, archaea (both high- and low-salt experiments) and eukaryotes; the latter are further divided into salt and glucose experiments. Nearly all datasets use UniProt IDs; if not given in the original publications they have been added using the UniProt mapping tool (`https://www.uniprot.org/mapping/`).

The vignettes have plots for each cancer type and cell culture condition and references for all data sources used. Because of their size, pre-built vignette HTML files are not included with the package; use `mkvig` to compile and view any of the vignettes.

The functions in this package were originally derived from code supporting the papers of Dick (2016 and 2017); the `oldvignettes` directory has vignettes to reproduce many of the plots in those papers. Updated data compilations and revised metrics, plots, and vignettes were developed for the papers of Dick et al. (2020) and Dick (2020).

References

Dick, J. M. (2016) Proteomic indicators of oxidation and hydration state in colorectal cancer. *PeerJ* **4**, e2238. doi: 10.7717/peerj.2238

Dick, J. M. (2017) Chemical composition and the potential for proteomic transformation in cancer, hypoxia, and hyperosmotic stress. *PeerJ* **5**, e3421. doi: 10.7717/peerj.3421

Dick, J. M., Yu, M. and Tan, J. (2020) Distinct trends in chemical composition of proteins from metagenomes in redox and salinity gradients. *bioRxiv*. doi: 10.1101/2020.04.01.020008

Dick, J. M. (2020) Water as a reactant in the differential expression of proteins in cancer. *bioRxiv*. doi: 10.1101/2020.04.09.035022

See Also

The **JMDplots** package on GitHub (`https://github.com/jedick/JMDplots`) has vignettes showing analysis of data from The Cancer Genome Atlas and The Human Protein Atlas, which are not included here because of the large size of the data files.

Examples

```
# List the data files for all studies
# (one study can have more than one dataset)
exprdata <- system.file("extdata/expression", package="canprot")
datafiles <- dir(exprdata, recursive=TRUE)
print(datafiles)
# Show the number of data files for each condition
table(dirname(datafiles))
```

`check_IDs` *Check UniProt IDs*

Description

Find the first ID for each protein that matches a known UniProt ID.

Usage

```
check_IDs(dat, IDcol, aa_file = NULL, updates_file = NULL)
```

Arguments

`dat`	data frame, protein expression data
`IDcol`	character, name of column that has the UniProt IDs
`aa_file`	character, name of file with additional amino acid compositions
`updates_file`	character, name of file with old to new ID mappings

Details

check_IDs is used to check for known UniProt IDs and to update obsolete IDs. The source IDs should be provided in the IDcol column of dat; multiple IDs for one protein can be separated by a semicolon.

The function keeps the first "known" ID for each protein, which must be present in one of these groups:

- The human_aa dataset of amino acid compositions.
- Old UniProt IDs that are mapped to new UniProt IDs in uniprot_updates or in updates_file if specified.
- IDs of proteins in aa_file, which lists amino acid compositions in the format described for human_aa (see extdata/protein/human_extra.csv for an example and thermo$protein for more details).

Value

dat is returned with possibly changed values in the column designated by IDcol; old IDs are replaced with new ones, the first known ID for each protein is kept, then proteins with no known IDs are assigned NA.

See Also

This function is used extensively by the pdat_ functions, where it is called before cleanup.

Examples

```
# Make up some data for this example
ID <- c("P61247;PXXXXX", "PYYYYY;P46777;P60174", "PZZZZZ")
dat <- data.frame(ID = ID, stringsAsFactors = FALSE)
# Get the first known ID for each protein; the third one is NA
check_IDs(dat, "ID")

# Update an old ID
dat <- data.frame(Entry = "P50224", stringsAsFactors = FALSE)
check_IDs(dat, "Entry")
```

---

cleanup *Clean Up Data*

Description

Remove proteins with unavailable IDs, ambiguous expression ratios, and duplicated IDs.

Usage

```
cleanup(dat, IDcol, up2 = NULL)
```

Arguments

dat	data frame, protein expression data
IDcol	character, name of column that has the UniProt IDs
up2	logical, TRUE for up-regulated proteins, FALSE for down-regulated proteins

Details

cleanup is used in the pdat_ functions to clean up the dataset given in dat. IDcol is the name of the column that has the UniProt IDs, and up2 indicates the expression change for each protein. The function removes proteins with unavailable (NA or "") or duplicated IDs. If up2 is provided, the function also remove unquantified proteins (those that have NA values of up2) and those with ambiguous expression ratios (up and down for the same ID). For each operation, a message is printed describing the number of proteins that are 'unavailable', 'unquantified', 'ambiguous', or 'duplicated'.

Alternatively, if IDcol is a logical value, it selects proteins to be unconditionally removed.

See Also

This function is used extensively by the pdat_ functions, where it is called after check_IDs (if needed).

Examples

```
# Set up a simple workflow
extdatadir <- system.file("extdata", package="canprot")
datadir <- paste0(extdatadir, "/expression/pancreatic/")
dataset <- "CYD+05"
dat <- read.csv(paste0(datadir, dataset, ".csv.xz"), as.is = TRUE)
up2 <- dat$Ratio..cancer.normal. > 1
# Remove two unavailable and one duplicated proteins
dat <- cleanup(dat, "Entry", up2)
# Now we can calculate the chemical compositions
pcomp <- protcomp(dat$Entry)

# Read another data file
datadir <- paste0(system.file("extdata", package="canprot"), "/expression/colorectal/")
dataset <- "STK+15"
dat <- read.csv(paste0(datadir, "STK+15.csv.xz"), as.is = TRUE)
# Remove unavailable proteins
dat <- cleanup(dat, "uniprot")
# Remove proteins that have less than 2-fold expression ratio
dat <- cleanup(dat, abs(log2(dat$invratio)) < 1)
```

CLES *Common Language Effect Size*

Description

Calculate the common language effect size.

Usage

```
CLES(x, y)
```

Arguments

x	numeric, data
y	numeric, data

Details

The common language statistic is defined for continuous data as "the probability that a score sampled at random from one distribution will be greater than a score sampled from some other distribution." (McGraw and Wong, 1992)

This function calculates the fraction of all possible pairings between elements of x and y where the difference ('y' value - 'x' value) is positive.

The example uses synthetic data; the actual data used by McGraw and Wong is from NCHS (1987, Tables 13 and 14).

References

McGraw, Kenneth O. and Wong, S. P. (1992) A common language effect size statistic. *Psychological Bulletin* **11**, 361–365. doi: 10.1037/00332909.111.2.361

National Center for Health Statistics (1987) *Anthropometric Reference Data and Prevalence of Overweight: United States, 1976-1980.* Data from the National Health Survey, Series 11, No. 238. DHHS Publication (PHS) No. 87-1688. U.S. Government Printing Office, Washington, DC. http://www.cdc.gov/nchs/data/series/sr_11/sr11_238.pdf

Examples

```
# Generate synthetic data for heights in inches of 18-24
# year-old males and females
height_male <- rnorm(988, 69.7, 2.8)
height_female <- rnorm(1066, 64.3, 2.6)
# The CLES is approximately 0.92 (McGraw and Wong, 1992)
CLES(height_female, height_male)
```

`diffplot` *Plot Compositional Differences*

Description

Make a plot showing differences of selected compositional metrics.

Usage

```
diffplot(comptab, vars = c("ZC", "nH2O"), col = "black", plot.rect = FALSE,
         pt.text = c(letters, LETTERS), cex.text = 0.85, oldstyle = FALSE,
         pch = 1, cex = 2.1, contour = TRUE, col.contour = par("fg"),
         probs = 0.5, add = FALSE, labtext = NULL)
cplab
```

Arguments

`comptab`	list or data frame, compositional differences generated by `get_comptab`
`vars`	character, which variables to plot
`col`	character or numeric, color(s) for the points
`plot.rect`	logical, plot a reference rectangle?
`pt.text`	character, text labels for the points
`cex.text`	numeric, size of text labels
`oldstyle`	logical, use old style plot?
`pch`	numeric, point symbol
`cex`	numeric, point size
`contour`	logical, add contour lines?
`col.contour`	character or numeric, color of contour lines
`probs`	numeric, probability level(s) for contours
`add`	logical, add to an existing plot?
`labtext`	character, text to add to axis labels

Details

A plot is created with points showing the differences between up- and down-regulated proteins for two compositional metrics, as calculated by `get_comptab`. The default setting of `vars` refers to average oxidation state of carbon (Z_C) as the x-variable and stoichiometric hydration state (n_{H_2O}) as the y-variable.

The colors of the points are controlled by `col`, which is recycled to be equal to the number of comparisons in `comptab`.

If `plot.rect` is TRUE, a shaded `rect`angle is drawn with coordinates -0.01, -0.01, 0.01, 0.01. This is useful for visualizing the different scales of multi-panel plots.

If pt.text is not NA or FALSE, text labels are added with size controlled by cex.text. The default value produces labels that are taken sequentially from the 26 lowercase Roman letters in alphabetical order (letters), followed by the set of uppercase letters (LETTERS).

For labtext = NULL, descriptive text ("median difference" or "mean difference") is added to the axis labels in parentheses. This text can be changed by giving a value in labtext, or NA to suppress the text.

cplab is a list of formatted labels used by diffplot. It is an exported object, available to the user and other packages.

Plot style

The overall style of the plot is controlled by oldstyle.

oldstyle = FALSE This is the current default style. Use pch and cex to control the point symbol and size. Contours are added for credible regions of highest probability density, computed using a 2-D kernel density estimate (kde2d). The color of contour lines is set by col.contour. contour can be a logical vector, indicating which points to include; set it to FALSE to omit the contour lines. probs specifies the probability levels; add more values to probs to get more contours.

oldstyle = TRUE This style is used for the historical (2017) vignettes in inst/oldvignettes. For each dataset, the point symbol is a filled square if the *p*-values of both the x-variable and y-variable are less than 0.05, a filled circle if the *p*-value of one of the x- or y-variables is less than 0.05, and an open circle otherwise. A solid line is drawn from the point to the corresponding axis if the rounded, absolute value of (CLES in percent - 50) of the x- or y-variable is greater than or equal 10. Otherwise, a dashed line is drawn from the point to the corresponding axis if the *p*-value of the x- or y-variable is less than 0.05. Otherwise, no line is drawn.

See Also

qdist to plot quantile distributions for a single dataset.

Examples

```
library(CHNOSZ)
# Make an old-style plot for two datasets
comptab <- lapply(c("JKMF10", "WDO+15_C.N"), function(dataset) {
  pdat <- pdat_colorectal(dataset)
  get_comptab(pdat, oldstyle = TRUE)
})
diffplot(comptab, oldstyle = TRUE)
```

Ehplot *Plot Eh*

Description

Show redox potential (Eh) scale.

Usage

```
Ehplot(basis = "QEC", T = 37, pH = 7.4)
```

Arguments

basis	character, keyword for basis species to use
T	numeric, temperature in degrees Celsius
pH	numeric, pH

Details

This function plots selected values of Eh (redox potential) as a function of $\log f_{O_2}$ and $\log a_{H_2O}$. The lines are labeled with the Eh value in volts. The temperature and pH can be adjusted using the T and pH arguments; this only affects the lines, but not the positions of labels.

References

Dick, J. M. (2016) Proteomic indicators of oxidation and hydration state in colorectal cancer. *PeerJ* **4**, e2238. doi: 10.7717/peerj.2238

See Also

This is used in `inst/oldvignettes/colorectal.Rmd` to reproduce Figure 6 of Dick (2016). Other conversions involving Eh are possible with `convert` (from **CHNOSZ**).

Examples

```
library(CHNOSZ)
Ehplot()
```

---

get_colors *Get Colors*

Description

Get colors for rank-difference (potential) diagrams.

Usage

```
get_colors(x, max50 = FALSE)
```

Arguments

x	numeric values
max50	logical, use most intense color for all values >= 50?

Details

`get_colors` returns a diverging (blue - light gray - red) color scale. Blue and red colors are associated with negative and positive values, respectively. The intensity of the color increases with the magnitude of the value. For accurate representation, the values should be in a percent scale (i.e. the maximum absolute value is not greater than 100). By default, a value of +/- 100 corresponds to greatest intensity. Set `max50` to TRUE to compress the scale so that greatest intesity is obtained at values of +/- 50 and higher.

See Also

These colors are used in `rankplot`. The colors were precomputed using **colorspace**::`diverge_hcl`.

`get_comptab` *Calculate Compositional Differences*

Description

Compute differences of carbon oxidation state, stoichiometric hydration state and other compositional metrics between groups of up- and down-regulated proteins.

Usage

```
get_comptab(pdat, var1 = "ZC", var2 = "nH2O", plot.it = FALSE,
  mfun = "median", oldstyle = FALSE)
```

Arguments

`pdat`	list, data object generated by a `pdat_` function
`var1`	character, the first variable
`var2`	character, the second variable
`plot.it`	logical, make a scatterplot?
`mfun`	character, either '`median`' or '`mean`'
`oldstyle`	logical, also calculate `CLES` and *p*-values?

Details

The available variables are:

'ZC'	average oxidation state of carbon (Z_C; see `ZCAA`)
'nH2O'	stoichiometric hydration state per residue (n_{H_2O}; see `H2OAA`)
'nC'	number of carbon atoms per residue
'nN'	number of nitrogen atoms per residue
'nS'	number of sulfur atoms per residue
'V0'	standard molal volume per residue
'nAA'	protein length (number of amino acids)
'GRAVY'	grand average of hydropathicity (see `GRAVY`)

'pI'	isoelectric point (see pI)
'PS_TPPG17'	phylostratum (see PS)
'PS_LMM16'	phylostratum (see PS)
'MW'	molecular weight per residue

Volume is calculated using amino acid group additivity as described by Dick et al. (2006).

Differentially expressed proteins are identified by the value of pdat$up2 (TRUE for up-regulated proteins and FALSE for down-regulated proteins). The differences are calculated as (median for up-regulated proteins) - (median for down-regulated proteins); if mfun is 'mean', means of the groups are used instead. If oldstyle is TRUE, the function also calculates the common language effect size (CLES, in percent) and *p*-value for each variable.

Phylostrata are not compositional metrics, but are retrieved by matching UniProt accession numbers in a data file (see PS). Because phylostratum numbers are discrete values, mean values are calculated regardless of the value of mfun.

Set plot.it to TRUE to make a scatterplot. Open red squares and filled blue circles stand for up-regulated and down-regulated proteins, respectively.

Value

A data frame is returned invisibly containing the columns 'dataset', 'description', 'n1' (number of down-regulated proteins), 'n2' (number of up-regulated proteins), followed two sets of columns for the variables. These are denoted generically as ('var.mfun1', 'var.mfun2', 'var.diff', 'var.CLES', 'var.p.value'), where 'var' is replaced by the name of var1 or var2, and 'mfun' is replaced by the value of mfun. For example, 'ZC.median1' and 'ZC.median2' are the median Z_C of the down- and up-regulated proteins, respectively.

References

Dick, J. M., LaRowe, D. E. and Helgeson, H. C. (2006) Temperature, pressure, and electrochemical constraints on protein speciation: Group additivity calculation of the standard molal thermodynamic properties of ionized unfolded proteins. *Biogeosciences* **3**, 311–336. https://doi.org/10.5194/bg-3-311-2006

Examples

```
library(CHNOSZ)
pd <- pdat_colorectal("JKMF10")
# default variables: ZC and nH2O
get_comptab(pd, plot.it = TRUE)
# protein length and per-residue volume
get_comptab(pd, "nAA", "V0", plot.it = TRUE)
```

groupplots *Plot Potential Diagrams for Groups of Datasets*

Description

Plot rank difference of chemical affinities for proteins in various datasets and merge the diagrams.

Usage

```
groupplots(group = "hypoxia_ZC_down", each100 = FALSE, res = 50, plot.it = TRUE)
mergedplot(gpresult, each100 = FALSE, res = 50)
```

Arguments

group	character, description of datasets to include
each100	logical, rescale rank difference of each dataset individually?
res	numeric, grid resolution for plots
plot.it	logical, make plots?
gpresult	list, value returned by groupplots

Details

groupplots makes weighted rank-difference of affinity (potential) diagrams (see rankplot) for each dataset found in the specified group. group consists of three parts joined by an underscore: the type of experiment ('colorectal', 'pancreatic', 'hypoxia', or 'osmotic'; see pdat_), the distinguishing compositional variable ('ZC' or 'H2O'), and the direction of change of that variable ('up' or 'down').

To identify the datasets in any group, compositional summaries for each dataset are read from pre-calculated tables in extdata/summary. Datasets are included for which the absolute mean difference of either 'ZC' or 'H2O' between up- and down-expressed proteins is greater than 0.01 and the other of 'ZC' or 'H2O' has p-value >= 0.05 and abs(CLES - 50) < 10.

groupplots makes calculations over a large range of $\log f_{O_2}$ and $\log a_{H_2O}$ in order to encompass the equipotential lines for most datasets. This way, the positions of the median and interquartiles of the equipotential lines can be calculated accurately for the mergedplot, which covers a smaller range of $\log f_{O_2}$ and $\log a_{H_2O}$.

See Also

This function is used in the historical vignette inst/oldvignettes/potential_diagrams.Rmd. rankplot and rankdiff are supporting functions.

Examples

```
## Not run:
gpresult <- groupplots("osmotic_H2O_down", res = 25)
mergedplot(gpresult, res = 25)

# reproduce Figure 3 of Dick, 2017
ZCgroups <- c("colorectal_ZC_up", "pancreatic_ZC_up", "hypoxia_ZC_down")
H2Ogroups <- c("colorectal_H2O_up", "pancreatic_H2O_up", "osmotic_H2O_down")
allgroups <- c(ZCgroups, H2Ogroups)
par(mfrow=c(2, 3))
for(group in allgroups) {
  gpresult <- groupplots(group, plot.it = FALSE)
  mergedplot(gpresult)
  title(main = group)
}
## End(Not run)
```

---

human *Amino Acid Compositions of Human Proteins*

Description

Data for amino acid compositions of proteins and conversion from old to new UniProt IDs.

Format

`human_aa` is a data frame with 25 columns in the format used for amino acid compositions in **CHNOSZ** (see `thermo`):

`protein`	character	Identification of protein
`organism`	character	Identification of organism
`ref`	character	Reference key for source of sequence data
`abbrv`	character	Abbreviation or other ID for protein (e.g. gene name)
`chains`	numeric	Number of polypeptide chains in the protein
`Ala...Tyr`	numeric	Number of each amino acid in the protein

The `protein` column contains UniProt IDs in the format `database|accession-isoform`, where `database` is most often '`sp`' (Swiss-Prot) or '`tr`' (TrEMBL), and `isoform` is an optional suffix indicating the isoform of the protein (particularly in the `human_additional` file).

Details

The amino acid compositions of human proteins are stored in three files under `extdata/protein`.

- `human_base.rds` contains amino acid compositions of canonical isoforms of manually reviewed proteins in the UniProt reference human proteome (computed from sequences in `UP000005640_9606.fasta.gz`, dated 2016-04-03).

- human_additional.rds contains amino acid compositions of additional proteins (UP000005640_9606_additional.fasta.gz) including isoforms and unreviewed sequences. In version 0.1.5, this file was trimmed to include only those proteins that are used in any of the datasets in the package.
- human_extra.csv contains amino acid compositions of other ("extra") proteins used in a dataset but not listed in one of the files above. These proteins may include obsolete, unreviewed, or newer additions to the UniProt database. Not all sequences here are strictly HUMAN (see the organism column and the ref column for the reference keys).

On loading the package, the individual data files are read and combined, and the result is assigned to the human_aa object in the human environment.

As an aid for processing datasets that list old (obsolete) UniProt IDs, the corresponding new (current) IDs are are stored in uniprot_updates. These ID mappings have been manually added as needed for individual datasets, and include proteins from humans as well as other organisms. check_IDs performs the conversion of old to new IDs.

See Also

Amino acid compositions of non-human proteins are stored in directories archaea, bacteria, cow, dog, mouse, rat, and yeast under extdata/aa. These files can be loaded in protcomp via the aa_file argument, which is used e.g. in pdat_osmotic_bact.

Examples

```
# The number of proteins
nrow(get("human_aa", human))
# The number of old to new ID mappings
nrow(get("uniprot_updates", human))
```

---

metrics *Calculate Compositional Metrics for Proteins*

Description

These functions calculate compositional metrics of proteins given a data frame of amino acid compositions.

Usage

```
ZCAA(AAcomp, nothing = NULL)
H2OAA(AAcomp, basis = "rQEC")
GRAVY(AAcomp)
pI(AAcomp)
MWAA(AAcomp)
```

Arguments

AAcomp	data frame, amino acid compositions
nothing	dummy argument
basis	character, basis species

Details

Columns in AAcomp should be named with the three-letter abbreviations for the amino acids (e.g. 'Ala'). The metrics are described below:

ZCAA Average oxidation state of carbon (Z_C) (Dick, 2014). nothing is an extra argument that does nothing. It is provided so that scripts can be written with do.call to run ZCAA and H2OAA with the same number of arguments.

H2OAA Stoichiometric hydration state (n_{H_2O}) per residue. The default for basis stands for stoichiometric hydration state as described in Dick et al. (2020). Briefly, the values are obtained from the residuals of a linear model for n_{H_2O} vs Z_C of the 20 common amino acids using the basis species glutamine - glutamic acid - cysteine - H_2O - O_2. A constant of 0.355 is subtracted from these residuals to make the mean for all human proteins = 0. basis can be changed to 'QEC' to use the stoichiometric water content calculated directly from the basis species (as in Dick, 2017).

GRAVY Grand average of hydropathicity. Values of the hydropathy index for individual amino acids are from Kyte and Doolittle (1982).

pI Isoelectric point. The net charge for each ionizable group was pre-calculated from pH 0 to 14 at intervals of 0.01. The isoelectric point is found as the pH where the sum of charges of all groups in the protein is closest to zero. The p*K* values for the terminal groups and sidechains are taken from Bjellqvist et al. (1993) and Bjellqvist et al. (1994); note that the calculation does not implement position-specific adjustments described in the latter paper. The number of N- and C-terminal groups is taken to be one, unless a value for chains (number of polypeptide chains) is given in AAcomp.

MWAA Molecular weight per residue.

Note that Z_C is a per-carbon average, but n_{H_2O} is a per-residue average. The contribution of H_2O from the terminal groups of proteins is counted, so shorter proteins have slightly greater n_{H_2O}.

Tests for a few proteins (see examples) indicate that GRAVY and pI are equal those calculated with the ProtParam tool (https://web.expasy.org/protparam/; Gasteiger et al., 2005).

References

Bjellqvist, B., Hughes, G. J., Pasquali, C., Paquet, N., Ravier, F., Sanchez, J.-C., Frutiger, S. and Hochstrasser, D. (1993) The focusing positions of polypeptides in immobilized pH gradients can be predicted from their amino acid sequences. *Electrophoresis* **14**, 1023–1031. doi: 10.1002/elps.11501401163

Bjellqvist, B. and Basse, B. and Olsen, E. and Celis, J. E. (1994) Reference points for comparisons of two-dimensional maps of proteins from different human cell types defined in a pH scale where isoelectric points correlate with polypeptide compositions. *Electrophoresis* **15**, 529–539. doi: 10.1002/elps.1150150171

Dick, J. M. (2014) Average oxidation state of carbon in proteins. *J. R. Soc. Interface* **11**, 20131095. doi: 10.1098/rsif.2013.1095

Dick, J. M. (2017) Chemical composition and the potential for proteomic transformation in cancer, hypoxia, and hyperosmotic stress. *PeerJ* **5**, e3421. doi: 10.7717/peerj.3421

Dick, J. M., Yu, M. and Tan, J. (2020) Distinct trends in chemical composition of proteins from metagenomes in redox and salinity gradients. *bioRxiv*. doi: 10.1101/2020.04.01.020008

Gasteiger, E., Hoogland, C., Gattiker, A., Duvaud, S., Wilkins, M. R., Appel, R. D. and Bairoch, A. (2005) Protein identification and analysis tools on the ExPASy server. In J. M. Walker (Ed.), *The Proteomics Protocols Handbook* (pp. 571–607). Totowa, NJ: Humana Press Inc. doi: 10.1385/1592598900:571

Kyte, J. and Doolittle, R. F. (1982) A simple method for displaying the hydropathic character of a protein. *J. Mol. Biol.* **157**, 105–132. doi: 10.1016/00222836(82)905150

See Also

Use PS to get phylostrata of proteins (based on matching IDs in a data file, not amino acid composition). For calculations of Z_C from chemical formulas of organic molecules (not only proteins), use the ZC function in **CHNOSZ**.

Examples

```
# we need CHNOSZ for these examples
require(CHNOSZ)

# for reference, compute ZC of alanine and glycine "by hand"
ZC.Gly <- ZC("C2H5NO2")
ZC.Ala <- ZC("C3H7NO2")
# define the composition of a Gly-Ala-Gly tripeptide
AAcomp <- data.frame(Gly = 2, Ala = 1)
# calculate the ZC of the tripeptide (value: 0.571)
ZC.GAG <- ZCAA(AAcomp)
# this is equal to the carbon-number-weighted average of the amino acids
nC.Gly <- 2 * 2
nC.Ala <- 1 * 3
ZC.average <- (nC.Gly * ZC.Gly + nC.Ala * ZC.Ala) / (nC.Ala + nC.Gly)
stopifnot(all.equal(ZC.GAG, ZC.average))

# compute the per-residue nH2O of Gly-Ala-Gly
basis("QEC")
nH2O.GAG <- species("Gly-Ala-Gly")$H2O
# divide by the length to get residue average (we keep the terminal H-OH)
nH2O.residue <- nH2O.GAG / 3
# compare with the value calculated by H2OAA() (-0.2)
nH2O.H2OAA <- H2OAA(AAcomp, "QEC")
stopifnot(all.equal(nH2O.residue, nH2O.H2OAA))

# calculate GRAVY for lysozyme and ribonuclease
# first get the protein index in CHNOSZ's list of proteins
iprotein <- pinfo(c("LYSC_CHICK", "RNAS1_BOVIN"))
# then get the amino acid compositions
```

```
AAcomp <- pinfo(iprotein)
# then calculate GRAVY
Gcalc <- as.numeric(GRAVY(AAcomp))
# these are equal to values obtained with ProtParam on uniprot.org
Gref <- c(-0.472, -0.663)
stopifnot(all.equal(round(Gcalc, 3), Gref))
# also calculate molecular weight of the proteins
MWcalc <- as.numeric(MWAA(AAcomp)) * protein.length(iprotein)
# https://web.expasy.org/cgi-bin/protparam/protparam1?P00698@19-147@
# https://web.expasy.org/cgi-bin/protparam/protparam1?P61823@27-150@
MWref <- c(14313.14, 13690.29)
stopifnot(all.equal(round(MWcalc, 2), MWref))

# calculate pI for a few proteins
iprotein <- pinfo(c("CSG_HALJP", "AMYA_PYRFU", "RNAS1_BOVIN", "LYSC_CHICK"))
AAcomp <- pinfo(iprotein)
pI_calc <- pI(AAcomp)
# reference values calculated with ProtParam on uniprot.org
# CSG_HALJP: residues 35-862 (sequence v1)
# AMYA_PYRFU: residues 2-649 (sequence v2)
# RNAS1_BOVIN: residues 27-150 (sequence v1)
# LYSC_CHICK: residues 19-147 (sequence v1)
pI_ref <- c(3.37, 5.46, 8.64, 9.32)
stopifnot(all.equal(as.numeric(pI_calc), pI_ref))
```

`mkvig` *Compile and View Vignettes from the Command Line*

Description

This function compiles the indicated vignette and opens it in the browser.

Usage

```
mkvig(vig = NULL)
```

Arguments

`vig`	character, name of a vignette without '`.Rmd`' extension

Details

Starting with version 0.2, in order to reduce package space and check time, pre-built vignettes are not included in the package. This function was added to compile the vignettes on demand and view them in a browser.

The available vignettes are listed here:

- *Cell culture* – '`hypoxia`', '`secreted`', '`osmotic_bact`', '`osmotic_euk`', '`osmotic_halo`', '`glucose`', '`3D`'

- *Cancer* – 'breast', 'colorectal', 'liver', 'lung', 'pancreatic', 'prostate'

Note that pandoc (including pandoc-citeproc), as a system dependency of **rmarkdown**, must be available. See **rmarkdown**'s 'pandoc' vignette (https://CRAN.R-project.org/package=rmarkdown/vignettes/pandoc.html) for installation tips.

See Also

The vignettes can also be run using e.g. `demo("glucose")`, and through the interactive help system (`help.start` > Packages > canprot > Code demos).

Examples

```
## Not run:
mkvig("colorectal")

## End(Not run)
```

`pdat_` *Get Protein Expression Data*

Description

Get data for protein expression and chemical composition.

Usage

```
pdat_breast(dataset = 2020, basis = "rQEC")
pdat_colorectal(dataset = 2020, basis = "rQEC")
pdat_liver(dataset = 2020, basis = "rQEC")
pdat_lung(dataset = 2020, basis = "rQEC")
pdat_pancreatic(dataset = 2020, basis = "rQEC")
pdat_prostate(dataset = 2020, basis = "rQEC")
pdat_hypoxia(dataset = 2020, basis = "rQEC")
pdat_secreted(dataset = 2020, basis = "rQEC")
pdat_3D(dataset = 2020, basis = "rQEC")
pdat_glucose(dataset = 2020, basis = "rQEC")
pdat_osmotic_bact(dataset = 2020, basis = "rQEC")
pdat_osmotic_euk(dataset = 2020, basis = "rQEC")
pdat_osmotic_halo(dataset = 2020, basis = "rQEC")
.pdat_multi(dataset = 2020, basis = "rQEC")
.pdat_osmotic(dataset = 2017, basis = "rQEC")
```

Arguments

`dataset`	character, dataset name
`basis`	character, keyword for basis species to use

Details

The pdat_ functions assemble lists of up- and down-regulated proteins and calculate chemical compositions using protcomp. After this, use get_comptab to make a table of compositional metrics that can be plotted with diffplot.

If dataset is '2020' (the default) or '2017', the function returns the names of all datasets in the compilation for the respective year.

Each dataset name starts with a reference key indicating the study (publication) where the data were reported. The reference keys are made by combining the first characters of the authors' family names with the 2-digit year of publication. For mutiple datasets from one study, the reference key is followed by an underscore and descriptive text for the particular dataset.

Provide one of the dataset names as the dataset argument to retrieve the data. The functions get protein expression data from the CSV files stored in extdata/expression/, under the subdirectory corresponding to the name of the pdat_ function. Some of the functions also read amino acid compositions (for non-human proteins) from the files in extdata/aa/.

Descriptions for each function:

- pdat_colorectal, pdat_pancreatic, pdat_breast, pdat_lung, pdat_prostate, and pdat_liver retrieve data for protein expression in different cancer types.
- pdat_hypoxia gets data for cellular extracts in hypoxia and pdat_secreted gets data for secreted proteins (e.g. exosomes) in hypoxia.
- pdat_3D retrieves data for 3D (e.g. tumor spheroids and aggregates) compared to 2D (monolayer) cell culture.
- .pdat_osmotic retrieves data for hyperosmotic stress, for the 2017 compilation only. In 2020, this compilation was expanded and split into pdat_osmotic_bact (bacteria), pdat_osmotic_euk (eukaryotic cells) and pdat_osmotic_halo (halophilic bacteria and archaea).
- pdat_glucose gets data for high-glucose experiments in eukaryotic cells.
- .pdat_multi retrieves data for studies that have multiple types of datasets (e.g. both cellular and secreted proteins in hypoxia), and is used internally by the specific functions (e.g. pdat_hypoxia and pdat_secreted).

Value

A list consisting of:

dataset the name of the dataset

basis basis species used for the calculations

description descriptive text for the dataset

pcomp compositional data generated by protcomp

up2 logical vector with length equal to the number of proteins; TRUE for up-regulated protein and FALSE for down-regulated proteins

Examples

```
library(CHNOSZ)
# list datasets in the 2017 complilation for colorectal cancer
pdat_colorectal(2017)
# process one dataset
pdat_colorectal("JKMF10")
```

---

protcomp *Protein Compositions*

Description

Get amino acid and chemical compositions of proteins.

Usage

```
protcomp(uniprot = NULL, basis = "rQEC",
         aa = NULL, aa_file = NULL)
```

Arguments

uniprot	character, UniProt IDs of proteins
basis	character, keyword for basis species to use
aa	data frame, amino acid compositions
aa_file	character, file name

Details

This function retrieves the amino acid compositions of one or more proteins specified by `uniprot` or `ip`, then calculates some chemical compositional properties using functions provided by **CHNOSZ**. The `basis` argument is used to select the basis species using a keyword (see `basis`). The default is '`rQEC`' where the number of H_2O is residual-corrected stoichiometric water content of amino acids (see `metrics`). Other options are '`CHNOS`' for CO_2, NH_3, H_2S, H_2O, and O_2, or '`QEC`' for glutamine, glutamic acid, cysteine, H_2O, and O_2.

This function depends on the amino acid compositions of human proteins, which are stored in the `human` environment when the package is attached. If `aa_file` is specified, additional amino acid compositions to be considered are read from this file, which should be in the same format as e.g. `human_extra.csv` (see also `thermo$protein`). Alternatively, the amino acid compositions can be given in `aa`, bypassing the search step.

Value

The function returns a list with elements `uniprot` (UniProt IDs as given in the arguments), `protein.formula` (elemental compositions of the proteins), `ZC` (average oxidation state of carbon), `protein.basis` (compositions of the proteins in terms of the basis species), `protein.length` (lengths of the amino acid sequences), `residue.basis` (per-residue compositions of the proteins in terms of the basis species), `residue.formula` (per-residue elemental compositions of the proteins), and `aa` (amino acid compositions of the proteins).

See Also

cleanup

Examples

```
library(CHNOSZ)
protcomp("P24298")
```

PS *Retrieve Phylostrata for Given UniProt IDs*

Description

Retrieves the phylostrata for protein-coding genes according to Liebeskind et al. (2016) or Trigos et al. (2017).

Usage

```
PS(uniprot, source = "TPPG17")
```

Arguments

uniprot	character, UniProt accession numbers
source	character, 'TPPG17' or 'LMM16'

Details

The phylostratum for each protein is found by matching the UniProt ID in one of these data files:

extdata/phylostrata/TPPG17.csv.xz This file has columns 'GeneID' (gene name), 'Entrez', 'Entry', and 'Phylostrata'. Except for 'Entry', the values are from Dataset S1 of Trigos et al. (2017). UniProt acession numbers in 'Entry' were generated using the UniProt mapping tool first for 'Entrez', followed by 'GeneID' for the unmatched genes. 'Entry' is NA for genes that remain unmatched to any proteins after both mapping steps.

extdata/phylostrata/LMM16.csv.xz This file has columns 'UniProt', 'modeAge', and 'PS'. The data are from file main_HUMAN.csv in Gene-Ages v1.0 (https://zenodo.org/record/51708; Liebeskind et al. (2016)). The modeAges were converted to phylostrata values 1-8 ('PS' column) in this order: Cellular_organisms, Euk_Archaea, Euk+Bac, Eukaryota, Opisthokonta, Eumetazoa, Vertebrata, Mammalia.

References

Trigos, A. S. and Pearson, R. B. and Papenfuss, A. T. and Goode, D. L. (2017) Altered interactions between unicellular and multicellular genes drive hallmarks of transformation in a diverse range of solid tumors. *Proc. Natl. Acad. Sci.* **114**, 6406–6411. doi: 10.1073/pnas.1617743114

Liebeskind, B. J. and McWhite, Claire J. and Marcotte, E. M. (2016) Towards consensus gene ages. *Genome Biol. Evol.* **8**, 1812–1823. doi: 10.1093/gbe/evw113

See Also

Call get_comptab with 'PS_TPPG17' or 'PS_LMM16' as a variable name to calculate mean differences of phylostrata for differentially expressed proteins.

Examples

```
# Get protein expression data for one dataset
pd <- pdat_colorectal("JKMF10")
opar <- par(mfrow = c(2, 1))
# Plot nH2O vs phylostrata with Trigos et al. data
get_comptab(pd, "PS_TPPG17", plot.it = TRUE)
# Plot nH2O vs phylostrata with Liebeskind et al. data
get_comptab(pd, "PS_LMM16", plot.it = TRUE)
par(opar)

# compare the two sources
PSdir <- system.file("extdata/phylostrata", package = "canprot")
TPPG17 <- read.csv(file.path(PSdir, "TPPG17.csv.xz"))
LMM16 <- read.csv(file.path(PSdir, "LMM16.csv.xz"))
IDs <- intersect(TPPG17$Entry, LMM16$UniProt)
PS_TPPG17 <- TPPG17$Phylostrata[match(IDs, TPPG17$Entry)]
PS_LMM16 <- LMM16$PS[match(IDs, LMM16$UniProt)]
plot(jitter(PS_TPPG17), jitter(PS_LMM16), pch = ".")
```

---

qdist *Quantile Distributions for One Dataset*

Description

Make a plot showing quantile distributions for up- and down-regulated proteins.

Usage

```
qdist(pdat, vars = c("ZC", "nH2O"), show.steps = FALSE)
```

Arguments

pdat	list, output of a pdat_ function for a single dataset
vars	character, which variables to plot
show.steps	logical, show the steps using plot.ecdf?

Details

This function makes a quantile distribution plot with lines for both up- and down-regulated proteins. The variable (var) can be 'ZC', 'H2O', or both (two plots are made for the latter). The horizontal axis is the variable and the vertical axis is the quantile point. A solid black line is drawn for the down-regulated proteins, and a dashed red line for the up-regulated proteins. The median difference is shown by a gray horizontal line drawn between the distributions at the 0.5 quantile point.

References

Jimenez, C. R. and Knol, J. C. and Meijer, G. A. and Fijneman, R. J. A. (2010) Proteomics of colorectal cancer: Overview of discovery studies and identification of commonly identified cancer-associated proteins and candidate CRC serum markers. *J. Proteomics* **73**, 1873–1895. doi: 10.1016/j.jprot.2010.06.004

See Also

`diffplot` to plot median differences for multiple datasets.

Examples

```
# Plot the data of Jimenez et al., 2020 for colorectal cancer
pdat <- pdat_colorectal("JKMF10")
qdist()
```

---

rankdiff *Weighted Difference of Sums of Ranks*

Description

Calculate rank-sum difference between two groups, weighted by the sizes of the groups.

Usage

```
rankdiff(rank1, rank2, n1 = NULL, n2 = NULL, as.fraction=TRUE)
```

Arguments

`rank1`	numeric, ranks in group 1
`rank2`	numeric, ranks in group 2
`n1`	numeric, size of group 1
`n2`	numeric, size of group 2
`as.fraction`	logical, calculate the fraction of maximum possible difference?

Details

In a combined ranking of two groups, the comparison of sum of ranks has an easy interpretation only for groups of equal size. The weighted rank difference is used to compare groups of unequal size. The weighting ensures that 1) opposite extreme configurations give weighted rank differences with equal magnitudes, and 2) an evenly distributed (interspersed) ranking of the two groups has a weighted rank difference of zero (Dick, 2016).

If `n1` and `n2` are not given, `rank1` and `rank2` are interpreted as vectors holding the ranks for the two groups. If the sizes of the groups are supplied in `n1` and `n2`, then the single values or higher-dimensional objects in `rank1` and `rank2` are interpreted as the non-weighted sums of ranks of the two groups.

References

Dick, J. M. (2016) Proteomic indicators of oxidation and hydration state in colorectal cancer. *PeerJ* **4**, e2238. doi: 10.7717/peerj.2238

See Also

This function is used in `groupplots`.

Examples

```
# rankings of H and C in H-H-H-H-C-C-C
rankdiff(1:4, 5:7, as.fraction=FALSE)  # 12
rankdiff(1:4, 5:7)  # 1

# rankings of H and C in C-C-C-H-H-H-H
rankdiff(4:7, 1:3, as.fraction=FALSE)  # -12
rankdiff(4:7, 1:3)  # -1

# rankings of H and C in H-C-H-C-H-C-H
rankdiff(c(1, 3, 5, 7), c(2, 4, 6))  # 0
```

`rankplot` *Plot Ranking of Chemical Affinities*

Description

Plot ranking of chemical affinities of groups of proteins.

Usage

```
rankplot(pdat, T = 37, what = "rankdiff", main = NULL, res = 300,
         plot.it = TRUE, xlim = c(-75, -55), ylim = c(-10, 10))
```

Arguments

`pdat`	list, data object generated by a `pdat_` function
`T`	numeric, temperature in degrees Celsius
`what`	character, "rankdiff" or "affinity"
`main`	character, text to use for title of plot
`res`	numeric, grid resolution for plot
`plot.it`	logical, draw a plot?
`xlim`	numeric, range of x-axis ($\log f_{O_2}$)
`ylim`	numeric, range of y-axis ($\log a_{H_2O}$)

Details

This function creates a $\log a_{H_2O}$ - $\log f_{O_2}$ diagram showing the relative stabilities of the two groups of proteins in the specified `dataset`. These groups consist of the relatively down- and up-expressed proteins identified by `up2` in one of the `pdat_` functions.

The function generates a colored `image` and `contour` plot showing the weighted difference of sums of ranks of chemical affinities of formation of proteins in the two groups (see `affinity` and `rankdiff`). Increasing intensity of blue or red colors (see `get_colors`) represent higher rankings of the down-expressed (`up2==FALSE`) or up-expressed (`up2==TRUE`) proteins, respectively. Alternatively, if `what` is "affinity", a maximum affinity diagram is produced (see `diagram` in CHNOSZ), with fields colored red or blue according to the relative expression of the protein.

If `main` is NULL, the title for the plot is taken from the description supplied in `pdat`.

Set `plot.it` to FALSE to skip the plotting and instead return a list containing the computed rank differences and x- and y- values and labels.

See Also

This function is used in `groupplots`.

Examples

```
library(CHNOSZ)
pdat <- pdat_colorectal("JKMF10")
rankplot(pdat, res=25)
rankplot(pdat, res=25, what="affinity")
```

---

recomp *Recompute Protein Compositions*

Description

Recompute compositional metrics given the existing output of one of the `pdat_` functions.

Usage

```
recomp(pdat, basis = "rQEC")
```

Arguments

`pdat`	list, result from a `pdat_` function
`basis`	character, keyword for basis species to use

Details

This function recomputes a pdat object for a different '`basis`'. This avoids the need to regenerate the list of UniProt IDs and search for amino acid compositions. It was previously used to speed up the processing of the vignettes.

Examples

```
# process the first dataset for prostate cancer using the QEC basis species
datasets <- pdat_prostate()
pdat <- pdat_prostate(datasets[1], basis = "QEC")
# recompute the compositions using the rQEC values
pdat2 <- recomp(pdat, "rQEC")
# compare the results
nH2O_QEC <- pdat$pcomp$residue.basis[, "H2O"]
nH2O_rQEC <- pdat2$pcomp$residue.basis[, "H2O"]
plot(nH2O_QEC, nH2O_rQEC)
# the slower way (not that it makes much difference for this example)
pdat2.slow <- pdat_prostate(datasets[1], basis = "rQEC")
stopifnot(identical(pdat2, pdat2.slow))
```

`xsummary` *Summarize Compositional Differences*

Description

Make an HTML table summarizing compositional differences.

Usage

```
xsummary(comptab, vars=c("ZC", "nH2O"))
xsummary2(comptab1, comptab2, comptab3)
xsummary3(comptab1, comptab2, comptab3)
```

Arguments

`comptab`	list or data frame, summary of comparisons generated by `get_comptab`
`vars`	character, two variables to tabulate
`comptab1`	list, output of `get_comptab`
`comptab2`	list, output of `get_comptab`
`comptab3`	list, output of `get_comptab`

Details

`xsummary` makes an HTML table (using `xtable`) and adds bold and underline formatting to highlight significant compositional differences. The *p*-value is bolded if it is less than 0.05, and the percent common language effect size (`CLES`) is bolded if it is <= 40 or >= 60. The mean (or median) difference is [underlined / bolded] if [only one of / both] the *p*-value and CLES pass these cutoffs.

The generated table is written to the console, and can be used in a vignette using the `results = "asis"` chunk option. The function also returns (invisibly) the data frame used to make the table; this data frame differs from `comptab` by having row names added (alphabetical one-letter IDs for the datasets).

`xsummary2` is an updated version that is used in the current vignettes in the package. It shows negative numbers in bold (*p*-value and CLES are not shown). `xsummary3` is a further revision that shows GRAVY and pI instead of phylostrata; it is used in the 'osmotic_' vignettes.

Examples

```
library(CHNOSZ)
comptab <- lapply(c("JKMF10", "WDO+15_C.N"), function(dataset) {
  pdat <- pdat_colorectal(dataset)
  get_comptab(pdat, oldstyle = TRUE)
})
xsummary(comptab)
```

Index