Package ‘idpr’

May 3, 2024

Type Package
Title Profiling and Analyzing Intrinsically Disordered Proteins in R
Version 1.14.0
Date 2024-04-23
Description ‘idpr’ aims to integrate tools for the computational analysis of intrinsically disordered proteins (IDPs) within R. This package is used to identify known characteristics of IDPs for a sequence of interest with easily reported and dynamic results. Additionally, this package includes tools for IDP-based sequence analysis to be used in conjunction with other R packages. Described in McFadden WM & Yanowitz JL (2022). ``idpr: A package for profiling and analyzing Intrinsically Disordered Proteins in R." PloS one, 17(4), e0266929. <https://doi.org/10.1371/journal.pone.0266929>.

BugReports https://github.com/wmm27/idpr/issues
License LGPL (>= 3)
Encoding UTF-8
LazyData true
biocViews StructuralPrediction, Proteomics, CellBiology
RoxygenNote 7.2.3
Depends R (>= 4.1.0)
Imports ggplot2 (>= 3.3.0), magrittr (>= 1.5), dplyr (>= 0.8.5), plyr (>= 1.8.6), jsonlite (>= 1.6.1), rlang (>= 0.4.6), Biostrings (>= 2.56.0), methods (>= 4.0.0)
Suggests knitr, rmarkdown, pwalign, msa, ape, testthat, seqinr
VignetteBuilder knitr
git_url https://git.bioconductor.org/packages/idpr
git_branch RELEASE_3_19
git_last_commit a78b77b
git_last_commit_date 2024-04-30
Repository Bioconductor 3.19
Date/Publication 2024-05-02
chargeCalculationGlobal

Protein Charge Calculation, Globally

Description

This function will determine the charge of a peptide using the Henderson-Hasselbalch Equation. The output is a data frame (default) or a plot of charge calculations along the peptide sequence. Charges are determined globally, or along the entire chain.
Usage

chargeCalculationGlobal(
  sequence,
  pKaSet = "IPC_protein",
  pH = 7,
  plotResults = FALSE,
  includeTermini = TRUE,
  sumTermini = TRUE,
  proteinName = NA,
  printCitation = FALSE,
  ...)
)

Arguments

sequence amino acid sequence as a character string or vector of individual residues. alternatively, a character string of the path to a .fasta/.fa file

pKaSet A character string or data frame. "IPC_protein" by default. Character string to load specific, preloaded pKa sets. c("EMBOSS", "DTASelect", "Solomons", "Sillerio", "Rodwell", "Lehninger", "Toseland", "Thurlkill", "Nozaki", "Dawson", "Bjellqvist", "ProMoST", "Vollhardt", "IPC_protein", "IPC_peptide") Alternatively, the user may supply a custom pKa dataset. The format must be a data frame where: Column 1 must be a character vector of residues named "AA" AND Column 2 must be a numeric vector of pKa values.

pH numeric value, 7.0 by default. The environmental pH used to calculate residue charge.

plotResults logical value, FALSE by default. This determines what is returned. If plotResults = FALSE, a data frame is returned with the position, residue, and charge (-1 to +1). If plotResults = TRUE, a graphical output is returned (ggplot) showing the charge distribution.

includeTermini, sumTermini Logical values, both TRUE by default. This determines how the calculation handles the N- and C- terminus. includeTermini determines if the calculation will use the charge of the amine and carboxyl groups at the ends of the peptide (When TRUE). These charges are ignored when includeTermini = FALSE. sumTermini determines if the charge of the first (likely Met, therefore uncharged), and final residue (varies) will be added to the termini charges, or if the N and C terminus will be returned as separate residues. When sumTermini = TRUE, charges are summed. When sumTermini = FALSE, the N and C terminus are added as a unique residue in the DF. This will impact averages by increasing the sequence length by 2. sumTermini is ignored if includeTermini = FALSE.

proteinName character string with length = 1. optional setting to include the name in the plot title.

printCitation Logical value. FALSE by default. When printCitation = TRUE the citation for the pKa set is printed. This allows for the user to easily obtain the dataset citation. Will not print if there is a custom dataset.

... any additional parameters, especially those for plotting.
Value

If `plotResults = FALSE`, a data frame is returned with the position, residue, and charge (-1 to +1). If `plotResults = TRUE`, a graphical output is returned (ggplot) showing the charge distribution.

Plot Colors

For users who wish to keep a common aesthetic, the following colors are used when `plotResults = TRUE`:

- Dynamic line colors:
  - Close to -1 = "#92140C"
  - Close to +1 = "#348AA7"
  - Close to 0 (midpoint) = "grey65" or "#A6A6A6"

See Also

`pKaData` for residue pKa values and `hendersonHasselbalch` for charge calculations.

Other charge functions: `chargeCalculationLocal()`, `hendersonHasselbalch()`, `netCharge()`

Examples

```r
#Amino acid sequences can be character strings
aaString <- "ACDEFGHIKLMNPQRSTVWY"
#Amino acid sequences can also be character vectors
aaVector <- c("A", "C", "D", "E", "F",
              "G", "H", "I", "K", "L",
              "M", "N", "P", "Q", "R",
              "S", "T", "V", "W", "Y")
#Alternatively, .fasta files can also be used by providing
#a character string of the path to the file.
exampleDF <- chargeCalculationGlobal(aaString)
head(exampleDF)
exampleDF <- chargeCalculationGlobal(aaVector)
head(exampleDF)

#Changing pKa set or pH used for calculations
exampleDF_pH5 <- chargeCalculationGlobal(aaString,
pH = 5)
head(exampleDF_pH5)
exampleDF_pH7 <- chargeCalculationGlobal(aaString,
pH = 7)
head(exampleDF_pH7)
exampleDF_EMBOSS <- chargeCalculationGlobal(aaString,
pH = 7,
   pKa = "EMBOSS")
head(exampleDF_EMBOSS)

#If the termini charge should not be included with includeTermini = F
```
chargeCalculationLocal

Charge Calculation Along a Protein Sequence

Description

This calculates the charge, as determined by the Henderson-Hasselbalch equation, for each window along the sequence. This function uses a sliding window. The output is either a graph or a data frame of calculated charges.

Usage

chargeCalculationLocal(
  sequence,  # protein sequence
  window = 9,  # window size
  proteinName = NA,  # protein name
  pH = 7,  # pH of the solution
  pKaSet = "IPC_protein",  # pKa set
  printCitation = FALSE,  # print citation
  plotResults = FALSE,  # plot results
  includeTermini = FALSE  # include termini
)
chargeCalculationLocal

Arguments

sequence  amino acid sequence as a single character string or vector of single characters. It also supports a single character string that specifies the location of a .fasta or .fa file.

window  a positive, odd integer. 9 by default. Sets the size of sliding window, must be an odd number. The window determines the number of residues to be analyzed and averaged for each position along the sequence.

proteinName  character string, optional. Used to add protein name to the title in ggplot. Ignored if plotResults = FALSE.

pH  numeric value, 7.0 by default. The environmental pH used to calculate residue charge.

pKaSet  A character string or data frame. "IPC_protein" by default. Character string to load specific, preloaded pKa sets. c("EMBOSS", "DTASelect", "Solomons", "Sillero", "Rodwell", "Lehninger", "Toseland", "Thurilikil", "Nozaki", "Dawson", "Bjellqvist", "ProMoST", "Vollhardt", "IPC_protein", "IPC_peptide") Alternatively, the user may supply a custom pKa dataset. The format must be a data frame where: Column 1 must be a character vector of residues named "AA" AND Column 2 must be a numeric vector of pKa values.

printCitation  Logical value. FALSE by default. When printCitation = TRUE the citation for the pKa set is printed. This allows for the user to easily obtain the dataset citation. Will not print if there is a custom dataset.

plotResults  logical value. TRUE by default. If plotResults = TRUE, a ggplot of window charges are returned. If plotResults = FALSE, a data frame of window charges are returned.

Value

see plotResults argument

Plot Colors

For users who wish to keep a common aesthetic, the following colors are used when plotResults = TRUE.

- Dynamic line colors:
  - Close to -1 = "#92140C"
  - Close to +1 = "#348AA7"
  - Close to 0 (midpoint) = "grey65" or "#A6A6A6"
See Also

pKaData for residue pKa values and citations. See hendersonHasselbalch for charge calculations.
Other charge functions: chargeCalculationGlobal(), hendersonHasselbalch(), netCharge()

Examples

# Amino acid sequences can be character strings
aaString <- "ACDEFGHIKLMNPQRSTVWY"
# Amino acid sequences can also be character vectors
aaVector <- c("A", "C", "D", "E", "F",
              "G", "H", "I", "K", "L",
              "M", "N", "P", "Q", "R",
              "S", "T", "V", "W", "Y")
# Alternatively,.fasta files can also be used by providing
# a character string of the path to the file.
exampleDF <- chargeCalculationLocal(aaString)
exampleDF <- chargeCalculationLocal(aaVector)
head(exampleDF)

# Changing window will alter the number of residues analyzed
exampleDF_window3 <- chargeCalculationLocal(aaString,
                                          window = 3)
exampleDF_window15 <- chargeCalculationLocal(aaString,
                                         window = 15)
head(exampleDF_window15)

# Changing pKa set or pH used for calculations
exampleDF_pH5 <- chargeCalculationLocal(aaString,
                               pH = 5)
head(exampleDF_pH5)
exampleDF_pH7 <- chargeCalculationLocal(aaString,
                               pH = 7)
head(exampleDF_pH7)
exampleDF_EMBOSS <- chargeCalculationLocal(aaString,
                                pH = 7,
                                pKa = "EMBOSS")
head(exampleDF_EMBOSS)

# plotResults = TRUE will output a ggplot
chargeCalculationLocal(aaString,
                      plot = TRUE)

# since it is a ggplot, you can change or annotate the plot
gg <- chargeCalculationLocal(aaVector,
                             window = 3,
                             plot = TRUE)

gg <- gg + ggplot2::ylab("Local Charge")
gg <- gg + ggplot2::geom_text(data = exampleDF_window3,
                                ggplot2::aes(label = CenterResidue,
                                             y = windowCharge + 0.1))
plot(gg)
chargeHydropathyPlot  Charge-Hydropathy Plot

**Description**

This function calculates the average net charge \( <R> \) and the average scaled hydropathy \( <H> \) and visualizes the data. There are known boundaries on the C-H plot that separate extended and collapsed proteins.

This was originally described in Uversky et al. (2000) [https://doi.org/10.1021/1097-0134(20001115)41:3<415::AID-PROT130>3.0.CO;2-7].

The plot returned is based on the charge-hydropathy plot from Uversky (2016) [https://doi.org/10.1080/21690707.2015.1135015].

See Uversky (2019) [https://doi.org/10.3389/fphy.2019.00010] for additional information and a recent review on the topic. This plot has also been referred to as a "Uversky Plot".

**Usage**

```r
chargeHydropathyPlot(
  sequence,
  displayInsolubility = TRUE,
  insolubleValue = 0.7,
  proteinName = NA,
  customPlotTitle = NA,
  pH = 7,
  pKaSet = "IPC_protein",
  plotResults = TRUE,
  ...
)
```

**Arguments**

- `sequence`  
amino acid sequence (or pathway to a fasta file) as a character string. Supports multiple sequences / files, as a character vector of strings. Additionally, this supports a single protein as character vectors. Multiple proteins are not supported as a character vector of single characters.

- `displayInsolubility`  
logical value, TRUE by default. This adds (or removes when FALSE) the vertical line separating collapsed proteins and insoluble proteins

- `insolubleValue`  
umerical value. 0.7 by default. Ignored when `displayInsolubility` = FALSE. Plots the vertical line \( < H > = displayInsolubility \).

- `proteinName`, `customPlotTitle`  
optional character string. NA by default. Used to either add the name of the protein to the plot title when there is only one protein, or to create a custom plot title for the output.

- `pH`  
numeric value, 7.0 by default. The environmental pH is used to calculate residue charge.
chargeHydropathyPlot

pKaSet

pKa set used for charge calculations. See netCharge for additional details

plotResults

logical value, TRUE by default. This determines what is returned. If plotResults = FALSE, a data frame is returned with the Sequence(s), Average Scaled Hydropathy, and Average Net Charge. If plotResults = TRUE, a graphical output is returned (ggplot) showing the Charge Hydropathy Plot (recommended).

... additional arguments to be passed to idpr::netCharge(), idpr::meanScaledHydropathy() or ggplot

Value

Graphical values of Charge-Hydropathy Plot

Plot Colors

For users who wish to keep a common aesthetic, the following colors are used when plotResults = TRUE.

- Point(s) = "chocolate1" or 
- Lines = "black"

References


See Also

netCharge and meanScaledHydropathy for functions used to calculate values.

Examples

#Amino acid sequences can be character strings
aaString <- "ACDEFGHIKLMNPQRSTVWY"

#Amino acid sequences can also be character vectors

#Alternatively, .fasta files can also be used by providing
DisorderMat

Disorder-based Substitution Matrices.

Description

The Disorder40, Disorder60, and Disorder85 Matrices were developed and described in Brown et al. (2009).
In short: There are substitution scoring matrices used to align proteins or regions which experience intrinsic disorder. The matrices were calculated using pairwise sequence alignments of protein families which here identified from 287 experimentally confirmed Intrinsically Disordered Proteins (IDPs). The IDPs contained at least 30 sequential residues of intrinsic disorder and protein families were found using BLAST.
There was not a comprehensive comparison to other frequently used substitution matrices (like BLOSUM and PAM) in terms of improving IDP sequence alignments. The authors note that the purpose of these matrices were made to compare evolutionary characteristics of disordered and ordered proteins. Please see the source material for additional information.

Trivedi and Nagarajaram (2019) compared EDSSMat62 against all three Disordered Matrices. Disorder40 and Disorder85 attain lower E-values for highly disordered proteins, on average, when compared to EDSSMat62. EDSSMat62 attained lower E-values when compared to Disorder60 for aligning highly disordered proteins. EDSSMat62 preforms better than all three Disorder matrices for IDPs enriched in ordered regions. Please see the referenced paper, specifically Supplementary.
Figures S18-20, for additional information and original comparison.

Additionally, please cite the source article when using Disorder40, Disorder60, or Disorder85.

Usage

Disorder40

Disorder60

Disorder85

Format

All matrices are symmetric. 24 residues are represented:

- Each of the standard 20 standard amino acids
- Four ambiguous residues:
  - B: Asparagine or Aspartic Acid (Asx)
  - Z: Glutamine or Glutamic Acid (Glx)
  - X: Unspecified or unknown amino acid
  - *: Stop

An object of class matrix (inherits from array) with 24 rows and 24 columns.

Optimal Gap Parameters

As mentioned in the Description, the intended use of these matrices was not to improve sequence alignments. Therefore, no gap penalty values are provided.

It should also be noted that a more recent work, Trivedi and Nagarajaram (2019), determined optimal parameters based on the disordered content of query sequences, as reported in the paper’s Supplementary Table S5.

<table>
<thead>
<tr>
<th>Matrix Name</th>
<th>Gap Open (LD)</th>
<th>Gap Extension (LD)</th>
<th>Gap Open (MD)</th>
<th>Gap Extension (MD)</th>
<th>Gap Open (HD)</th>
<th>Gap Extension (HD)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disorder40</td>
<td>-20</td>
<td>-1</td>
<td>-7</td>
<td>-1</td>
<td>-7</td>
<td>-7</td>
</tr>
<tr>
<td>Disorder60</td>
<td>-20</td>
<td>-1</td>
<td>-16</td>
<td>-1</td>
<td>-11</td>
<td>-11</td>
</tr>
<tr>
<td>Disorder85</td>
<td>-20</td>
<td>-1</td>
<td>-16</td>
<td>-1</td>
<td>-7</td>
<td>-7</td>
</tr>
</tbody>
</table>

Please see the referenced paper for additional information and original reporting. Additionally, please see EDSSMat.

Additional Reference

Trivedi, R., Nagarajaram, H.A. Amino acid substitution scoring matrices specific to intrinsically disordered regions in proteins. Sci Rep 9, 16380 (2019). [https://doi.org/10.1038/s41598-019-52532-8](https://doi.org/10.1038/s41598-019-52532-8)
DUNMat

Source

See Also
Disordered Matrices Vignette within the idpr package and EDSSMat62
Other IDP-based Substitution Matrices: DUNMat, EDSSMat

DUNMat
A Substitution Matrix for Aligning Intrinsically Disordered Proteins

Description
This matrix was developed and described in Radivojac et al. (2002).
The name “DUNMat” is taken from Trivedi and Nagarajaram (2019). This is to keep naming consistent and distinct from other matrices named “Disorder”.
In short: This is a substitution scoring matrix used to align proteins or regions which experience intrinsic disorder. The scores for this matrix are derived from proteins that have long regions of disorder (LDR), defined in this paper as an intrinsically disordered region (IDR) of at least 40 sequential residues. 55 protein families with LDRs were used to generate the data. Direct comparisons were not made against BLOSUM or PAM matrices within the source paper due to differences in scaling, however, when ranking its performance, it preformed the best in aligning proteins with less than 50% sequence identity. Please see the source material, specifically, table 2, for additional information.

Trivedi and Nagarajaram (2019) compared EDSSMat62 and DUNMat and show that DUNMat, on average, attained smaller E-values in the dataset of IDPs enriched in ordered regions, while EDSSMat62 attained smaller E-values in sets of highly disordered IDPs. Please see the referenced paper, specifically Supplementary Figure S21, for additional information and original comparison.

Additionally, please cite the source article when using the “DUNMat” Matrix.

Usage
DUNMat

Format
A symmetrical matrix. 20x20 representing the 20 standard amino acids

Optimal Gap Parameters
These values were described in the source article and reported in Table 2. After the optimal parameters were determined, the authors further refined the gap costs. Therefore, it is recommended to use these parameters for any alignment utilizing this matrix. These were:
It should also be noted that a more recent work, Trivedi and Nagarajaram (2019), determined optimal parameters based on the disordered content of query sequences, as reported in the paper’s Supplementary Table S5.

Please see the referenced paper for additional information and original reporting. Additionally, please see EDSSMat.

**Additional Reference**


**Source**


**See Also**

EDSSMat62 and the Disordered Matrices Vignette within idpr

Other IDP-based Substitution Matrices: DisorderMat, EDSSMat

---

**Description**

The EDSSMat series of matrices were developed and described in Trivedi and Nagarajaram (2019). In short: These are substitution scoring matrices used to align proteins or regions which experience intrinsic disorder. Alignment blocks, used to compute the matrix values, were composed of predicted intrinsically disordered regions. When compared to other, more frequently used substitution matrices (like BLOSUM and PAM), EDSSMat had significantly smaller E-values when aligning regions of disorder. Additionally, EDSSMat62 was shown to identify both close and distant homologs of a specific IDP while other matrices could only identify some close homologs. See the source article for additional information and for comparisons to other matrices.

Additionally, please cite the source article when using any EDSSMat matrix.

<table>
<thead>
<tr>
<th>Matrix Name</th>
<th>Gap Open (LD)</th>
<th>Gap Extension (LD)</th>
<th>Gap Open (MD)</th>
<th>Gap Extension (MD)</th>
<th>Gap Open (HD)</th>
<th>Gap Extension (HD)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DUNMat</td>
<td>-6</td>
<td>-1</td>
<td>-6</td>
<td>-1</td>
<td>-16</td>
<td>-2</td>
</tr>
</tbody>
</table>
Usage

EDSSMat50
EDSSMat60
EDSSMat62
EDSSMat70
EDSSMat75
EDSSMat80
EDSSMat90

Format

All matrices are symmetric. 24 residues are represented:

- Each of the standard 20 standard amino acids
- Four ambiguous residues:
  - B: Asparagine or Aspartic Acid (Asx)
  - Z: Glutamine or Glutamic Acid (Glx)
  - X: Unspecified or unknown amino acid
  - *: Stop

An object of class `matrix` (inherits from `array`) with 24 rows and 24 columns.
An object of class `matrix` (inherits from `array`) with 24 rows and 24 columns.
An object of class `matrix` (inherits from `array`) with 24 rows and 24 columns.
An object of class `matrix` (inherits from `array`) with 24 rows and 24 columns.
An object of class `matrix` (inherits from `array`) with 24 rows and 24 columns.
An object of class `matrix` (inherits from `array`) with 24 rows and 24 columns.
An object of class `matrix` (inherits from `array`) with 24 rows and 24 columns.

Matrices

There are 7 reported EDSSMat matrices. Each vary depending on the percent identity threshold used to cluster protein sequences. EDSSMat50 clustered proteins with 50% identity or higher, EDSSMat62 clustered proteins with 62% identity or higher, etc.

See Usage Section for available matrices

Optimal Gap Parameters

These values were described in the source article and reported in Supplemental Table S5. Therefore, it is recommended to use these parameters for any alignment utilizing the respective EDSS matrix. These were determined for 3 categories: Proteins containing Less Disorder (LD), defined as [0-20%] disorder, Moderate Disorder (MD), defined as [20-40%] disorder, and High Disorder (HD),


Defined as (40-100\%) disorder. Please see the source article for additional information.

<table>
<thead>
<tr>
<th>Matrix Name</th>
<th>Gap Open (LD)</th>
<th>Gap Extension (LD)</th>
<th>Gap Open (MD)</th>
<th>Gap Extension (MD)</th>
<th>Gap Open (HD)</th>
<th>Gap Extension (HD)</th>
</tr>
</thead>
<tbody>
<tr>
<td>EDSSMat60</td>
<td>-7</td>
<td>-1</td>
<td>-6</td>
<td>-2</td>
<td>-14</td>
<td>-3</td>
</tr>
<tr>
<td>EDSSMat62</td>
<td>-8</td>
<td>-1</td>
<td>-5</td>
<td>-2</td>
<td>-19</td>
<td>-2</td>
</tr>
<tr>
<td>EDSSMat70</td>
<td>-7</td>
<td>-1</td>
<td>-5</td>
<td>-2</td>
<td>-19</td>
<td>-2</td>
</tr>
<tr>
<td>EDSSMat75</td>
<td>-8</td>
<td>-1</td>
<td>-5</td>
<td>-2</td>
<td>-19</td>
<td>-15</td>
</tr>
<tr>
<td>EDSSMat80</td>
<td>-7</td>
<td>-1</td>
<td>-5</td>
<td>-2</td>
<td>-19</td>
<td>-15</td>
</tr>
<tr>
<td>EDSSMat90</td>
<td>-7</td>
<td>-1</td>
<td>-5</td>
<td>-2</td>
<td>-19</td>
<td>-15</td>
</tr>
</tbody>
</table>

Source


See Also

Disordered Matrices Vignette within the idpr package

Other IDP-based Substitution Matrices: DUNMat, DisorderMat

---

foldIndexR

Prediction of Intrinsic Disorder with FoldIndex method in R

Description

This is used to calculate the prediction of intrinsic disorder based on the scaled hydropathy and absolute net charge of an amino acid sequence using a sliding window. FoldIndex described this relationship and implemented it graphically in 2005 by Prilusky, Felder, et al, and this tool has been implemented into multiple disorder prediction programs. When windows have a negative score (<0) sequences are predicted as disordered. When windows have a positive score (>0) sequences are predicted as disordered. Graphically, this cutoff is displayed by the dashed line at y = 0. Calculations are at pH 7.0 based on the described method and the default is a sliding window of size 51.

Usage

foldIndexR(
  sequence,
  window = 51,
  proteinName = NA,
  pKaSet = "IPC_protein",
  plotResults = TRUE,
  ...
)
Arguments

- **sequence**
  amino acid sequence as a single character string, a vector of single characters, or an AAString object. It also supports a single character string that specifies the path to a .fasta or .fa file.

- **window**
  a positive, odd integer. 51 by default. Sets the size of sliding window, must be an odd number. The window determines the number of residues to be analyzed and averaged for each position along the sequence.

- **proteinName**
  character string with length = 1. optional setting to replace the name of the plot if plotResults = TRUE.

- **pKaSet**
  A character string or data frame. "IPC_protein" by default. Character string to load specific, preloaded pKa sets. c("EMBOSS", "DTASelect", "Solomons", "Silero", "Rodwell", "Lehninger", "Toseland", "Thurlkill", "Nozaki", "Dawson", "Bjellqvist", "ProMoST", "Vollhardt", "IPC_protein", "IPC_peptide") Alternatively, the user may supply a custom pKa dataset. The format must be a data frame where: Column 1 must be a character vector of residues named "AA" AND Column 2 must be a numeric vector of pKa values.

- **plotResults**
  logical value, TRUE by default. If plotResults = TRUE a plot will be the output. If plotResults = FALSE the output is a data frame with scores for each window analyzed.

- ... any additional parameters, especially those for plotting.

Details

The output is either a data frame or graph showing the calculated scores for each window along the sequence. The equation used was originally described in Uversky et al. (2000) [https://doi.org/10.1002/1097-0134(20001115)41:3<415::AID-PROT130>3.0.CO;2-7].

Value

see plotResults argument

Plot Colors

For users who wish to keep a common aesthetic, the following colors are used when plotResults = TRUE:

- Dynamic line colors:
  - Close to -1 = "#9672E6"
  - Close to 1 = "#D1A63F"
  - Close to midpoint = "grey65" or "#A6A6A6"
hendersonHasselbalch

References


See Also

KDNorm for residue hydropathy values. See pKaData for residue pKa values and citations. See hendersonHasselbalch for charge calculations.

Other scaled hydropathy functions: KDNorm, meanScaledHydropathy(), scaledHydropathyGlobal(), scaledHydropathyLocal()

hendersonHasselbalch  Henderson-Hasselbalch Equation

Description

This function calculates the ionic charge of a residue at a specific pH when given the pKa. $pH = pKa + \log([A^-]/[HA])$ Known, charged residues are accepted as well as the protein termini and general property to allow customized calculations. The output is a ratio comparing acid to conjugate base for acidic residues or a ratio comparing conjugate base to acid for basic residues.

Usage

hendersonHasselbalch(pKa, pH = 7, residue)

Arguments

pKa numeric value. The point where $A^- = HA$.

pH numeric value. The pH of the environment. 7.0 by default

residue individual character or character string. accepted values are the exact aa c("C", "D", "E", "H", "K", "R", "Y"), termini c("COOH","COO","NH2","NH3"), or a general property c("acid", "base", "negative", "positive").

Value

a numeric value giving the ratio of charged to uncharged residues.
See Also

pKaData for residue pKa values and citations. See other charge functions for use.

Other charge functions: chargeCalculationGlobal(), chargeCalculationLocal(), netCharge()

Examples

#Calculating Lysine charge using the EMBOSS pKa data
EMBOSS_pKa <- pKaData[, 1:2]
EMBOSS_pKa

Lys_pKa <- EMBOSS_pKa[EMBOSS_pKa$AA == "K",]
Lys_pKa$EMBOSS #This is Lysines pKa

hendersonHasselbalch(
  pKa = as.numeric(Lys_pKa$EMBOSS),
  pH = 7.0,
  residue = "K")

#residue = supports general properties as well
hendersonHasselbalch(
  pKa = as.numeric(Lys_pKa$EMBOSS),
  pH = 7.0,
  residue = "base")

hendersonHasselbalch(
  pKa = as.numeric(Lys_pKa$EMBOSS),
  pH = 7.0,
  residue = "positive")

#CALCULATIONS ARE DEPENDENT ON RESIDUE PROPERTY!
hendersonHasselbalch(
  pKa = as.numeric(Lys_pKa$EMBOSS),
  pH = 7.0,
  residue = "acid") #Inaccurate Description

#You can also calculate charge at different pHs
hendersonHasselbalch(
  pKa = as.numeric(Lys_pKa$EMBOSS),
  pH = 5.5,
  residue = "K")

hendersonHasselbalch(
  pKa = as.numeric(Lys_pKa$EMBOSS),
  pH = 8,
  residue = "K")
Description

idpr aims to integrate tools for the computational analysis of intrinsically disordered proteins (IDPs) within R. This package is used to identify known characteristics of IDPs for a sequence of interest with easily reported and dynamic results. Additionally, this package includes tools for IDP-based sequence analysis to be used in conjunction with other R packages. Please see the idpr vignettes for details on idpr functions and theory. browseVignettes("idpr")

References


idprofile

IDp PROFILE From idpr Package

Description

The IDPProfile is a summation of many features of the idpr package, conveniently grouped into one function for quick analysis. This combines many plotting functions in this package. These include:
chargeHydropathyPlot
chargeCalculationLocal
scaledHydropathyLocal
structuralTendencyPlot
foldIndexR

All of the above linked functions only require the sequence argument to output plots of characteristics associated with IDPs. The function also includes options for IUPred functions. The function does one of the following based on user-specified parameters:
iupred
iupredAnchor
iupredRedox

The IUPred function used depends on the argument of iupredType. All require the UniProt Accession to make a proper connection to the IUPred2A REST API. If the UniProt Accession is not specified, the IUPred plot is skipped.

Usage

idprofile(
  sequence,
  uniprotAccession = NA,
  proteinName = NA,
  iupredType = "long",
  window = 9,
  foldIndexWindowSize = 51,
  pH = 7,
  foldIndex_pH = 7,
  pKaSet = "IPC_protein",
  structuralTendencyType = "bar",
)

structuralTendencySummarize = FALSE,
disorderNeutral = c("D", "T", "R"),
)

Arguments

sequence  amino acid sequence as a single character string or vector of single characters. It also supports a single character string that specifies the location of a .fasta or .fa file.

uniprotAccession  character string specifying the UniProt Accession of the protein of interest. Used to fetch predictions from IUPred2 REST API. Default is NA. Keep as NA if you do not have a UniProt Accession.

proteinName  character string, optional. Used to add protein name to the title in ggplot.

iupredType  character string specifying the type of IUPred2 prediction to retrieve. Can be c("long", "short", "glob", "anchor", "redox"). "long" by default. "long", "short", and "glob" use the iupred function and specify the type of plot. Both "redox" and "anchor" use "long" for predictions, but are context dependent. "anchor" uses iupredAnchor to get predictions of disorder with IUPred2 and predictions of induced folding based on ANCHOR2 predictions (Shown with a red line). "redox" uses iupredRedox to make predictions of disorder based on environmental conditions. Regions of predicted environmental sensitivity are highlighted. See the respective functions for more details. This is skipped if uniprotAccession = NA.

window  a positive, odd integer. 9 by default. Sets the size of sliding window, must be an odd number. The window determines the number of residues to be analyzed and averaged for each position along the sequence. For chargeCalculationLocal and scaledHydropathyLocal.

foldIndexWindowSize  a positive, odd integer. 51 by default. Sets the size of sliding window, must be an odd number. The window determines the number of residues to be scored and averaged for each position along the sequence.

pH  numeric value, 7.0 by default. The environmental pH used to calculate residue charge.

foldIndex_pH  numeric value, 7.0 by default. The environmental pH used to calculate residue charge. FoldIndex specifically uses pH = 7.0 in settings and thus, is distinct from changing pH in the other calculations.

pKaSet  A character string or data frame. "IPC_protein" by default. Character string to load specific, preloaded pKa sets. c("EMBOSS", "DTASelect", "Solomons", "Sillero", "Rodwell", "Lehninger", "Toseland", "Thurlkill", "Nozaki", "Dawson", "Bjellqvist", "ProMoST", "Vollhardt", "IPC_protein", "IPC_peptide") Alternatively, the user may supply a custom pKa dataset. The format must be a data frame where: Column 1 must be a character vector of residues named "AA" AND Column 2 must be a numeric vector of pKa values.
structuralTendencyType

a character string specifying the type of plot the structuralTendencyPlot should output. Can be "bar" or "pie". Equivalent argument to graphType= in the linked function. "bar" by default.

structuralTendencySummarize

a logical value specifying the structuralTendencyPlot should be summarized into broad categories. Equivalent argument to summarize= in the linked function. FALSE by default

disorderPromoting, disorderNeutral, orderPromoting

character vectors of individual residues to be matched with the input sequence. Defaults:

• disorderPromoting = c("P", "E", "S", "Q", "K", "A", "G")
• disorderNeutral = c("D", "T", "R")

It is not recommended to change these. Arguments passed to structuralTendencyPlot

Details


Value

4 or 5 plots, depending if a UniProt Accession is provided.

Citations for each Plot

• chargeHydropathyPlot
  – If a pKa set is specified, see pKaData
• chargeCalculationLocal
  – If a pKa set is specified, see pKaData
• scaledHydropathyLocal
• **structuralTendencyPlot**
  

• **foldIndexR**
  
  
  
  – Also see citations for hydropathy and charge plots above

• **iupred, iupredAnchor, iupredRedox**
  
  

See Also

chargeHydropathyPlot
chargeCalculationLocal
scaledHydropathyLocal
structuralTendencyPlot
foldIndexR
iupred
iupredAnchor
iupredRedox

Examples

---

```r
# For most functions, a protein sequence is all that is needed.

# The UniProt ID is optional but recommended for IUPred results.
proteinID <- "P04637"
p53Seq <- idpr:::TP53Sequences[2]
## Not run:
idprofile(
  sequence = p53Seq,
  uniprotAccession = proteinID)

# Changing the iupred to redox
## and getting a pie chart for structuralTendency.
idprofile(
  sequence = p53Seq,
  uniprotAccession = proteinID,
pKaSet = EMBOSs,
```

---

**Description**

This function makes a connection to the IUPred2A REST API based on the type of analysis and UniProt accession number. This requires the user to know the accession number of their protein and a connection to the internet. The results are then formatted to match output in the idpr package.

Predictions are made on a scale of 0-1, where any residues with a score over 0.5 are predicted to be disordered, and any residue scoring below 0.5 are predicted to be ordered (when using "long" and "short" predictions).

The output is either a graph (ggplot) or data frame of predictions.

**iupred()** is used for standard predictions of intrinsic disorder of an amino acid sequence. This is the core of predictions. Predictions vary by iupredType (details below) The results are either a ggplot or data frame of the fetched IUPred2 long predictions.

**iupredAnchor()** is used to combine the output of IUPred2 long with ANCHOR2 predictions. ANCHOR2 is a context-dependent predictor of binding regions for protein-protein interactions. The results are either a ggplot with 2 lines, one for IUPred2 long and another for ANCHOR predictions, or a data frame with both IUPred2 long and ANCHOR Predictions. Values are fetched by the IUPred2A REST API.

**iupredRedox()** is used to predict redox-sensitive regions that may experience induced folding upon changing environments. This is a context-dependent predictor of disordered regions depending on a reducing (plus) or oxidizing (minus) environment. The results can be a ggplot with two IUPred2 long predictions, one for plus and another for minus environments, with redox sensitive regions shaded (if predicted). Alternatively, the results can be a data frame with both IUPred2 long plus and minus predictions as well as a column of logical values where a residue that is TRUE is predicted to be in a redox sensitive region. Values are fetched by the IUPred2A REST API.

IUPred2 website is located at [https://iupred2a.elte.hu/](https://iupred2a.elte.hu/). For detailed information on using IUPred2A, please refer to Erdős & Dosztány (2020) Analyzing protein disorder with IUPred2A. Current Protocols in Bioinformatics, 70, e99. Additionally, please see Mészáros et al (2019) for further information, theory, and applications of IUPred2A.

Please cite these articles if you use any iupred function.

**Usage**

```r
iupred(
  uniprotAccession,
  iupredType = "redox",
  structuralTendencyType = "pie")
```

## End(Not run)
iupredType = "long",
plotResults = TRUE,
proteinName = NA
)

iupredAnchor(uniprotAccession, plotResults = TRUE, proteinName = NA)
iupredRedox(uniprotAccession, plotResults = TRUE, proteinName = NA)

Arguments

uniprotAccession
character string specifying the UniProt Accession of the protein of interest. Used to fetch predictions from IUPreds REST API

iupredType
character string. "long" by default. accepted types are c("long", "short", "glob"). See "Prediction Type" information below.

plotResults
logical value. TRUE by default. If plotResults = TRUE, a ggplot of IUPred predictions is returned If plotResults = FALSE, a data frame of predictions is returned.

proteinName
character string, optional. Used to add protein name to the title in ggplot. Ignored if plotResults = FALSE.

Value

see plotResults argument.

Prediction Type

Information from https://iupred2a.elte.hu/help_new on 5.22.20 Additionally, see the sources for further details and source information. This is only relevant for iupred(). iupredAnchor() and iupredRedox() always utilize "long" for data in the REST API.

• Long predictions of disorder (Default)
  – when iupredType = "long"
  – Optimized for global predictions of disorder, specifically disordered regions over 30 amino acids in length.
  – "long" is always used for iupredAnchor() and iupredRedox().

• Short predictions of disorder
  – when iupredType = "short"
  – Best for predicting small regions of disorder, especially in mostly structured proteins.
  – Has adjustments for termini, since sequence ends are often disordered.

• Structured predictions
  – when iupredType = "glob"
  – Used to predict regions of globular folding.
  – please see Erdős & Dosztány (2020) for further information on interpreting these results.
Plot Colors

For users who wish to keep a common aesthetic, the following colors are used when plotResults = TRUE.

- iupred() iupredType = 'long', 'short', or 'glob'. Additionally, the 'long' prediction with iupredAnchor().
  - Dynamic iupred line colors:
    * Close to 0 = "darkolivegreen3" or "#A2CD5A"
    * Close to 1 = "darkorchid1" or "#BF3EFF"
    * Close to 0.5 (midpoint) = "grey65" or "#A6A6A6"
- iupredAnchor:
  - Solid Line (ANCHOR2 Score) = "#92140C"
- iupredRedox:
  - iupredPlus line = "darkorchid1" or "#BF3EFF"
  - iupredMin line = "#348AA7"
  - redox sensitive regions = "#5DD39E"

Source


Examples

```r
#A UniProt Accession must be specified.
##this example uses human P53.
TP53_UniProt <- "P04637"
## Not run:
#Getting data as a data frame
exampleDF_long <- iupred(uniprotAccession = TP53_UniProt, 
iupredType = "long", 
plotResults = FALSE)
head(exampleDF_long)

exampleDF_short <- iupred(uniprotAccession = TP53_UniProt, 
iupredType = "short", 
plotResults = FALSE)
head(exampleDF_short)

exampleDF_anchor <- iupredAnchor(uniprotAccession = TP53_UniProt, 
plotResults = FALSE)
head(exampleDF_anchor)
```
exampleDF_redox <- iupredRedox(uniprotAccession = TP53_UniProt,
plotResults = FALSE)
head(exampleDF_redox)

#Plotting
iupred(uniprotAccession = TP53_UniProt,
iupredType = "long",
plotResults = TRUE)

iupred(uniprotAccession = TP53_UniProt,
iupredType = "short",
plotResults = TRUE)

iupredAnchor(uniprotAccession = TP53_UniProt,
plotResults = TRUE)

iupredRedox(uniprotAccession = TP53_UniProt,
plotResults = TRUE)

## End(Not run)
#A valid internet connection is needed to make
##A connection with the IUPred REST API

---

**KDNorm**

*Kyte and Doolittle Scaled Hydropathy Index*

**Description**

A dataset containing a measure of hydropathy for each amino acid residue as reported by Kyte J. and Doolittle R.F. (1982). Values normalized from 0 to 1. If you use these values, please cite the source article.

**Usage**

KDNorm

**Format**

a data frame with 20 rows, one for each standard amino acid, and 2 variables

- **V1** Amino acid residues as a single letter
- **V2** Scaled Hydropathy of a residue, measured 0-1

**Source**

meanScaledHydropathy

See Also
Other scaled hydropathy functions: foldIndexR(), meanScaledHydropathy(), scaledHydropathyGlobal(), scaledHydropathyLocal()

**meanScaledHydropathy**  
*Calculate the Mean Scaled Hydropathy*

**Description**
This function utilizes the scaledHydropathyGlobal() function and easily returns the averaged hydropathy as a numeric value.

**Usage**
```r
meanScaledHydropathy(sequence, roundScore = NA)
```

**Arguments**
- `sequence`  
amino acid sequence as a single character string, a vector of single characters, or an AAString object. It also supports a single character string that specifies the path to a .fasta or .fa file.
- `roundScore`  
Number of decimals the score will be rounded to. NA by default.

**Value**
A numeric value equal to the Mean Scaled Hydropathy.

**References**

**See Also**
KDNorm for residue values.
Other scaled hydropathy functions: KDNorm, foldIndexR(), scaledHydropathyGlobal(), scaledHydropathyLocal()

**Examples**
```r
#Amino acid sequences can be character strings
aaString <- "ACDEFGHIKLMNPQRSTVWY"
#Amino acid sequences can also be character vectors
aaVector <- c("A", "C", "D", "E", "F",
    "G", "H", "I", "K", "L",
    "M", "N", "P", "Q", "R",
    "S", "T", "V", "W", "Y")
#Alternatively, .fasta files can also be used by providing
```
#Calculate the mean scaled hydropathy
meanScaledHydropathy(aaString)
meanScaledHydropathy(aaVector)

netCharge

Protein Charge Calculation, Net Charge

Description

This function will determine the net charge of a peptide using the Henderson-Hasselbalch Equation. The output is a numeric value describing the total net charge or the average net charge.

Usage

netCharge(
  sequence,
  pKaSet = "IPC_protein",
  pH = 7,
  includeTermini = TRUE,
  averaged = FALSE
)

Arguments

sequence  amino acid sequence as a character string or vector of individual residues. alternatively, a character string of the path to a .fasta / .fa file
pKaSet    A character string or data frame. "IPC_protein" by default. Character string to load specific, preloaded pKa sets. c("EMBOSS", "DTASelect", "Solomons", "Sillero", "Rodwell", "Lehninger", "Toseland", "Thurlkill", "Nozaki", "Dawson", "Bjellqvist", "ProMoST", "Vollhardt", "IPC_protein", "IPC_peptide") Alternatively, the user may supply a custom pKa dataset. The format must be a data frame where: Column 1 must be a character vector of residues named "AA" AND Column 2 must be a numeric vector of pKa values.
pH        numeric value, 7.0 by default. The environmental pH used to calculate residue charge.
includeTermini Logical value, TRUE by default. This determines how the calculation handles the N- and C- terminus. includeTermini determines if the calculation will use the charge of the amine and carboxyl groups at the ends of the peptide (When TRUE). These charges are ignored when includeTermini = FALSE.
averaged  logical value. FALSE by default. When averaged = FALSE, the total net charge is returned. When averaged = TRUE, the total net charge is averaged by the sequence length. This gives a value of -1 to +1.

Value

numeric value. Either the net charge or average net charge, depending on the value of the averaged argument
See Also

`pKaData` for residue pKa values and citations. See `hendersonHasselbalch` for charge calculations.

Other charge functions: `chargeCalculationGlobal()`, `chargeCalculationLocal()`, `hendersonHasselbalch()`

Examples

```r
# Amino acid sequences can be character strings
aaString <- "ACDEFGHIKLMNPQRSTVWY"
# Amino acid sequences can also be character vectors
aaVector <- c("A", "C", "D", "E", "F",
              "G", "H", "I", "K", "L",
              "M", "N", "P", "Q", "R",
              "S", "T", "V", "W", "Y")

# Alternatively, .fasta files can also be used by providing a character string # of the path to the file.

# Calculate the Net Charge
netCharge(aaString,
         averaged = FALSE)
netCharge(aaVector,
         averaged = FALSE)

# Calculate the Average Net Charge
netCharge(aaString,
         averaged = TRUE)
netCharge(aaVector,
         averaged = TRUE)

# Change the pH
netCharge(aaString,
         pH = 8)
netCharge(aaString,
         pH = 7)
netCharge(aaString,
         pH = 5.5)

# Specify which pKa set to use
netCharge(aaString,
         pKaSet = "IPC_protein") # Default
netCharge(aaString,
         pKaSet = "IPC_peptide")
netCharge(aaString,
         pKaSet = "Dawson")
netCharge(aaString,
         pKaSet = "EMBOSS")

# Should the termini be included in charge calculations?
netCharge(aaString,
          includeTermini = TRUE) # Default
netCharge(aaString,
          includeTermini = FALSE)
```
Description

A dataset containing the various pKa accepted values for each charged amino acid residue. N- and C-terminus values are also included. See "IPC - Isoelectric Point Calculator" Kozlowski (2016) for information on variability in pKa Data sets. https://doi.org/10.1186/s13062-016-0159-9

Citations are also contained in the data frame for convenience. Please cite the specific pKa set source and/or Kozlowski (2016).

Usage

pKaData

Format

a data frame with 10 rows and 16 variables.

AA  Amino acid residues as a single letter.
    Residues are: Cys (C), Asp (D), Glu (E), His (H), Lys (K), Arg (R), and Tyr (Y).
    N- and C-termini as NH2 and COOH, respectively. "citation" in the final row

EMBOSS  pKa Dataset from https://doi.org/10.1016/S0168-9525(00)02024-2
DTASelect  pKa Dataset from https://doi.org/10.1021/pr015504q
Solomons  pKa Dataset from: https://doi.org/10.1186/s13062-016-0159-9
    pKa Dataset original source: ISBN: 978-1-118-87576-6
Sillero  pKa Dataset from https://doi.org/10.1016/0003-2697(89)90136-X
Rodwell  pKa Dataset from https://doi.org/10.1016/0003-2697(82)90611-X
Toseland  pKa Dataset from https://doi.org/10.1093/nar/gkj035
Thurlkill  pKa Dataset from https://doi.org/10.1110/ps.051840806
Nozaki  pKa Dataset from: https://doi.org/10.1110/ps.051840806
    pKa Dataset original source: https://doi.org/10.1016/S0076-6879(67)11088-4
Dawson  pKa Dataset from: https://doi.org/10.1186/s13062-016-0159-9
    pKa Dataset original source: ISBN: 9780198552994
Bjellqvist  pKa Dataset from: https://doi.org/10.1186/s13062-016-0159-9
    pKa Dataset original source: https://doi.org/10.1002/elps.1150150171
ProMoST  pKa Dataset from https://doi.org/10.1093/nar/gkh356
IPC_protein  pKa Dataset from https://doi.org/10.1186/s13062-016-0159-9
IPC_peptide  pKa Dataset from https://doi.org/10.1186/s13062-016-0159-9
Additional Information

Values for NH₂ and COOH are averages of values provided within the Lehringer, ProMoST, and Vollhardt datasets. Lehringer and Vollhardt both are the Seventh edition. Lehringer varies from data presented in the IPC paper.

When values could not be sourced to the original source, values were taken from Kozlowski (2016), https://doi.org/10.1186/s13062-016-0159-9. Both Kozlowski (2016) and the original source DOI (where available) or ISBN are provided within the Format section of this documentation.

Full Citations

Dawson, Elliott, Elliott, & Jones, 2002; Halligan et al., 2004; Kozlowski, 2016; Nelson & Cox, 2017; Nozaki & Tanford, 1967; Rice, Longden, & Bleasby, 2000; Rodwell, 1982; Sillero & Ribeiro, 1989; Tabb, McDonald, & Yates, 2002; TG, 1992; Thurlkill, Grimsley, Scholtz, & Pace, 2006; Toseland, McSparron, Davies, & Flower, 2006; Vollhardt & Schore, 2014


### scaledHydropathyGlobal

**Protein Scaled Hydropathy Calculations**

#### Description

This is used to calculate the scaled hydropathy of an amino acid sequence for each residue in the sequence. The output is either a data frame or graph showing the matched scores for each residue along the sequence.

#### Usage

```r
scaledHydropathyGlobal(sequence, plotResults = FALSE, proteinName = NA, ...)```

#### Arguments

- `sequence`: amino acid sequence as a single character string, a vector of single characters, or an AAString object. It also supports a single character string that specifies the path to a .fasta or .fa file.

- `plotResults`: logical value, FALSE by default. If `plotResults = TRUE` a plot will be the output. If `plotResults = FALSE` the output is a data frame for each residue.

- `proteinName`: character string with length = 1. optional setting to include the name in the plot title.

- `...`: any additional parameters, especially those for plotting.

#### Value

If `plotResults = TRUE`, a graphical representation data. Average is shown by the horizontal line. If `plotResults = FALSE`, a data frame is reported with each amino acid and each residue value shown. Score for each residue shown in the column "Hydropathy".

#### Plot Colors

For users who wish to keep a common aesthetic, the following colors are used when `plotResults = TRUE`.

- Dynamic line colors:
- Close to 0 = "skyblue3" or "#6CA6CD"
- Close to 1 = "chocolate1" or "#FF7F24"
- Close to midpoint = "grey65" or "#A6A6A6"

References


See Also

KDNorm for residue values.

Other scaled hydropathy functions: KDNorm, foldIndexR(), meanScaledHydropathy(), scaledHydropathyLocal()

Examples

#Amino acid sequences can be character strings
aaString <- "ACDEFGHIKLMNPQRSTVWY"

#Amino acid sequences can also be character vectors

#Alternatively, .fasta files can also be used by providing
#The path to the file as a character string

eampleDF <- scaledHydropathyGlobal(aaString,
plotResults = FALSE)

head(exampleDF)

eampleDF <- scaledHydropathyGlobal(aaVector,
plotResults = FALSE)

head(exampleDF)

#plotResults = TRUE will output a ggplot
scaledHydropathyGlobal(aaString,
plotResults = TRUE)

#since it is a ggplot, you can change or annotate the plot

gg <- scaledHydropathyGlobal(aaVector,
plotResults = TRUE)

gg <- gg + ggplot2::ylab("Local Hydropathy")

gg <- gg + ggplot2::geom_text(data = exampleDF,

plot(gg)
Calculate the Average Scaled Hydropathy of an Amino Acid Sequence

Description

This is used to calculate the scaled hydropathy of an amino acid sequence using a sliding window. The output is either a data frame or graph showing the calculated scores for each window along the sequence.

Usage

```r
scaledHydropathyLocal(
  sequence,
  window = 9,
  plotResults = TRUE,
  proteinName = NA,
  ...
)
```

Arguments

- **sequence**: amino acid sequence as a single character string, a vector of single characters, or an AAString object. It also supports a single character string that specifies the path to a .fasta or .fa file.
- **window**: a positive, odd integer. 9 by default. Sets the size of sliding window, must be an odd number. The window determines the number of residues to be analyzed and averaged for each position along the sequence.
- **plotResults**: logical value, TRUE by default. If `plotResults = TRUE` a plot will be the output. If `plotResults = FALSE` the output is a data frame with scores for each window analyzed.
- **proteinName**: character string with length = 1. optional setting to replace the name of the plot if hydropathy = TRUE.
- **...**: any additional parameters, especially those for plotting.

Value

see `plotResults` argument

Plot Colors

For users who wish to keep a common aesthetic, the following colors are used when `plotResults = TRUE`.

- Dynamic line colors:
  - Close to 0 = "skyblue3" or "#6CA6CD"
  - Close to 1 = "chocolate1" or "#FF7F24"
  - Close to midpoint = "grey65" or "#A6A6A6"
References


See Also

KDNorm for residue values.
Other scaled hydropathy functions: KDNorm, foldIndexR(), meanScaledHydropathy(), scaledHydropathyGlobal()

Examples

# Amino acid sequences can be character strings
aaString <- "ACDEFGHIKLMNPQRSTVWY"
# Amino acid sequences can also be character vectors
aaVector <- c("A", "C", "D", "E", "F",
              "G", "H", "I", "K", "L",
              "M", "N", "P", "Q", "R",
              "S", "T", "V", "W", "Y")

# Alternatively, .fasta files can also be used by providing
## The path to the file as a character string.
exampleDF <- scaledHydropathyLocal(aaString,
plotResults = FALSE)
head(exampleDF)

exampleDF <- scaledHydropathyLocal(aaVector,
plotResults = FALSE)
head(exampleDF)

# Changing window will alter the number of residues analyzed
exampleDF_window3 <- scaledHydropathyLocal(aaString,
window = 3,
plotResults = FALSE)
head(exampleDF_window3)
exampleDF_window15 <- scaledHydropathyLocal(aaString,
window = 15,
plotResults = FALSE)
head(exampleDF_window15)

# plotResults = TRUE will output a ggplot
scaledHydropathyLocal(aaString,
plot = TRUE)

# since it is a ggplot, you can change or annotate the plot
gg <- scaledHydropathyLocal(aaVector,
window = 3,
plot = TRUE)

gg <- gg + ggplot2::ylab("Local Hydropathy")

gg <- gg + ggplot2::geom_text(data = exampleDF_window3,
ggplot2::aes(label = CenterResidue,
y = WindowHydropathy + 0.1))

plot(gg)
sequenceCheck  
**Sequence Check Function**

### Description

This is used to validate a sequence of amino acids. It can additionally be used to load an amino acid sequence. It can also be used to coerce a sequence into a specific format.

### Usage

```r
sequenceCheck(
  sequence,
  method = "stop",
  outputType = "string",
  nonstandardResidues = NA,
  suppressAAWarning = FALSE,
  suppressOutputMessage = FALSE
)
```

### Arguments

- **sequence**: amino acid sequence as a single character string, a vector of single characters, or an AAString object. It also supports a single character string that specifies the path to a .fasta or .fa file.
- **method**: Required Setting. `method = c("stop", "warn")`. "stop" by default. "stop" Reports invalid residues as an error and prevents the function from continuing. "warn" Reports invalid residues through a warning. Any invalid sequences will be reported as intended.
- **outputType**: Required Setting. "string" By default. `outputType = c("string", "vector", "none")"string" returns the sequence as a single string of amino acids. "vector" returns the sequence as a vector of individual characters. "none" prevents the function from returning a sequence.
- **nonstandardResidues**: Optional setting. Expands the amino acid alphabet. NA or Character vector required. Default values are "ACDEFGHIKLMNPQRSTVWY". Additional letters added here. `nonstandardResidues = c("O", "U")` to allow Pyrrolysine (O) and Selenocysteine (U).
- **suppressAAWarning**: If using nonstandardResidues, a warning will be issued. Set `suppressAAWarning = T` to confirm addition of non-standard residues.
- **suppressOutputMessage**: Set `suppressOutputMessage = T` to prevent sequence validity message
Value

A message and sequence are returned. If suppressOutputMessage = T, the message is not returned. If outputType = "None"), the sequence is not returned. Otherwise, outputType will determine the format of the returned sequence. If the sequence contains an error, it will be reported based on the value of method. The Sequence will be assigned to the value "Sequence" if sequenceName is not specified. Otherwise the sequence is assigned to the value of sequenceName. This allows the sequences to be called by the user.

Examples

# Amino acid sequences can be character strings
aaString <- "ACDEFGHIKLMNPQRSTVWY"

# Amino acid sequences can also be character vectors
aaVector <- c("A", "C", "D", "E", "F",
               "G", "H", "I", "K", "L",
               "M", "N", "P", "Q", "R",
               "S", "T", "V", "W", "Y")

# Alternatively, .fasta files can also be used by providing
## The path to the file as a character string
## Not run:
sequenceCheck(aaString)
sequenceCheck(aaVector)

# To allow O and U
sequenceCheck(aaString,
             nonstandardResidues = c("O", "U"),
             suppressAAWarning = TRUE)

# To turn off output message
sequenceCheck(aaString,
             suppressOutputMessage = TRUE)

# To change string to be a vector
sequenceCheck(aaString,
             outputType = "vector")

# To not return a sequence but check the input
sequenceCheck(aaVector,
             outputType = "none")

## End(Not run)
sequenceMap

Description

This is a graphical function used to visualize data along an amino acid sequence. The purpose of this function is to show the entire sequence and color residues based on properties. This may help identify important residues along a protein. This was designed with the goal of visualizing discrete values, but has since been expanded to visualize numeric/continuous values.

Usage

sequenceMap(
  sequence,
  property,
  nbResidues = 30,
  labelType = "both",
  everyN = c(1, 10),
  labelLocation = c("on", "below"),
  rotationAngle = c(0, 0),
  customColors = NA
)

Arguments

sequence amino acid sequence as a single character string, a vector of single characters, or an AAString object. It also supports a single character string that specifies the path to a .fasta or .fa file.

property a vector with length equal to sequence length. This is what is visualized on the function. Can be discrete or continuous values.

nbResidues numeric value, 30 by default. The number of residues to display on each row of the plot. It is not recommended to be over 50 or under 10 for standard sequences. Optimal value may vary between sequences of extreme lengths.

labelType character string, "both" by default. accepted values are labelType = c("both", "AA", "number", "none"). "both" shows both amino acid residue and residue number. "AA" and "number" show either the amino acid residue or the residue number, respectively. "none" only shows graphical values without labels. NOTE: When using "both", *everyN*, *labelLocation*, and *rotationAngle* all require vectors of length = 2 where the first value applies to the "AA" parameter and the second value applies to the "number" parameter. When using "AA" or "number", *everyN*, *labelLocation*, and *rotationAngle* require a single value. If a vector is provided, only the first value will be used.

everyN numeric value or vector of numeric values with length = 2. This is used to show every Nth amino acid and/or residue number. To show every value, set everyN = 1 or everyN = c(1, 1).

labelLocation character string or vector of character strings with length = 2. When labelLocation = "on", the text is layered on top of the graphical output. When labelLocation = "below", the text is placed below the graphical output. If labelType = "both", do not set labelLocation = c("on", "on") or labelLocation = c("below", "below").
sequenceMap

rotationAngle numeric value or vector of numeric values with length = 2. This value is used to rotate text. Especially useful when printing many residue numbers.

customColors vector of colors as character strings. NA by default. Used to support custom plot colors. If property is a discrete scale, a character vector of colors with length = number of unique discrete observations is required. If property is a continuous scale, a character vector of the colors for c("highColor","lowColor","midColor"). Set NA to skip custom colors.

Value

A ggplot.

See Also

sequenceMapCoordinates for mapping coordinates

Examples

#Get a data frame returned from another function
aaVector <- c("A", "C", "D", "E", "F",
"G", "H", "I", "K", "L",
"M", "N", "P", "Q", "R",
"S", "T", "V", "W", "Y")

## As a continuous property
exampleDF_cont <- chargeCalculationGlobal(sequence = aaVector)
head(exampleDF_cont)

## Or as a discrete property
exampleDF_disc <- structuralTendency(sequence = aaVector)
head(exampleDF_disc)

sequenceMap(sequence = exampleDF_cont$AA,
property = exampleDF_cont$Charge,
nbResidues = 3,
labelType = "both")

sequenceMap(sequence = exampleDF_disc$AA,
property = exampleDF_disc$Tendency,
nbResidues = 3,
labelType = "both")

#Change the layout of labels
sequenceMap(
sequence = exampleDF_disc$AA,
property = exampleDF_disc$Tendency,
nbResidues = 3,
latype = "number") #Only residue number labels

sequenceMap(
sequence = exampleDF_disc$AA,
property = exampleDF_disc$Tendency,
nbResidues = 3,
latype = "number") #Only residue number labels
sequenceMap(
  sequence = exampleDF_disc$AA,
  property = exampleDF_disc$Tendency,
  nbResidues = 3,
  labelType = "none")  # No labels

# The text can also be rotated for ease of reading, especially helpful for larger sequences.
sequenceMap(
  sequence = exampleDF_disc$AA,
  property = exampleDF_disc$Tendency,
  labelType = "number",
  labelLocation = "on",
  rotationAngle = 90)

# Specify colors for continuous values
sequenceMap(
  sequence = exampleDF_cont$AA,
  property = exampleDF_cont$Charge,
  customColors = c("purple", "pink", "grey90"))

# or discrete values
sequenceMap(
  sequence = exampleDF_disc$AA,
  property = exampleDF_disc$Tendency,
  customColors = c("#999999", "#E69F00", "#56B4E9"))

# Change the number of residues on each line with nbResidues
# or discrete values
sequenceMap(
  sequence = exampleDF_disc$AA,
  property = exampleDF_disc$Tendency,
  nbResidues = 1)
sequenceMap(
  sequence = exampleDF_disc$AA,
  property = exampleDF_disc$Tendency,
  nbResidues = 3)
sequenceMap(
  sequence = exampleDF_disc$AA,
  property = exampleDF_disc$Tendency,
  nbResidues = 10)

# Use sequenceMapCoordinates for additional annotations
gg <- sequenceMap(sequence = exampleDF_disc$AA,
  property = exampleDF_disc$Tendency,
  nbResidues = 3,
  labelType = "both")

# Change the nbResidues to correspond to the sequenceMap setting
mapCoordDF <- sequenceMapCoordinates(aaVector,
sequenceMapCoordinates

Description

This is a function used to create a coordinate grid for the sequenceMap function. It is based on the length of the sequence being mapped, and how many residues per line are specified. The function wraps the sequence to have a number of columns that is the sequence length / number of residues per row, rounded up.

This is intended for use within the sequenceMap function, however, this can also be used to identify the coordinates of residues within the ggplot coordinate plane for addition annotations.

Usage

sequenceMapCoordinates(sequence, nbResidues = 30)

Arguments

- **sequence**: amino acid sequence as a single character string, a vector of single characters, or an AAString object. It also supports a single character string that specifies the path to a .fasta or .fa file.
- **nbResidues**: numeric value, 30 by default. The number of residues to display on each row of the plot. It is not recommended to be over 50 or under 10 for standard sequences. Optimal value may vary between sequences of extreme lengths.

Value

A data frame with rows containing the amino acid sequence, residue position within the sequence, as well as the row and column of each residue within the ggplot output of sequenceMap().
See Also

`sequenceMapCoordinates` for mapping coordinates

Examples

```r
# Amino acid sequences can be character strings
aaString <- c("ACDEFGHIKLMNPQRSTVWY")
# Amino acid sequences can also be character vectors
aaVector <- c("A", "C", "D", "E", "F", 
"G", "H", "I", "K", "L", 
"M", "N", "P", "Q", "R", 
"S", "T", "V", "W", "Y")
# Alternatively, .fasta files can also be used by providing 
The path to the file as a character string
exampleDF <- sequenceMapCoordinates(aaString,
                                       nbResidues = 10)
head(exampleDF)
exampleDF <- sequenceMapCoordinates(aaVector,
                                       nbResidues = 10)
head(exampleDF)

# Getting a data frame for plotting with sequenceMap
tendencyDF <- structuralTendency(sequence = aaVector)
# Making a sequenceMap ggplot to annotate
gg <- sequenceMap(sequence = tendencyDF$AA,
                   property = tendencyDF$Tendency,
                   nbResidues = 3,
                   labelType = "both")

# Change the nbResidues to correspond to the sequenceMap setting
mapCoordDF <- sequenceMapCoordinates(aaVector,
                                       nbResidues = 3)
head(mapCoordDF)

# Subsetting for positive residues
mapCoordDF_subset <- mapCoordDF$AA %in% c("K", "R", "H")
mapCoordDF_subset <- mapCoordDF[mapCoordDF_subset,]

# Use mapCoordDF to annotate positive residues with a plus
library(ggplot2)
gg <- gg + geom_point(inherit.aes = FALSE,
                       data = mapCoordDF_subset,
                       aes(x = col + 0.5, # to center on the residue
                            y = row + 0.2), # to move above on the residue
                       color = "purple",
                       size = 3,
                       shape = 3)
plot(gg)
```
sequencePlot

Description

This is a graphical function used to visualize numeric data along an amino acid sequence.

Usage

```r
sequencePlot(
  position,
  property,
  hline = NA,
  propertyLimits = NA,
  dynamicColor = NA,
  customColors = NA,
  midpoint = hline,
  customTitle = NA
)
```

Arguments

- **position**: numeric vector of residue positions. Typically `c(1 : sequenceLength)`. i.e. a sequence with 215 amino acids has a vector of values 1 to 215. This is the X axis.
- **property**: vector of values, typically numeric. Equal in length to position. This is the Y axis.
- **hline**, **propertyLimits**: optional, numeric values or numeric vectors. Prints horizontal lines. Set to NA to skip (default). *hline* specifies the location for a dashed, grey line to be printed underneath the plot's data line. Good for separating cutoff values. *propertyLimits* specifies the location for a solid, black line to be printed. Good for showing maximum and minimum values.
- **dynamicColor**: optional vector. Typically numeric. Equal in length to position. Can be used to set colors based on values. Can be categorical (discrete) or continuous. Set to NA to skip (default).
- **customColors**: optional vector of colors as character strings. Used to support custom plot colors. If property is a discrete scale, a character vector of colors with length = number of unique discrete observations is required. If property is a continuous scale, a character vector of the colors for c("highColor","lowColor","midColor"). Set NA to skip custom colors (default). Ignored if dynamicColor = NA.
- **midpoint**: needed for proper scales of customColors. The default value is equal to hline (if provided). If there is no hline, the average of propertyLimits is the midpoint (if provided). If neither is provided, the value will be NA. The user can explicitly assign the midpoint to avoid this or to overwrite the defaults.
- **customTitle**: optional, character string. Allows adding custom title. Set to NA to skip (default).
structuralTendency

### Description

Each amino acid residue has a tendency to impact the order / disorder of the amino acid sequence. Some residues are disorder promoting, meaning they tend to favor disorder over ordered structures. These are typically hydrophilic, charged, or small residues. Order promoting residues tend to be aliphatic, hydrophobic, aromatic, or form tertiary structures. Disorder neutral residues neither favor order nor disordered structures.

### Usage

```r
structuralTendency(
  sequence,
```
Arguments

sequence  amino acid sequence as a single character string, a vector of single characters, or an AAString object. It also supports a single character string that specifies the path to a .fasta or .fa file.

disorderPromoting, disorderNeutral, orderPromoting  character vectors of individual residues to be matched with the input sequence. Defaults:
  • disorderPromoting = c("P", "E", "S", "Q", "K", "A", "G")
  • disorderNeutral = c("D", "T", "R")

It is not recommended to change these. These definitions are from Uversky (2013).

printCitation  logical, FALSE by default. When printCitation = TRUE, a citation to Uversky, V. N. (2013) is printed. This is the paper categorizing the structural impact of each residue that is used as the default settings.

Value

a data frame containing each residue from the sequence matched with its structural tendency, defined by disorderPromoting, disorderNeutral, and orderPromoting. For convenient plotting see structuralTendencyPlot.

References


See Also

Other structural tendency: structuralTendencyPlot()

Examples

#Amino acid sequences can be character strings
aaString <- "ACDEFGHIKLMNPQRSTVWY"
#Amino acid sequences can also be character vectors
aaVector <- c("A", "C", "D", "E", "F",
              "G", "H", "I", "K", "L”,
              "M", "N", "P", "Q", "R",
              "S", "T", "V", "W", "Y")
Alternatively, .fasta files can also be used by providing the path to the file as a character string.

```
exampleDF <- structuralTendency(aaString)
head(exampleDF)
exampleDF <- structuralTendency(aaVector)
head(exampleDF)
```

This example shows if a user changes the default definition of residues. These residues are labeled as such from Dunker et al (2001), "Intrinsically disordered protein."

```
exampleDF <- structuralTendency(aaString,
    disorderNeutral = c("H", "M", "T", "D"),
    orderPromoting = c("W", "C", "F", "I", "Y", "V", "L", "N"))
head(exampleDF)
```

---

**structuralTendencyPlot**

**Plotting Structural Tendency of Amino Acid Sequence**

**Description**

Convenient graphing for the `structuralTendency` function.

**Usage**

```
structuralTendencyPlot(
    sequence,
    graphType = "pie",
    summarize = FALSE,
    proteinName = NA,
    alphabetical = FALSE,
    disorderNeutral = c("D", "T", "R"),
    ...
)
```

**Arguments**

- `sequence`: amino acid sequence (or pathway to a fasta file) as a character string. Supports multiple sequences / files, as a character vector of strings.
- `graphType`: character string, required. `graphType` must be set to c("pie", "bar", "none"). When `graphType = "pie"`, the output is a pie chart. When `graphType = "bar"`, the output is a bar chart. When `graphType = "none"`, the output is the data frame that would otherwise be used to plot the data.
**summarize** logical value, FALSE by default. When summarize = TRUE, each residue is aggregated into Disorder Tendency Groups. (See `structuralTendency` for more details). When summarize = FALSE, residue identity is preserved, and the output is colored by Disorder Tendency Groups.

**proteinName** optional character string. NA by default. Used to either add the name of the protein to the plot title.

**alphabetical** logical value, FALSE by default. Order of residues on plot axis. Only relevant when summarize = FALSE, otherwise is ignored. If FALSE, ordering is grouped by Disorder Tendency (P, E, S, ..., W, C). If TRUE, the residues are ordered alphabetically (A, C, D, E, ..., W, Y).

**disorderPromoting, disorderNeutral, orderPromoting** character vectors of individual residues to be matched with the input sequence. Defaults:

- disorderNeutral = c("D", "T", "R")

It is not recommended to change these.

... additional arguments to be passed to `structuralTendency` and `ggplot`

**Value**

a data frame containing each residue from the sequence matched with its structural tendency, defined by disorderPromoting, disorderNeutral, and orderPromoting.

**Plot Colors**

For users who wish to keep a common aesthetic, the following colors are used when graphType = "bar" or "pie"

- Disorder Neutral = "#F0B5B3"
- Disorder Promoting = "darkolivegreen3" or "#A2CD5A"
- Order Promoting = "darkorchid1" or "#BF3EFF"

**References**


**See Also**

Other structural tendency: `structuralTendency()`
Examples

# Amino acid sequences can be character strings
aaString <- "ACDEFGHIKLMNPQRSTVWY"

# Amino acid sequences can also be character vectors
aaVector <- c("A", "C", "D", "E", "F",
              "G", "H", "I", "K", "L",
              "M", "N", "P", "Q", "R",
              "S", "T", "V", "W", "Y")

# Alternatively, .fasta files can also be used by providing
## The path to the file as a character string

structuralTendencyPlot(aaString)
structuralTendencyPlot(aaVector)

# The plot can be a pie chart (default)
structuralTendencyPlot(aaString,
                        graphType = "pie")

# Or the plot can be a bar graph
structuralTendencyPlot(aaString,
                        graphType = "bar")

# To display general tendency rather than residues, set summarize = T
structuralTendencyPlot(aaString,
                        graphType = "pie",
                        summarize = TRUE)

structuralTendencyPlot(aaString,
                        graphType = "bar",
                        summarize = TRUE)

# If you wish to export this as a dataframe, set graphType = "none"
exampleDF <- structuralTendencyPlot(aaString,
                                      graphType = "none")

head(exampleDF)

# If using a different definition of disordered residues
## These residues are labeled as such from Dunker et al (2001),
## "Intrinsically disordered protein."

structuralTendencyPlot(aaString,
                        disorderNeutral = c("H", "M", "T", "D"),
                        orderPromoting = c("W", "C", "F", "I", "Y", "V", "L", "N"),
                        graphType = "bar",
                        alphabetical = TRUE)
**Description**

This is a vector of sequences as character strings. This contains the amino acid sequence of Human Cellular tumor antigen p53 (UniProt ID: P04637) and sequences of several homologous sequences. These sequences in TP53Sequences selected due to their highly similar identity on UniProt (The UniProt Consortium, 2019).

**Usage**

TP53Sequences

GorillaTP53

**Format**

An object of class character of length 9.

An object of class character of length 1.

**UniProt IDs**

- P02340
- P04637
- P10361
- Q29537
- Q00366
- O09185
- Q9TTA1
- Q95330
- A0A2I2Y7Z8*

* The Gorilla p53 sequence is not within the TP53Sequences vector, and is its own object named GorillaTP53. This is because the Gorilla p53 is unreviewed on UniProt, so we chose to exclude it from the list of otherwise SwissProt sequences.

**Source**

Index

* Example Sequences
  TP53Sequences, 48

* IDP-based Substitution Matrices
  DisorderMat, 10
  DUNMat, 12
  EDSSMat, 13

* charge functions
  chargeCalculationGlobal, 2
  chargeCalculationLocal, 5
  hendersonHasselbalch, 17
  netCharge, 28

* datasets
  DisorderMat, 10
  DUNMat, 12
  EDSSMat, 13
  KDNorm, 26
  pKaData, 30
  TP53Sequences, 48

* scaled hydropathy functions
  foldIndexR, 15
  KDNorm, 26
  meanScaledHydropathy, 27
  scaledHydropathyGlobal, 32
  scaledHydropathyLocal, 34

* structural tendency
  structuralTendency, 44
  structuralTendencyPlot, 46

  chargeCalculationGlobal, 2, 7, 18, 29
  chargeCalculationLocal, 4, 5, 18, 19, 21, 22, 29
  chargeHydropathyPlot, 8, 19, 21, 22

  Disorder40 (DisorderMat), 10
  Disorder60 (DisorderMat), 10
  Disorder85 (DisorderMat), 10
  DisorderMat, 10, 13, 15
  DUNMat, 12, 12, 15
  EDSSMat, 11–13, 13

EDSSMat50 / EDSSMat, 13
EDSSMat60 / EDSSMat, 13
EDSSMat62 / EDSSMat, 13
EDSSMat70 / EDSSMat, 13
EDSSMat75 / EDSSMat, 13
EDSSMat80 / EDSSMat, 13
EDSSMat90 / EDSSMat, 13
foldIndexR, 15, 19, 22, 27, 33, 35

ggplot, 9, 47
GorillaTP53 (TP53Sequences), 48

hendersonHasselbalch, 4, 7, 17, 17, 29, 32

idpr, 18
idpr::meanScaledHydropathy(), 9
idpr::netCharge(), 9
idprofile, 19
iupred, 19, 20, 22, 23
iupredAnchor, 19, 20, 22
iupredAnchor (iupred), 23
iupredRedox, 19, 20, 22
iupredRedox (iupred), 23

KDNorm, 17, 26, 27, 33, 35

meanScaledHydropathy, 9, 17, 27, 33, 35

netCharge, 4, 7, 9, 18, 28

pKaData, 4, 7, 17, 18, 21, 29, 30

scaledHydropathyGlobal, 17, 27, 32, 35
scaledHydropathyLocal, 17, 19, 21, 22, 27, 33, 34

sequenceCheck, 36
sequenceMap, 37, 41
sequenceMapCoordinates, 39, 41, 42
sequencePlot, 43

structuralTendency, 44, 46, 47
structuralTendencyPlot, 44, 46, 47

TP53Sequences, 48