IgFamily v0.12.1h

Technical manual v0.2.0

Overview

Features

­­­­­

*For further information on features, refer to associated entries -*

**- FASTA utility tools:**

Read FASTA files into runtime. Custom FASTA creation utility provides functionality for tailored output files. This utlility can be used to create required FASTA format –

>[ACCESSION]|[NAME]|[TYPE]|[SPECIES]|

Accession field is a housekeeping field and is not required for runtime. Name, type, and species fields are used in data structure creation, association, and function.

|  |  |
| --- | --- |
| FASTA parse function | pp. |
| FASTA file utilities | pp. |

**- Peptide file compatibility:**

Read peptide files into runtime. Currently supported are *PEAKS v8.0 DE NOVO* de novo peptides .csv export, *PEAKS v8.0 SPIDER* protein peptide .csv export, and *NOVOR v1.1* de novo peptides .csv export.

*PEAKS v8.0 DE NOVO* de novo peptides peptide file assigns scan number, peptide with modification, and amino acid local confidence score. Note that here *PEAKS* *v8.0 DE NOVO* assigns individual export accessions to replicate peptide assignments.

*PEAKS v8.0 SPIDER* protein peptide file assigns scan number, peptide with modification, spectral count, and -10IgP certainty score.

*NOVOR v1.1* de novo peptides file assigns scan number, peptide with modification, and amino acid local confidence score. Note that here *NOVOR v1.1* assigns individual export accessions to replicate peptide assignments.

|  |  |
| --- | --- |
| Peptide file parse functions | pp. |

**- *msconvert* external integration**

*msconvert* is able to be called through a user defined interface option to convert .wiff and .wiff.scan files into the *Mascot Generic Format* .mgf file type. Various command line options may be selected with peak-picking as the default option. The generated file is created in the same folder as the input file.

|  |  |
| --- | --- |
| *msconvert* convert command line functions | pp. |
|  |  |

**- *NOVOR v1.1* external integration**

*NOVOR v1.1* is able to be called through a user defined interface option to generate *NOVOR v1.1* de novo peptide files in the .csv file type. Various command line options may be selected. The generated file is created in the same folder as the input file.

|  |  |
| --- | --- |
| *NOVOR v1.1* convert command line functions | pp. |

**- Local directory runtime functionality**

The *IgFamily* program is able to be run in local directory or filesystem directory file mode. In local directory file mode the user places the required files in the *IgFamily* root directory and executes the program. The input file and output files are moved to a folder created in the root directory with the name of the input file sample - The input file contains the sample name and is supported in the three input data types.

|  |  |
| --- | --- |
| Local directory runtimefunctions | pp. |

**- Data filesystem**

The *IgFamily* program is able to be run in local directory or filesystem directory file mode. In filesystem mode the program accessions a dedicated file association structure to retrieve and export files. The filesystem is currently defined on the *FATELVIS* network assisted storage device. The user is required to accession a file initially, however there is proposed functionality for dynamic file management with the *ISO/IEC TS 18822:2015* filesystem library.

|  |  |
| --- | --- |
| Data filesystem functions | pp. |

**- Runtime user interface**

On execution of the *IgFamily* program, the user is greeted with an interactive menu. The user is able to access FASTA file utilities, *msconvert* command line tools, *NOVOR v1.1* command line tools, and select program parameters. Program parameters include local or filesystem file modes, FASTA file selection, and peptide assignment selection.

|  |  |
| --- | --- |
| User interface functions | pp. |

**- FASTA data structuring**

Following user selection of runtime parameters, the *IgFamily* program will parse the FASTA file(s). The data are stored on contiguous RAM buffers at runtime. The [NAME], [TYPE], and [SPECIES] fields have runtime functionality, although the [ACCESSION] field is retained for FASTA utility functions. An additional data type is created to define those FASTA accessions that are immunoglobulin.

|  |  |
| --- | --- |
| FASTA data structuring functions | pp. |

**- Peptide data structuring**

Following user selection of runtime parameters, the *IgFamily* program will parse the data .csv file(s). The data are stored on contiguous RAM buffers at runtime. From this data, the raw peptide creates two additional data types for the peptide with modifications removed and the peptide truncated based on associated de novo local confidence scores. Truncation requires a moving average of 85% for amino acid local confidence and a minimum peptide length of 5. In the event of a peptide cleaved at a midpoint such that two peptides of length >= 5 are produced, both are assigned to unique data structures.

|  |  |
| --- | --- |
| Peptide data structuring functions | pp. |

**- External blastp integration**

Following runtime data structuring of FASTA and peptide data, a blastp reference database is created from the current FASTA data, and an input peptide query list is created from the current peptide data. The input peptide query list is tested against the blast reference database and a results file generated as an output file. The blastp output is parsed into runtime and assigned to a homology data structure. Data assigned are the blastp query, the blastp query alignment, the blastp subject, the blastp subject database accession, the blastp query alignment index, the blastp subject alignment index, and the blastp sequence alignment expectation value. Blastp is programmed to allow up to 200 matches for each query, and a generous threshold for alignment acceptance.

|  |  |
| --- | --- |
| Peptide data structuring functions | pp. |

**- Homology data structuring**

Generated and parsed homology data is runtime associated with respective FASTA and peptide data. Additional data types are created -

- Transformed expectation value: The e-value transformed as a metric of relative likelihood.

- Conjugated expectation value: The e-value transformed to consider overall family evidence.

- Homology parameter density: The density of the e-value (0<=par<=1) compared to all

|  |  |
| --- | --- |
| Peptide data structuring functions | pp. |

Narrative

**- *There’s method in models***

“A basic issue with interpreting model-based estimates is in knowing the meaning of parameters. There is no consensus about what a parameter means, however, because different people take different philosophical stances towards models. The perspective in this book is a common Bayesian perspective: Posterior probabilities of parameter values describe the relative compatibility of different states of the world with the data, according to the model. These are small world numbers. Reasonable people may disagree with the large world meaning, and the details of those disagreements depend strongly upon context. Such disagreements are productive, because they lead to model criticism and revision.”

The distinctiveness of a peptide can be modelled by relative sequence similarity. In particular, the likelihood of an amino acid substitution is determined largely by physicochemical properties. The Poisson distribution (Figure) is routinely used to model similarity. If sequence similarity is considered as a distance, with query sequences to a more similar subject match being spatially ‘closer’, then relative distance of a query sequence is approximated by a Poisson model. Strongly distinct peptides by definition are most likely derived from a single or a few proteins and as such the majority of the assignment likelihood is centred on those proteins, represented by a low Poisson rate factor. In contrast, peptides that are not able to be resolved from many proteins have their assignment likelihood shared and a high Poisson rate factor. As a note of technical interest, a high Poisson rate factor approaches a normal distribution *on average*. The BLAST (Basic Local Alignment Search Tool) is a primary utility for realising assignment likelihood for a peptide query in comparison to a protein subject database.



Figure - A Poisson distribution showing three representative scaling factors. Peptides with greater distinctiveness are likely to have a small scaling factor.

Although useful as a standalone tool, the BLAST algorithm is designed as an aid for peptide or protein identification. It is limited in its ability to consider evidence of a sample that may be known or determinable. As a pedagogical example, consider a hypothetical sample where many peptides have already been analysed. Assume that the majority of those peptides are strongly distinct for the gene family IGHV1-69. It is intuitive that further peptides for this sample should be more likely to be assigned to IGHV1-69. Strongly distinct peptides may receive only a smaller relative evidence boost, by nature of their assignment being near-certain initially, however those peptides that are potentially shared among a few peptides will benefit from the evidence contained in the entirety of the sample. Those peptides that are heavily shared may need a greater amount of evidence to escape the constraints of a relatively weak assignment. In particular, peptides that are shared in this way may never be confidently assigned as the assignment variance could be greater than the resolving power.

The standard method for adjustment of the peptide assignment Poisson model is with a secondary model that adjusts the model on the basis on prior or determined evidence. In general any model has functionality to transform a Poisson. However, some models are more suitable by their ability to simplify the combination. These models are known as *conjugate priors* or *conjugate models*. An appropriate conjugate model for the Poisson distribution is an exponential distribution.



Figure - An exponential distribution showing three representative scaling factors. The scaling factor describes how strongly prior or determined evidence is considered.

The Poisson-exponential conjugation is sufficient to represent peptide assignment with consideration of evidence determinable from the sample, but how is the evidence itself represented? The canonical model for describing elements that originate from one of many categories is the multinomial distribution. It is clear that any peptide that is observed can only physically result from a single protein. However, the ontological question is how likely such a peptide is a result from one of potentially many proteins. The multinomial distribution allows insight from several equivalent questions, some more intuitive than others -

- If any peptide **P** is sampled, without knowing the peptide sequence, what is the likelihood that it belongs to some protein **G**?

- If any peptide **P** is sampled, knowing the peptide sequence, what is the likelihood that it belongs to some protein **G**?

- Without knowing the origin protein **G**, what is the likelihood of sampling a peptide **P**?

- Knowing the origin protein **G**, what is the likelihood of sampling a peptide **P**?

Figure - A multinomial distribution for categorical variables. Here the categories are gene families with non-zero probability for the sample WM16\_B01+B01a\_HAGG\_isolated\_RF\_MJ1. The x-axis shows the gene family and the y-axis shows probability density. Note that the summation of probability densities is equal to 1.

The value of multinomial categories is typically the product of a scoring function that assigns a score for each peptide through an appropriate model. The scoring model implemented here is a *general linear model* (GLM). A GLM assigns a score as a linear combination of selected parameters. All combinations of parameters are possible, with some selection of parameters providing a better representation of the reality of the data. Their exists at least one optimal GLM for any observed data set. The GLM used here is the following -

Where is the score of the peptide for gene family ***i***, is the conjugated density, the value of the conjugated homology for gene family ***i*** in comparison to all other gene families, is...?

Function

**IgFamily v0.12.1h**

1. Initialise default settings and prompt user to confirm default settings or select custom settings.

2. If selected, perform msconvert file conversion:

2a. If (perform\_wiff\_fileconversion) perform wiff\_fileconversion().

3. If selected, perform Novor de novo peptide assignment:

3a. If (perform\_novor\_denovo) perform novor\_denovo().

4. Parse FASTA files into raw data structures:

4a. For (selected\_FASTA\_file) perform parse\_FASTA();

5. Parse data files into raw data structures:

5a. If (peptide\_assignment\_method == PEAKS\_database) perform parse\_PEAKS\_database\_peptides().

5b. If (peptide\_assignment\_method == PEAKS\_denovo) perform parse\_PEAKS\_denovo\_peptides().

5c. If (peptide\_assignment\_method == NOVOR\_denovo) perform parse\_NOVOR\_denovo\_peptides().

6. Assign raw data structures to designed data structures:

6a. Perform create\_v\_peptide\_data().

6b. Perform create\_v\_peptide\_analysis().

6c. Perform create\_v\_protein\_data().

7. From designed data structures, create blastp input file and blastp database:

7a. Perform create\_blastp\_input().

7b. Perform create\_blastp\_database().

8. Create a system process and direct blastp.exe to created input file and database:

8a. Perform systemcall\_blastp().

9. Parse blastp output file to raw data structures:

9a Perform parse\_homology\_data().

10. Transform homology data and associate homology data to peptide and protein data structures:

10a. Perform transform\_homology\_data().

10b. Perform associate\_homology\_data\_to\_peptide\_data().

10c. Perform associate\_homology\_data\_to\_protein\_data().

11. Through homology data association to peptide and protein data, determine homology\_density and score:

11a. Perform determine\_homology\_data\_parameters().

12. Create protein\_analysis data structures:

12a. Perform create\_v\_protein\_analysis().

13. Determine protein\_analysis parameters:

13a. Perform determine\_protein\_score\_density().

13b. Perform determine\_sequence\_coverage().

13c. Perform sort\_v\_protein\_analysis().

14. Determine most likely germline allele representation and create new blastp input:

14a. Perform select\_protein\_analysis\_by\_score().

14b. Perform create\_blastp\_database\_refined().

15. Create a system process and direct blastp.exe to created input file and database:

15a. Perform systemcall\_blastp().

16. Parse blastp output file to raw data structures:

16a Perform parse\_homology\_data().

17. Transform homology data and associate homology data to peptide and protein data structures:

17a. Perform transform\_homology\_data().

17b. Perform associate\_homology\_data\_to\_peptide\_data().

17c. Perform associate\_homology\_data\_to\_protein\_data().

18. Align query data to subject data:

18. create\_blastp\_query\_alignment().

19. Through homology data association to peptide and protein data, determine homology\_density and score:

19a. Perform determine\_homology\_data\_parameters().

20. Create protein\_analysis data structure:

20a. Perform create\_v\_protein\_analysis().

21. Conjugate homology\_data and protein\_analysis score through iterative process, until cluster condition is achieved:

21a. While (count\_ClusterProportion() > select\_nGeneFamilies) Perform conjugate\_homology ().

22. Determine protein\_analysis parameters:

22a. Perform determine\_protein\_score\_density().

22b. Perform determine\_sequence\_coverage().

22c. Perform sort\_v\_protein\_analysis().

23. Create consensus protein construct for \_ProteinAnalysis data:

23a. Perform create\_ProteinConstruct().

24. Create multinomial data frame:

24a. Perform create\_MultinomialData().

25. Create report and output data:

25a. Perform fout\_v\_PeptideData().

25b. Perform fout\_v\_ProteinData().

25c. Perform fout\_v\_PeptideAnalysis().

25d. Perform fout\_v\_ProteinAnalysis().

25e. Perform fout\_v\_HomologyData().

25f. Perform fout\_Multinomial().

25g. Perform fout\_MultinomialElement().

25h. Perform fout\_MultinomialElementNoMatch().

25i. Perform fout\_MultinomialContaminantsReport().

25j. Perform fout\_MultinomialContaminantsList().

25k. Perform fout\_MultinomialProteinScore().

25l. Perform fout\_MultinomialProteinDensity().

25m. Perform fout\_ProteinPseudoabundanceScore().

25n. Perform fout\_HTMLReport().

25o. Perform fout\_Filesystem().

Considerations

**- *A unique peptide not by any other name***

Unique peptides can provide strong support for a gene family and have been used in our lab and among the literature for a range of studies. However, there is a difficulty with unique peptides and immunoglobulins, such that the *uniqueness* or *distinctiveness* of a peptide is not necessarily determined by complete peptide sequence identity.

Consider the IGHV1-69 unique peptide shown in Table X from a truncated homology\_summary output.

Table X - The IGHV1-69 unique peptide FGTANYAQK.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| query | subject | subject\_accession | mismatch\_count | homology | homology\_density\_conjugated |
| FGTANYAQK | FGTANYAQK | IGHV1-69 | 0 | 47 | 22256.4 |
| FGTANYAQK | GNTNYAQK | IGHV1-45 | 3 | 19 | 2112.32 |
| FGTANYAQK | GNTNYAQK | IGHV1-18 | 3 | 19 | 2112.32 |
| FGTANYAQK | GNTNYAQK | IGHV1-58 | 3 | 18 | 1835.31 |
| FGTANYAQK | GGTNYAQK | IGHV1-2 | 3 | 18 | 1835.31 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| query | subject | subject\_accession | mismatch\_count | homology\_density |
| YYVDSVK | YYVDSVK | IGHV3-7 | 0 | 0.472811 |
| YYVDSVK | YYVDSVK | IGHV3-52 | 0 | 0.472811 |
| YYVDSVK | HYVDSVK | IGHV3-16 | 1 | 0.00565536 |
| YYVDSVK | YYADSVK | IGHV3-53 | 1 | 0.00451191 |
| YYVDSVK | YYADSVK | IGHV3-66 | 1 | 0.00451191 |
| YYVDSVK | YYADSVK | IGHV3-23 | 1 | 0.00451191 |
| YYVDSVK | YYADSVK | IGHV3-43 | 1 | 0.00451191 |
| YYVDSVK | YYADSVK | IGHV3-30-5 | 1 | 0.00451191 |
| YYVDSVK | YYADSVK | IGHV3-30 | 1 | 0.00451191 |
| YYVDSVK | YYADSVK | IGHV3-30-3 | 1 | 0.00451191 |
| YYVDSVK | YYADSVK | IGHV3-69-1 | 1 | 0.00451191 |
| YYVDSVK | YYADSVK | IGHV3-33 | 1 | 0.00451191 |
| YYVDSVK | YYADSVK | IGHV3-64 | 1 | 0.00400709 |
| YYVDSVK | YYADSVK | IGHV3-48 | 1 | 0.00400709 |
| YYVDSVK | HYADSVK | IGHV3-35 | 2 | 5.24E-05 |
| YYVDSVK | YADSVK | IGHV3-20 | 2 | 2.61E-05 |
| YYVDSVK | YYADSV | IGHV3-47 | 2 | 2.29E-05 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| query | subject | subject\_accession | mismatch\_count | homology\_density |
| YYADSVK | YYADSVK | IGHV3-23 | 0 | 0.0829693 |
| YYADSVK | YYADSVK | IGHV3-53 | 0 | 0.0829693 |
| YYADSVK | YYADSVK | IGHV3-64 | 0 | 0.0829693 |
| YYADSVK | YYADSVK | IGHV3-48 | 0 | 0.0829693 |
| YYADSVK | YYADSVK | IGHV3-66 | 0 | 0.0829693 |
| YYADSVK | YYADSVK | IGHV3-21 | 0 | 0.0829693 |
| YYADSVK | YYADSVK | IGHV3-43 | 0 | 0.0829693 |
| YYADSVK | YYADSVK | IGHV3-69-1 | 0 | 0.0829693 |
| YYADSVK | YYADSVK | IGHV3-30-5 | 0 | 0.0829693 |
| YYADSVK | YYADSVK | IGHV3-30 | 0 | 0.0829693 |
| YYADSVK | YYADSVK | IGHV3-30-3 | 0 | 0.0829693 |
| YYADSVK | YYADSVK | IGHV3-33 | 0 | 0.0829693 |
| YYADSVK | HYADSVK | IGHV3-35 | 1 | 0.000888038 |
| YYADSVK | YYVDSVK | IGHV3-52 | 1 | 0.000791754 |
| YYADSVK | YADSVK | IGHV3-20 | 1 | 0.000448287 |
| YYADSVK | YYVDSVK | IGHV3-7 | 1 | 0.000703167 |
| YYADSVK | YYADSV | IGHV3-47 | 1 | 0.00039813 |
| YYADSVK | YADSVK | IGHV3-74 | 1 | 0.00039813 |
| YYADSVK | YADSVK | IGHV3-9 | 1 | 0.000352115 |
| YYADSVK | YADSVK | IGHV3-11 | 1 | 0.000352115 |
| YYADSVK | HHADSVK | IGHV3-32 | 2 | 8.10E-06 |
| YYADSVK | HYVDSVK | IGHV3-16 | 2 | 7.10E-06 |
| YYADSVK | HHADSVK | IGHV3-29 | 2 | 7.10E-06 |
| YYADSVK | HHADSVK | IGHV3-30-42 | 2 | 7.10E-06 |
| YYADSVK | HHADSVK | IGHV3-30-22 | 2 | 7.10E-06 |

**- *Distance: It’s all relative***

Gah

Proposals

**- Analysis of data generation and reproduction**

**- Analysis of germline phylogeny**

**- Analysis of germline divergence**

**- Blastp internal integration**

There are notable limitations with using blastp through an external command line. Although parameters are able to be directed to blastp during IgFamily runtime, the modifiable parameters, while thorough, is restricted. In particular, blastp seems to set a hard-coded threshold for score output. This would not be an oversight for simple protein identification, but consider the following homology matches -

**- Blastp custom substitution matrix**

The BLOSUM62 substitution matrix is most often used for defining the rate of amino acid substitution. Specifically it describes the relative likelihood of any amino acid change occurring against an expected query amino acid.



**- Blastp custom conservation weighting**

The BLOSUM62 substitution matrix described above further assumes that the rate of substitution is spatially uniform - That is, any amino acid is equally likely to be substituted than any other. It is known that this is not the case for immunoglobulins, with a relatively greater likelihood of mutation in the hypervariable regions.

**- Advanced statistical modellng**

**- User interface and interaction**

**- Data filesystem**

**- Further automation**

**- De novo and database processes**

The algorithmic differences between de novo and database peptide assignment are suspected to confer some level of error to spectral assignment. It is not known what factors are responsible for incorrect assignment in either case.

Production

Version: v0.9.6

Release: 2016-09-17

Codebase: 4,857 source lines of code

Dependency: 18 files

Version history: 139 commits

Codebase additions: 21,040 source lines of code

Codebase deletions: 16,083 source lines of code

Development environment: Microsoft Visual Studio Community 2015

Version control: Git