Package 'QuartPAC'

September 30, 2024

Type Package

Title Identification of mutational clusters in protein quaternary structures

Version 1.36.0

Date 2024-04-23

Author Gregory Ryslik, Yuwei Cheng, Hongyu Zhao

Maintainer Gregory Ryslik <gregory.ryslik@yale.edu>

Description Identifies clustering of somatic mutations in proteins over the entire quaternary structure.

License GPL-2

biocViews Clustering, Proteomics, SomaticMutation

Depends iPAC, GraphPAC, SpacePAC, data.table

Imports Biostrings, pwalign

Suggests RUnit, BiocGenerics, rgl

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

git_branch RELEASE_3_19

git_last_commit 19f8acc

git_last_commit_date 2024-04-30

Repository Bioconductor 3.19

Date/Publication 2024-09-29

Contents

QuartPAC-package *Identifying mutational clusters while incorporating protein quaternary structure.*

Description

QuartPAC is a companion package to **iPAC**, **GraphPAC** and **iPAC**. It allows one to use the methodologies proposed by each of those packages to be applied to the protein quaternary structure.

Details

QuartPAC is designed to identify mutational clustering in 3D space when looking at the entire quaternary protein structure. It does this by applying the algorithms proposed in **iPAC**, **GraphPAC** and **SpacePAC** over the entire assembly after correctly preprocessing the mutational and positional data. QuartPAC typically follows a three step process. Step 1 involves reading the mutational data - see `getMutations` for more information. Step 2 involves reading in the structural information to create the protein assembly and is explained in `makeAlignedSuperStructure`. Finally, Step 3 performs the statistical analysis and reports the significant p-values – see `QuartCluster` for more information.

The clustering results give the *serial number* values from the *.pdb1 file.

Author(s)

Gregory Ryslik, Yuwei Cheng, Hongyu Zhao. Maintainer: Gregory Ryslik <gregory.ryslik@yale.edu>

References

Gregory Ryslik and Hongyu Zhao (2012). iPAC: Identification of Protein Amino acid Clustering. R package version 1.8.0. http://www.bioconductor.org/.

Gregory Ryslik and Hongyu Zhao (2013). GraphPAC: Identification of Mutational Clusters in Proteins via a Graph Theoretical Approach. R Package version 1.6.0 http://www.bioconductor.org/.

Gregory Ryslik and Hongyu Zhao (2013). SpacePAC: Identification of Mutational Clusters in 3D Protein Space via Simulation. R package version 1.2.0. http://www.bioconductor.org/.

The UniProt Consortium. Activities at the Universal Protein Resource (UniProt). Nucleic Acids Res. 42: D191-D198 (2014).

Examples

```
#read the mutational data
mutation_files <- list(
        system.file("extdata","HFE_Q30201_MutationOutput.txt", package = "QuartPAC"),
        system.file("extdata","B2M_P61769_MutationOutput.txt", package = "QuartPAC")
)
uniprots <- list("Q30201","P61769")
mutation.data <- getMutations(mutation_files = mutation_files, uniprots = uniprots)

#read the pdb file
```

```
pdb.location <- "https://files.rcsb.org/view/1A6Z.pdb"
assembly.location <- "https://files.rcsb.org/download/1A6Z.pdb1"
structural.data <- makeAlignedSuperStructure(pdb.location, assembly.location)

#Perform Analysis
#We use a very high alpha level here with no multiple comparison adjustment
#to make sure that each method provides shows a result.
#Lower alpha cut offs are typically used.
(quart_results <- quartCluster(mutation.data, structural.data, perform.ipac = "Y", perform.graphpac = "Y",
                               perform.spacepac = "Y", create.map = "N", MultComp = "None",
                         alpha = .3, radii.vector = c(1:3), show.low.level.messages = "Y"))
```

`getMutations` *Get Mutational Data*

Description

Reads the mutation matrices and fasta information for each protein subunit within the quaternary structure.

Usage

```
getMutations(mutation_files, uniprots)
```

Arguments

`mutation_files`	A list of strings where each string is the path to a mutation matrix. A mutation matrix is a matrix of 0's (no mutation) and 1's (mutation). Each column represents a specific amino acid in the protein and each row represents an individual sample (test subject, cell line, etc). If column i in row j had a 1, that would mean that the ith amino acid for person j had a nonsynonomous mutation. As the quaternary structure can be comprised of several proteins (each with their unique uniprot id), each matrix represents the protein referenced by a specific uniprot identifier.
`uniprots`	A list of uniprots. The list provides the uniprot id for each of the matrices described in the *mutation.data* parameter.

Details

The ordering in both *mutation_files* and *uniprots* must be the identical. For example, suppose that the quaternary structure is comprised of two proteins, A and B. If the first element in *mutation_files* points to the mutation matrix for protein A, that means that the first element of *uniprots*, must be a string with the uniprot id of protein A.

Value

`mut_tables` A list of the mutation matrices. There should be one mutation matrix for each uniprot id in the entire assembly.

`uniprots` The uniprot ids for each of the mutation matrices. The uniprot id's are shown in the same order as the mutation matrices.

`aa_counts` The number of amino acids for each uniprot. This corresponds to the number of columns in the mutation matrix that is provided as input.

`canonical_lengths` The length of the protein as shown in the uniprot database. The uniprot ID must be available on uniprot.org.

Note

To see examples of mutation matrices, please look in the /emphextdata folder of the package.

References

The UniProt Consortium. *Activities at the Universal Protein Resource (UniProt)*. Nucleic Acids Res. 42: D191-D198 (2014).

Examples

```
mutation_files <- list(
system.file("extdata","HFE_Q30201_MutationOutput.txt", package = "QuartPAC"),
          system.file("extdata","B2M_P61769_MutationOutput.txt", package = "QuartPAC")
)
uniprots <- list("Q30201","P61769")

(mutation.data <- getMutations(mutation_files = mutation_files, uniprots = uniprots))
```

---

makeAlignedSuperStructure

Create Protein Assembly Information

Description

Reads the information in the PDB files to build the quaternary structure.

Usage

```
makeAlignedSuperStructure(PDB_location, Assembly_location)
```

Arguments

PDB_location	A string pointing to the location of the *.pdb file. The pdb file provides uniprot and structural information for each residue.
Assembly_location	A string pointing to the lcoation of the *.pdb1 file. The pdb1 file provides the positional information of all the residues in the structure when they are in the asembly.

Value

aligned_structure	The aligned structure. For each residue, you get the XYZ coordinate as well as other structural information. The absPos column in the *aligned_structure* variable matches the absPos column in the *raw_structure* and is used for debugging purposes. The *canonical_pos* column represents the residue number in the fasta sequence for the subunit with the specified uniprot id. The *canonical_pos* column can repeat. For instance, if there are two proteins, A and B, in the assembly, both proteins may have residue #5 in them. The *absPos* column will not repeat.
aa_table	A table showing the pairwise alignment between the residue sequence as specified by the pdb file and the residue sequence specified by the fasta sequence in the uniprot database. This table is mainly used for debugging.
raw_structure	The raw structure as read in from the *.pdb1 file after some cleaning. For instance, row's with NA for uniprot values are dropped. Further, only the carbon-alpha rows are kept.

Note

This method reads fasta information from www.uniprot.org. Thus an internet connection is required in order to execute properly.

This functionality is still in beta. The user is encouraged to check the alignment created by the method.

Examples

```
#read the pdb file
pdb.location <- "https://files.rcsb.org/view/1A6Z.pdb"
assembly.location <- "https://files.rcsb.org/download/1A6Z.pdb1"


(alignment.results <- makeAlignedSuperStructure(pdb.location, assembly.location))
```

---

quartCluster *Quaternary Protein Structure Clustering Analysis*

Description

Perform the clustering analysis over the protein quaternary structure. One can select to use the **iPAC**, **GraphPAC** or **SpacePAC** methods. The output will show the relevant information from each algorithm that was selected.

Usage

```
quartCluster(mutation_data, alignment, perform.ipac = "Y", perform.graphpac = "Y",
 perform.spacepac = "Y", insertion.type = "cheapest_insertion",
 MultComp = "Bonferroni", alpha = 0.05, show.low.level.messages = "N",
 ipac.method = "MDS", spacepac.method = "SimMax", create.map = "Y",
 Show.Graph = "Y", Graph.Output.Path = NULL, Graph.File.Name = "Map.pdf",
 Graph.Title = "Mapping", fix.start.pos = "Y", numsims = 1000,
 simMaxSpheres = 3, radii.vector = c(1, 2, 3, 4, 5, 6, 7, 8, 9, 10),
 OriginX = "", OriginY = "", OriginZ = "")
```

Arguments

`mutation_data`	The mutation data in the format outputted by `getMutations`.
`alignment`	The assembly structural information outputted by `makeAlignedSuperStructure`.
`perform.ipac`	Whether or not to perform the **iPAC** algorithm. Either a "Y" or a "N".
`perform.graphpac`	Whether or not to perform the **GraphPAC** algorithm. Either a "Y" or a "N".
`perform.spacepac`	Whether or not to perform the **SpacePAC** algorithm. Either a "Y" or a "N".
`insertion.type`	Specifies the type of insertion method used in the **GraphpAC** package. Please see the **GraphPAC** for more details.
`MultComp`	Specifies the multiple comparison adjustment required by the **iPAC** and **GraphPAC** packages. Options are: "Bonferroni", "BH", or "None". Please see the **iPAC** and **GraphPAC** packages for details.
`alpha`	The significance level required in order to find a mutational cluster significant using the **iPAC** and **GraphPAC** algorithms.
`show.low.level.messages`	Whether to display the output messages generated by the **iPAC**, **GraphPAC** and **SpacePAC** algorithms. Either a "Y" or a "N". Commonly used for debugging.
`ipac.method`	The type of approach used by **iPAC** to map the protein to 1D space. This parameter usually set to "MDS", but can be set to "linear" as well. See the **iPAC** package for more details.
`spacepac.method`	The type of approach used by **SpacePAC** to identify clustering. The options are either "SimMax" or "Poisson. This parameter usually set to "SimMax". See the **SpacePAC** package for more details.
`create.map`	Whether a graphical representation of the **iPAC** algorithm's dimension reduction from 3D to 1D space should be diplsayed. Either "Y" or "N".

`Show.Graph` Whether to show the **iPAC** package dimension reduction chart on the screen. Warning: You must be running R in a GUI environment, otherwise, an error will occur.

`Graph.Output.Path` Where to save the dimension reduction chart. This is useful if you want to save the chart automatically or can't display it on the screen (for instance, you are running R in a terminal window).The Graph.File.Name variable must be set as well.

`Graph.File.Name` If you would like the chart saved automatically to the disk, specify the output file name. The Graph.Output.Path variable must be set as well.

`Graph.Title` The title of the graph to be created.

`fix.start.pos` For the **GraphPAC** package, the heuristic solver for the traveling salesman problem starts the path at a random amino acid. In order to make the results easily reproducible, the default starts the path on the first amino acid in the protein. Please see the **GraphPAC** package for more details.

`numsims` The number of times to simulate the distribution of mutations over the protein quaternary structure for the **SpacePAC** algorithm. For each simulation, given m total mutations and n total amino acids, each amino acid has a m/n probability of mutation.

`simMaxSpheres` For the **SpacePAC** algorithm, the maximum number of spheres to consider. Currently, the implementation allows for simMaxspheres to be either 1, 2 or 3.

`radii.vector` This applies to the **SpacePAC** algorithm and denotes the vector of radii that will be considered. At each sphere radius, the best sphere combination is found. See the *SpaceClust* method in the **SpacePAC** package for further details

`OriginX` If the "Linear" method is chosen for the **iPAC** algorithm, this specifies the x-coordinate part of the fixed point. See the vignette in the **iPAC** package for more details.

`OriginY` If the "Linear" method is chosen for the **iPAC** algorithm, this specifies the y-coordinate part of the fixed point. See the vignette in the **iPAC** package for more details.

`OriginZ` If the "Linear" method is chosen for the **iPAC** algorithm, this specifies the z-coordinate part of the fixed point. See the vignette in the **iPAC** package for more details.

Value

`ipac` The clustering results using the **iPAC** algorithm. See the **iPAC** packagee for more details of each sub item.

`graphpac` The clustering results using the **GraphPAC** algorithm. See the **GraphPAC** package for more details of each sub item.

`spacepac` The clustering results using the **SpacePAC** algorithm. See the **SpacePAC** package for more details of each sub item.

`ipac_messages` Any messages that might of been reported by the **iPAC** algorithm. Typically, warning or error messages are displayed here.

graphpac_messages
: Any messages that might of been reported by the **GraphPAC** algorithm. Typically, warning or error messages are displayed here.

spacepac_messages
: Any messages that might of been reported by the **SpacePAC** algorithm. Typically, warning or error messages are displayed here.

Note

The clustering results give the *serial number* values from the *.pdb1 file.

Most of the parameters simply pass the requisite values to the underlying **iPAC**, **GraphPAC** and **SpacePAC** algorithms. The user should be aware of the parameters for these algorithms as this package is designed to extend them to quaternary structures.

References

Gregory Ryslik and Hongyu Zhao (2012). iPAC: Identification of Protein Amino acid Clustering. R package version 1.8.0. Gregory Ryslik and Hongyu Zhao (2012). GraphPAC: Identification of Mutational Clusters in Proteins via a Graph Theoretical Approach.. R package version 1.6.0. Gregory Ryslik and Hongyu Zhao (2013). SpacePAC: Identification of Mutational Clusters in 3D Protein Space via Simulation.. R package version 1.2.0. Michael Hahsler and Kurt Hornik (2014). TSP: Traveling Salesperson Problem (TSP). R package version 1.0-9. http://CRAN.R-project.org/package=TSP

Examples

```
#read the mutational data
mutation_files <- list(
        system.file("extdata","HFE_Q30201_MutationOutput.txt", package = "QuartPAC"),
        system.file("extdata","B2M_P61769_MutationOutput.txt", package = "QuartPAC")
)
uniprots <- list("Q30201","P61769")
mutation.data <- getMutations(mutation_files = mutation_files, uniprots = uniprots)

#read the pdb file
pdb.location <- "https://files.rcsb.org/view/1A6Z.pdb"
assembly.location <- "https://files.rcsb.org/download/1A6Z.pdb1"
structural.data <- makeAlignedSuperStructure(pdb.location, assembly.location)
## Not run:
#Perform Analysis
#We use a very high alpha level here with no multiple comparison adjustment
#to make sure that each method provides shows a result.
#Lower alpha cut offs are typically used.
(quart_results <- quartCluster(mutation.data, structural.data, perform.ipac = "Y", perform.graphpac = "Y",
                               perform.spacepac = "Y", create.map = "N", MultComp = "None",
                          alpha = .3, radii.vector = c(1:3), show.low.level.messages = "Y"))

## End(Not run)
```

Index