Package ‘GraphPAC’

April 14, 2017

Type Package

Title Identification of Mutational Clusters in Proteins via a Graph Theoretical Approach.

Version 1.16.0

Date 2012-12-9

Author Gregory Ryslik, Hongyu Zhao

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

Description Identifies mutational clusters of amino acids in a protein while utilizing the proteins tertiary structure via a graph theoretical model.

License GPL-2

Depends R(>= 2.15), iPAC, igraph, TSP, RMallow

Suggests RUnit, BiocGenerics

Repository Bioconductor

biocViews Clustering, Proteomics

NeedsCompilation no

R topics documented:

GraphPAC-package ................................................................. 1
Find.TSP.Path ................................................................. 3
GraphClust ................................................................. 4
Plot.Protein ................................................................. 6

Index 8

GraphPAC-package Using Graph Theory to Identify Mutational Clusters of Amino Acids.
GraphPAC-package

Description

The GraphPAC package identifies statistically significant clusters of non-synonymous amino acid mutations and is a sister package to iPAC. GraphPAC reorders the protein into a one dimensional space via a graph theoretical approach. Specifically, the traveling salesman problem (TSP) is solved heuristically via the TSP package. Once solved, the mutational data is reordered to follow the hamiltonian path and the nmc algorithm is run to find the mutational clusters on the remapped protein. Unlike the MDS remapping approach that is used in iPAC, distant amino acids no longer have an effect on each other’s position in one dimensional space allowing for a closer representation of the underlying biology.

Details

Please see the documentation for “get.Positions”, “get.AlignedPositions”, and “Plot-Protein.Linear” in the iPAC package. There you will find information on getting basic positional data and plotting functions.

Author(s)

Gregory Ryslik Hongyu Zhao
Maintainer: Gregory A. Ryslik <gregory.ryslik@yale.edu>

References


Examples

```r
## Not run:
#Load the positional and mutational data
CIF<"http://www.pdb.org/pdb/files/3GFT.cif"
Fasta<"http://www.uniprot.org/uniprot/P01116-2.fasta"
KRAS.Positions<get.Positions(CIF,Fasta, "A")
data(KRAS.Mutations)

#Calculate the required clusters
GraphClust(KRAS.Mutations,KRAS.Positions$Positions,insertion.type = "cheapest_insertion", alpha = 0.05, MultComp = "Bonferroni")

## End(Not run)
```
**Find.TSP.Path**

**Description**

Employs a heuristic approach to solve the traveling salesman problem.

**Usage**

```r
Find.TSP.Path(PositionList, mutation.matrix, insertion.type = "cheapest_insertion", fix.start.pos = "Y")
```

**Arguments**

- `PositionList`: A dataframe consisting of six columns: 1) Residue Name, 2) Amino Acid number in the protein, 3) Side Chain, 4) X-coordinate, 5) Y-coordinate and 6) Z-coordinate. Please see `get.Positions` and `get.AlignedPositions` in the iPAC package for further information on how to construct this matrix.

- `mutation.matrix`: A matrix of 0’s (no mutation) and 1’s (mutation) where each column represents an amino acid in the protein and each row represents an individual sample (test subject, cell line, etc). Thus if column i in row j had a 1, that would mean that the ith amino acid for person j had a nonsynonomous mutation.

- `insertion.type`: Specifies the type of insertion method used. Please see the TSP package for more details.

- `fix.start.pos`: The TSP package starts the path at a random amino acid. Such that the results are easily reproducible, the default starts the path on the first amino acid in the protein.

**Value**

- `candidate.path`: A numeric vector of the sequence found through the protein.

- `candidate.path.distance`: The distance traveled along the candidate path.

- `dist.matrix`: The distance matrix between any two pairwise amino acids.

- `linear.path.distance`: The distance traveled if one were to visit the amino acids in the original sequence (1 -> 2 -> 3 -> ...->N)

**References**


Examples

#Load the position and mutational data
CIF<="http://www.pdb.org/pdb/files/3GFT.cif"
Fasta<="http://www.uniprot.org/uniprot/P01116-2.fasta"
KRAS.Positions<-get.Positions(CIF,Fasta, "A")
data(KRAS.Mutations)

#Save all the results to path.results
path.results <- Find.TSP.Path(KRAS.Positions$Positions, KRAS.Mutations)

GraphClust

Description

Finds mutational clusters after reordering the protein using the traveling salesman approach.

Usage

GraphClust(mutation.data, position.data, insertion.type = "cheapest_insertion", alpha = 0.05,
MultComp = "Bonferroni", fix.start.pos = "Y", Include.Culled = "Y",
Include.Full = "Y")

Arguments

mutation.data A matrix of 0’s (no mutation) and 1’s (mutation) where each column represents an amino acid in the protein and each row represents an individual sample (test subject, cell line, etc). Thus if column i in row j had a 1, that would mean that the ith amino acid for person j had a nonsynonomous mutation.

position.data A dataframe consisting of six columns: 1) Residue Name, 2) Amino Acid number in the protein, 3) Side Chain, 4) X-coordinate, 5) Y-coordinate and 6) Z-coordinate. Please see get.Positions and get.AlignedPositions in the iPAC package for further information on how to construct this matrix.

insertion.type Specifies the type of insertion method used. Please see the TSP package for more details.

alpha The significance level required in order to find a mutational cluster significance. Please see the NMC package for further information.

MultComp The multiple comparison adjustment required as all pairwise mutations are considered. Options are: “Bonferroni”, “BH”, or “None”.

fix.start.pos The TSP package starts the path at a random amino acid. Such that the results are easily reproducible, the default starts the path on the first amino acid in the protein.

Include.Culled If "Y", the standard NMC algorithm will be run on the protein after removing the amino acids for which there is no positional data.

Include.Full If "Y", the standard NMC algorithm will be run on the full protein sequence.
Details

The protein reordering is done using the TSP package available on CRAN. This hamiltonian path then serves as the new protein ordering.

The position data can be created via the “getAlignedPositions” or the “getPositions” functions available via the imported iPAC package.

The mutation matrix must have the default R column headings “V1”, “V2”,.....“VN”, where N is the last amino acid in the protein. No positions should be skipped in the mutation matrix.

When unmapping back to the original space, the end points of the cluster in the mapped space are used as the endpoints of the cluster in the unmapped space.

Value

- Remapped: This shows the clusters found while taking the 3D structure into account and remapping the protein using a traveling salesman approach.
- OriginalCulled: This shows the clusters found if you run the NMC algorithm on the canonical linear protein, but with the amino acids for which we don’t have 3D positional data removed.
- Original: This shows the clusters found if you run the NMC algorithm on the canonical linear protein with all the amino acids.
- candidate.path: This shows the path found by the TSP package that heuristically minimizes the total distance through the protein.
- path.distance: The length of the candidate path if traveled from start to finish.
- linear.path.distance: The length of the sequential path 1,2,3,...,N (where N is the total number of amino acids in the protein).
- protein.graph: A graph object created by the igraph package that has edges between amino acids on the candidate.path. This can be passed to plotting functions to create visual representations.
- missing.positions: This shows which amino acids are present in the mutation matrix but for which we do not have positions. These amino acids are cut from the protein when calculating the Remapped and OriginalCulled results.

References


Examples

```r
## Not run:
#Load the positional and mutatioanl data
CIF<"http://www.pdb.org/pdb/files/3GFT.cif"
```
Fasta<="http://www.uniprot.org/uniprot/P01116-2.fasta"
KRAS.Positions<-get.Positions(CIF,Fasta, "A")
data(KRAS.Mutations)

#Calculate the required clusters
GraphClust(KRAS.Mutations,KRAS.Positions$Positions,insertion.type = "cheapest_insertion",
alpha = 0.05, MultComp = "Bonferroni")

## End(Not run)

---

**Plot.Protein**

**Description**

Creates a circular interactive plot of the path through the protein.

**Usage**

```r
Plot.Protein(graph, path, vertex.size = 5, color.palette = "heat")
```

**Arguments**

- `graph`: The graph object returned by `GraphClust` ($protein.graph$).
- `path`: The path returned by `GraphClust` ($candidate.path$).
- `vertex.size`: How large you want each vertex to be.
- `color.palette`: Possible options are: "heat", "gray", "topo", "cm".

**Details**

This will plot the amino acids in a circular directed graph. The vertices can be dragged around to enhance the visual representation. This is meant to complement the `Plot.Protein.Linear` function in `iPAC` which is also applicable in this package.

**Note**

This function is based on the “tkplot” function in `igraph`. Please see the documentation for that package for the necessary requirements. Special thanks to Dr. Gábor Csárdí (creator of the `igraph` package) for his help.

**References**


Examples

## Not run:
#Loads the mutational and positional data
CIF<-"http://www.pdb.org/pdb/files/3GFT.cif"
Fasta<-"http://www.uniprot.org/uniprot/P01116-2.fasta"
KRAS.Positions<-get.Positions(CIF,Fasta, "A")
data(KRAS.Mutations)

#gets the cluster results and graph object
my.graph.clusters <- GraphClust(KRAS.Mutations,KRAS.Positions$Positions,
insertion.type = "cheapest_insertion",alpha = 0.05,
MultComp = "Bonferroni")

Plot.Protein(my.graph.clusters$protein.graph, my.graph.clusters$candidate.path,
vertex.size=5, color.palette="heat")

## End(Not run)
Index

*Topic **Amino Acids**
  Plot.Protein, 6

*Topic **Clusters**
  GraphClust, 4

*Topic **Graph**
  Find.TSP.Path, 3
  Plot.Protein, 6

*Topic **Mutations**
  GraphClust, 4

*Topic **Order**
  Plot.Protein, 6

*Topic **Traveling Salesman**
  Find.TSP.Path, 3

Find.TSP.Path, 3

GraphClust, 4
GraphPAC (GraphPAC-package), 1
GraphPAC-package, 1

Plot.Protein, 6