

---

# ELASPIC Documentation

*Release 1.0*

**kimlab**

February 01, 2016



<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Database pipeline . . . . .	2
1.2	Local pipeline . . . . .	3
<b>2</b>	<b>Installation Guide</b>	<b>5</b>
2.1	Installing Python and ELASPIC . . . . .	5
2.2	Downloading external datasets . . . . .	6
2.3	Updating the configuration file . . . . .	6
2.4	Importing precalculated data . . . . .	8
<b>3</b>	<b>Command Line Interface</b>	<b>11</b>
3.1	elaspic run . . . . .	11
3.2	elaspic train . . . . .	12
3.3	elaspic database . . . . .	12
<b>4</b>	<b>Benchmarks</b>	<b>13</b>
<b>5</b>	<b>Statistics</b>	<b>15</b>
<b>6</b>	<b>Database</b>	<b>17</b>
6.1	Database schema . . . . .	17
6.2	Database tables . . . . .	17
<b>7</b>	<b>Modules</b>	<b>27</b>
7.1	elaspic package . . . . .	27
7.2	Indices and tables . . . . .	53
	<b>Python Module Index</b>	<b>55</b>



## Introduction

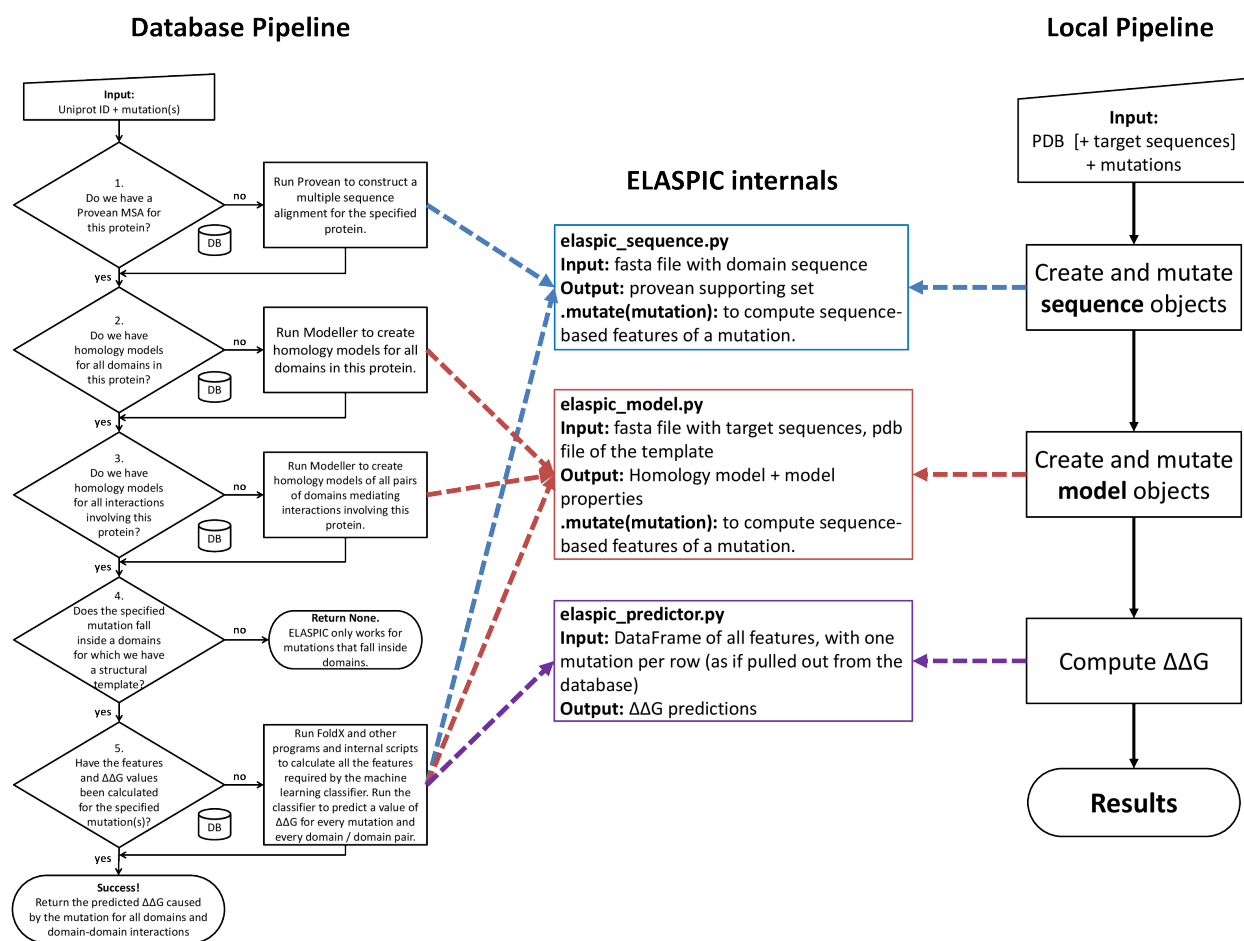
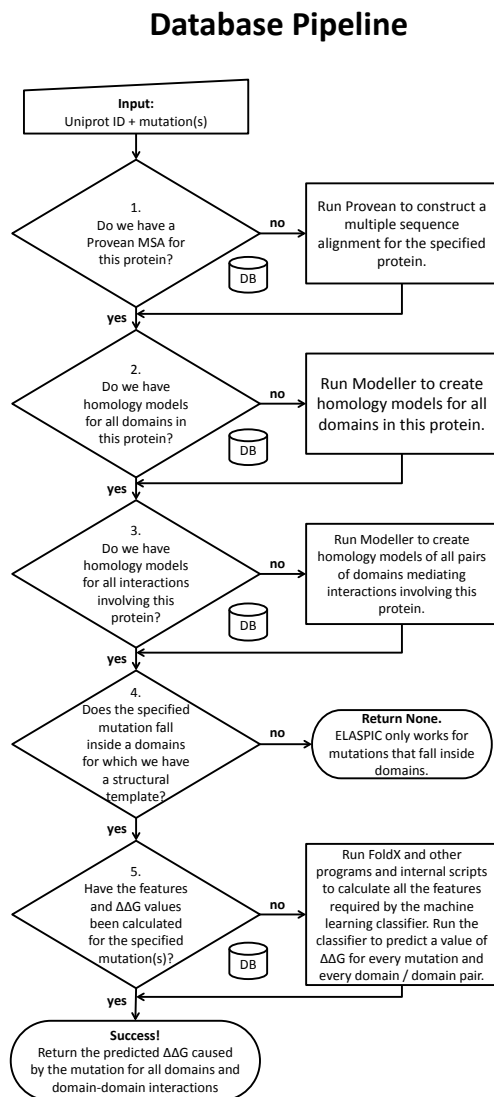


Fig. 1.1: Flowchart describing the ELASPIC pipeline.

ELASPIC can be run using two different pipelines: the *Local pipeline* and the *Database pipeline*.

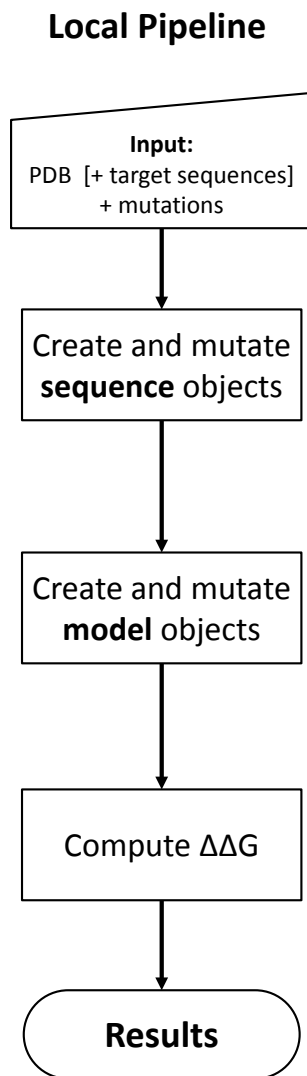
## 1.1 Database pipeline



The database pipeline allows mutations to be performed on a proteome-wide scale, without having to specify a structural template for each protein. This pipeline requires a local copy of *ELASPIC domain definitions and templates*, as well as a local copy of the *BLAST and PDB databases*.

The general overview of the database pipeline is presented in the figure to the right. A user runs the ELASPIC pipeline specifying the Uniprot ID of the protein being mutated, and one or more mutations affecting that protein. At each decision node, the pipeline queries the database to check whether or not the required information has been previously calculated. If the required data has not been calculated, the pipeline calculates it on the fly and stores the results in the database for later retrieval. The pipeline proceeds until homology models of all domains in the protein, and all domain-domain interactions involving the protein, have been calculated, and the  $\Delta\Delta G$  has been predicted for every specified mutation.

## 1.2 Local pipeline



The local pipeline works without downloading and installing a local copy of the ELASPIC and PDB databases, but requires a PDB structure or template to be provided for every protein. Pipeline output is saved as *JSON* files inside the working directory, rather than being uploaded to the database as in the case of the database pipeline. The general overview of the local pipeline is presented in the figure to the right.

The local pipeline still requires a local copy of the *Blast* nr database.





---

## Installation Guide

---

### In order to use the ELASPIC *Local pipeline* of your computer:

1. Install Python and ELASPIC (*Installing Python and ELASPIC*).
2. Download the BLAST database and preferably also the PDB database to a local folder (*Downloading external datasets*).

### In order to use the ELASPIC *Database pipeline*, in addition to the steps above:

1. Create a local database and modify the configuration file to match your system and database setting (*Updating the configuration file*).
2. Download Profs domain definitions for your organism of interest, and upload the data to a local database (*Importing precalculated data*).

## 2.1 Installing Python and ELASPIC

1. Download and install the [Anaconda Python Distribution](#) (Python 3) for Linux.
2. Add `bioconda`, `salilab`, and `ostrokach` channels to your `~/.condarc` file:

```
conda config --add channels ostrokach
conda config --add channels salilab
conda config --add channels bioconda
```

3. Obtain a [Modeller license](#), and export the license as `KEY_MODELLE`R in your `~/.bashrc` file:

```
# ~/.bashrc
export KEY_MODELLE=XXXXXXX
```

4. Install ELASPIC and all its dependencies into a new conda environment:

```
conda create -n elaspic elaspic
```

5. Activate the new environment and use `elaspic`:

```
source activate elaspic
elaspic --help
```

## 2.2 Downloading external datasets

### 2.2.1 Blast

Download and extract the *nr* and *pdbaa* databases from <ftp://ftp.ncbi.nlm.nih.gov/blast/db/>, and change the *blast\_db\_dir* variable in your *configuration file* to point to the directory containing the uncompressed files.

### 2.2.2 PDB

Download the contents of the <ftp://ftp.wwpdb.org/pub/pdb/data/structures/divided/pdb/> folder, and change the *pdb\_dir* variable in your *configuration file* to point to the directory containing the downloaded data.

## 2.3 Updating the configuration file

Edit the ELASPIC configuration file `./config/config_file.ini` to match your system:

1. Settings in the `[SEQUENCE]` section should be modified to match the location of your local BLAST and PDB databases.
  2. Settings in the `[DATABASE]` section should be modified to match the local MySQL, PostgreSQL, or SQLite database.
  3. Settings in the `[DEFAULT]` and `[MODEL]` may be left unchanged, since the default values are good enough in most cases.
- 

### 2.3.1 Configuration options

#### [DEFAULT]

**global\_temp\_dir** Location for storing temporary files. It will be used only if the `TMPDIR` environmental variable is not set. **Default = `'/tmp/'`.**

**temp\_dir string** A folder in the `global_temp_dir` that will contain all the files that are relevant to ELASPIC. Inside this folder, every job will create its own unique subfolder. **Default = `'elaspic/'`.**

**debug** Whether or not to show detailed debugging information. If `True`, the logging level will be set to `logging.DEBUG`. If `False`, the logging level will be set to `logging.INFO`. **Default = `True`.**

**look\_for\_interactions** Whether or not to compute models of protein-protein interactions. **Default = `True`.**

**remake\_provean\_supset** Whether or not to remake the Provean supporting set if one or more sequences cannot be found in the BLAST database. **Default = `False`.**

**n\_cores** Number of cores to use by programs that support multithreading. **Default = `1`.**

**web\_server** Whether or not the ELASPIC pipeline is being run as part of a webserver. **Default = `False`.**

**provean\_temp\_dir** Location to store provean temporary files if working on any note other than *beagle* or *banting*. For internal use only. **Default = `'`.**

**copy\_data** Whether or not to copy calculated data back to the archive. Set to `'False'` if you are planning to copy the data yourself (e.g. from inside a PBS or SGE script). **Default = `True`.**

**[SEQUENCE]**

**blast\_db\_dir** Location of the blast **nr** and **pdbaa** databases.

**blast\_db\_dir\_fallback** Place to look for blast **nr** and **pdbaa** databases if *blast\_db\_dir* does not exist.

**matrix\_type** Substitution matrix for calculating the mutation conservation score. **Default = 'blosum80'**.

**gap\_start** Penalty for starting a gap when calculating the mutation conservation score. **Default = -16**.

**gap\_extend** Penalty for extending a gap when calculating the mutation conservation score. **Default = -4**.

**[MODEL]**

**modeller\_runs** Number of models that MODELLER should make before choosing the best one. Not implemented!  
**Default = 1**.

**foldx\_water**

- **-CRYSTAL**: use water molecules in the crystal structure to bridge two protein atoms.
- **-PREDICT**: predict water molecules that make 2 or more hydrogen bonds to the protein.
- **-COMPARE**: compare predicted water bridges with bridges observed in the crystal structure.
- **-IGNORE**: don't predict water molecules. **Default**.

Source: <http://foldx.crg.es/manual3.jsp>.

**foldx\_num\_of\_runs** Number of times that FoldX should evaluate a given mutation. **Default = 1**.

**[DATABASE]**

**db\_type** The database that you are using. Supported databases are *MySQL*, *PostgreSQL*, and *SQLite*.

**sqlite\_db\_dir** Location of the SQLite database. Required only if *db\_type* is *SQLite*.

**db\_schema** The name of the schema that holds all elaspic data.

**db\_schema\_uniprot** The name of the database schema that holds uniprot sequences. Defaults to *db\_schema*.

**db\_database** The name of the database that contains *db\_schema* and *db\_schema\_uniprot*. Required only if *db\_type* is *PostgreSQL*. Defaults to *db\_schema*.

**db\_username** The username for the database. Required only if *db\_type* is *MySQL* or *PostgreSQL*.

**db\_password** The password for the database. Required only if *db\_type* is *MySQL* or *PostgreSQL*.

**db\_url** The IP address of the database. Required only if *db\_type* is *MySQL* or *PostgreSQL*.

**db\_port** The listening port of the database. Required only if *db\_type* is *MySQL* or *PostgreSQL*.

**db\_socket** Path to the socket file, if it is not in the default location. Used only if *db\_url* is *localhost*. For example:  
*/usr/local/mysql15/mysqld.sock* for *MySQL* and */var/lib/postgresql* for *PostgreSQL*.

**schema\_version** Database schema to use for storing and retrieving data. **Default = 'elaspic'**.

**archive\_type**

- **extracted**: all archive files are contained in an extracted directory tree.
- **7zip**: archive is made of three compressed 7zip files (*provean/provean.7z*, *uniprot\_domain/uniprot\_domain.7z*, *uniprot\_domain\_pair/uniprot\_domain\_pair.7z*), provided on the [elaspic downloads page](#).

**archive\_dir** Location for storing and retrieving precalculated data.

**pdb\_dir** Location of all pdb structures, equivalent to the “data/data/structures/divided/pdb/” folder in the PDB ftp site. Optional.

## 2.3.2 Environmental variables

### PATH

A colon-separated list of paths where ELASPIC should look for required programs, such as BLAST, T-coffee, Modeller, and cd-hit.

### TMPDIR

Location to store all temporary files and folders.

## 2.4 Importing precalculated data

### 2.4.1 ELASPIC downloads page

The [ELASPIC downloads page](#) contains all precalculated data that is required to run the ELASPIC pipeline on a local machine.

The `*.tsv.gz` files correspond to different tables of the *ELASPIC database*:

- The `domain.tar.gz` file in the root folder contains Profs domain definitions for files in the PDB, and corresponds to the *domain* table.
- The `domain_contact.tar.gz` file in the root folder contains a list of interactions between those domains, and corresponds to the *domain\_contact* table.
- All other tables are split into separate folders according to the organism of origin. The files are named using the `{table_name}.tsv.gz` convention, where `table_name` is the name of the table in the database.

The `*.7z` files contain precalculated data:

- The *provean*, *uniprot\_domain*, and *uniprot\_domain\_pair* subfolders contain precalculated provean supporting sets, and homology models of protein domains and domain-domain interactions, respectively.

Precalculated mutations:

- The *Homo\_sapiens* folder contains an additional subfolder `precalculated_mutations`, which contains  $\Delta\Delta G$  scores for mutations in various datasets.

---

**Note:** The `configure_test.sh` and `run_test.sh` scripts in the `./scripts` folder contain examples of how to download and set up a local copy of the database.

---

### 2.4.2 Downloading data

In order to run up ELASPIC on a local computer, you need to download precalculated data for your organism of interest. If your goal is to only test the pipeline, you can download a test dataset from the folder `current_release/Homo_sapiens_test`.

To download all precalculated data for a given organism, use the `wget` command:

```
# Download external files
wget -P "${TEST_DIR}/elaspic.kimlab.org" \
    http://elaspic.kimlab.org/static/download/current_release/domain.tsv.gz
wget -P "${TEST_DIR}/elaspic.kimlab.org" \
    http://elaspic.kimlab.org/static/download/current_release/domain_contact.tsv.gz
wget -P "${TEST_DIR}" \
    -r --no-parent --reject "index.html*" --cut-dirs=4 \
    http://elaspic.kimlab.org/static/download/current_release/Homo_sapiens_test/
```

You need to extract the provean supporting sets and domain homology models into a folder specified by the *archive\_dir* variable in your *configuration\_file*:

```
mkdir archive # Set 'archive_dir' variable in the config file to this folder

7z x "${TEST_DIR}/elaspic.kimlab.org/provean/provean.7z" -o"archive"
7z x "${TEST_DIR}/elaspic.kimlab.org/uniprot_domain/uniprot_domain.7z" -o"archive"
7z x "${TEST_DIR}/elaspic.kimlab.org/uniprot_domain_pair/uniprot_domain_pair.7z" -o"archive"
```

### 2.4.3 Importing data into a database

You also need to create a local SQL database and fill it with precalculated data.

Modify the database variables in the ELASPIC *configuration\_file* to match your local *MySQL*, *PostgreSQL*, or *SQLite* database, and use the *elaspic database* CLI to create a new database and fill it with precalculated data.

First, you need to create an empty database:

```
elaspic database -c {your_configuration_file}.ini create
```

Next, you need to load all precalculated data for the organism in question to your database:

```
elaspic database -c {your_configuration_file}.ini load_data
```

To delete the database that you just created, run:

```
elaspic database -c {your_configuration_file}.ini delete
```



---

## Command Line Interface

---

After following instructions in the *Installation Guide*, you should be able to run ELASPIC from the command line using the `elaspic` command:

```
$ elaspic --help
usage: elaspic [-h] {run,database,train} ...

optional arguments:
  -h, --help            show this help message and exit

command:
  {run,database,train}
  run                   Run ELASPIC.
  database              Perform maintenance tasks on the ELASPIC database.
  train                Train the ELASPIC classifiers.
```

Type `--help` to see the options available for each subcommand:

- `elaspic run --help`
- `elaspic database --help`
- `elaspic database load_data --help`
- etc...

### 3.1 elaspic run

Run the ELASPIC pipeline.

If you wish to mutate an existing PDB, you should specify the name of the PDB file to be mutated, and the mutation(s):

```
elaspic run \
  --structure_file {structure_file} \
  --mutations {mutations}
```

If you wish to first create a homology model of a protein, you should provide a fasta file containing the sequence of the protein to be modelled, a PDB file containing the structural template, and the mutation(s):

```
elaspic run \
  --sequence_file {sequence_file} \
  --structure_file {structure_file} \
  --mutations {mutations}
```

If you wish to perform mutagenesis on a proteome-wide scale, you need to download protein domain definitions from the [elaspic downloads page](#), and optionally a local copy of the PDB database. After saving your database information to a configuration file, you can run specify the uniprot id and mutation(s):

```
elaspic run \  
  --config_file {config_file} \  
  --uniprot_id {uniprot_id} \  
  --mutations {mutations}
```

## 3.2 elaspic train

Train the machine learning predictor for the ELASPIC pipeline.

This is automatically done at install time, and you *do not* need to do this again unless you update your `scikit-learn` version.

## 3.3 elaspic database

Perform maintenance tasks on the ELASPIC database.

You must provide a configuration file containing the details of your database installation for any of these commands to work. For more information about configuration files, see *Updating the configuration file*.

### 3.3.1 elaspic database create

Create a new database schema.

### 3.3.2 elaspic database load\_data

Load data to the database.

### 3.3.3 elaspic database delete

Delete the database schema.



---

**Benchmarks**

---

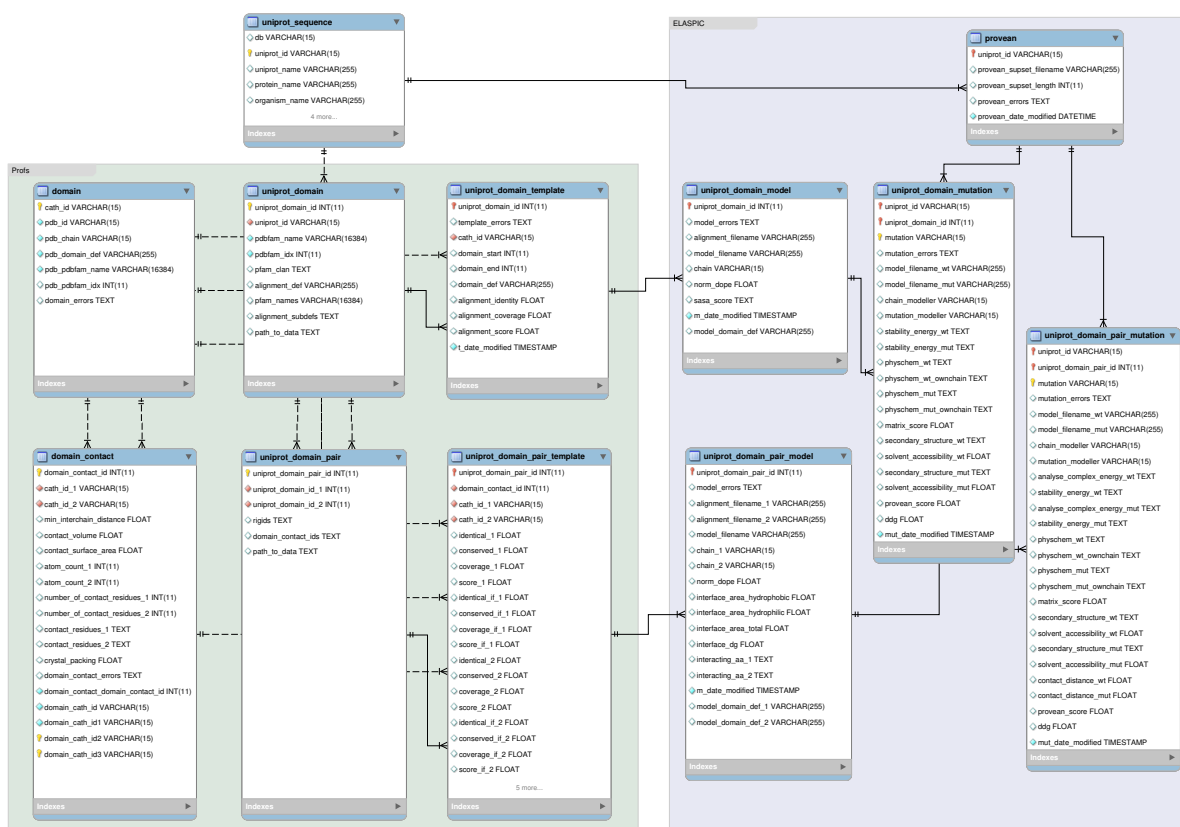
Work in progres...



Work in progres...



## 6.1 Database schema



## 6.2 Database tables

### 6.2.1 domain

Profs domain definitions for all proteins in the PDB.

**Columns:**

**cath\_id** Unique id identifying each domain in the PDB. Constructed by concatenating the `pdb_id`, `pdb_chain`, and an index specifying the order of the domain in the chain.

**pdb\_id** The PDB id in which the domain is found.

**pdb\_chain** The PDB chain in which the domain is found.

**pdb\_domain\_def** Domain definitions of the domain, in PDB RESNUM coordinates.

**pdb\_pdbfam\_name** The Profs name of the domain.

**pdb\_pdbfam\_idx** An integer specifying the number of times a domain with domain name `pdb_pdbfam_name` has occurred in this chain up to this point. It is used to make every (`pdb_id`, `pdb_chain`, `pdb_pdbfam_name`, `pdb_pdbfam_idx`) tuple unique.

**domain\_errors** List of errors that occurred when annotating this domain, or when using this domain to make structural homology models.

## 6.2.2 domain\_contact

Interactions between Profs domains in the PDB. Only interactions that were predicted to be biologically relevant by [NOXclass](#) are included in this table.

### Columns:

**domain\_contact\_id** A unique integer identifying each domain pair.

**cath\_id\_1** Unique id identifying the first interacting domain in the *domain* table.

**cath\_id\_2** Unique id identifying the second interacting domain in the *domain* table.

**min\_interchain\_distance** The closest that any residue in domain one comes to any residue in domain two.

**contact\_volume** The volume covered by contacting residues.

**contact\_surface\_area** The surface area of the contacting regions of the first and second domains.

**atom\_count\_1** The number of atoms in the first domain.

**atom\_count\_2** The number of atoms in the second domain.

**number\_of\_contact\_residues\_1** The number of residues in the first domain that come within 5 of the second domain.

**number\_of\_contact\_residues\_2** The number of residues in the second domain that come within 5 of the first domain.

**contact\_residues\_1** A list of all residues in the first domain that come within 5 of the second domain. The residue number corresponds to the position of the residue in the domain.

**contact\_residues\_2** A list of all residues in the second domain that come within 5 of the first domain. The residue number corresponds to the position of the residue in the domain.

**crystal\_packing** The probability that the interaction is a crystallization artifacts, as defined by [NOXclass](#).

**domain\_contact\_errors** List of errors that occurred when annotating this domain pair, or when using this domain as a template for making structural homology models.

## 6.2.3 uniprot\_sequence

Protein sequences from the Uniprot KB, obtained by parsing *uniprot\_sprot\_fasta.gz*, *uniprot\_trembl\_fasta.gz*, and *homo\_sapiens\_variation.txt* files from the [Uniprot ftp site](#).

**Columns:**

**db** The database to which the protein sequence belongs. Possible values are *sp* for SwissProt and *tr* for TrEMBL.

**uniprot\_id** The uniprot id of the protein.

**uniprot\_name** The uniprot name of the protein.

**protein\_name** The protein name.

**organism\_name** Name of the organism in which this protein is found.

**gene\_name** Name of the gene that codes for this protein sequence.

**protein\_existence** Evidence for the existence of the protein:

1. Experimental evidence at protein level
2. Experimental evidence at transcript level
3. Protein inferred from homology
4. Protein predicted
5. Protein uncertain

**sequence\_version** Version of the protein amino acid sequence.

**uniprot\_sequence** Amino acid sequence of the protein.

## 6.2.4 provean

Description of the [Provean](#) supporting set calculated for a protein sequence. The construction of a supporting set is the most lengthy step in running Provean. Therefore, the supporting set is precalculated and stored for every protein sequence.

**Columns:**

**uniprot\_id** The uniprot id of the protein.

**provean\_supset\_filename** The filename of the Provean supporting set. The supporting set contains the ids and sequences of all proteins in the NCBI nr database that are used by Provean to construct a multiple sequence alignment for the given protein.

**provean\_supset\_length** The number of sequences in Provean supporting set.

**provean\_errors** List of errors that occurred while the Provean supporting set was being calculated.

**provean\_date\_modified** Date and time that this row was last modified.

## 6.2.5 uniprot\_domain

Pfam domain definitions for proteins in the *uniprot\_sequence* table. This table was obtained by downloading Pfam domain definitions for all known proteins from the [SIMAP](#) website, and mapping the protein sequence to uniprot using the MD5 hash of each sequence.

**Columns:**

**uniprot\_domain\_id** Unique id identifying each domain.

**uniprot\_id** The uniprot id of the protein containing the domain.

**pdbfam\_name** The Profs name of the domain. In most cases this will be equivalent to the Pfam name of the domain.

**pdbfam\_idx** The index of the Profs domain. `pdbfam_idx` ranges from 1 to the number of domains with the name `pdbfam_name` in the given protein. The `(pdbfam_name, pdbfam_idx)` tuple uniquely identifies each domain.

**pfam\_clan** The Pfam clan to which this Profs domain belongs.

**alignment\_def** Alignment domain definitions of the Profs domain. This field is obtained by removing gaps in the `alignment_subdefs` column.

**pfam\_names** Pfam names of all Pfam domains that were combined to create the given Profs domain.

**alignment\_subdefs** Comma-separated list of domain definitions for all Pfam domains that were merged to create the given Profs domain.

**path\_to\_data** Location for storing homology models, mutation results, and all other data that are relevant to this domain. This path is prefixed by *archive\_dir*.

## 6.2.6 uniprot\_domain\_template

Structural templates for domains in the *uniprot\_domain* table. Lists PDB crystal structures that will be used for making homology models.

### Columns:

**uniprot\_domain\_id** An integer which uniquely identifies each uniprot domain in the *uniprot\_domain* table.

**template\_errors** List of errors that occurred during the process for finding the template.

**cath\_id** The unique id identifying the structural template of the domain.

**domain\_start** The Uniprot position of the first amino acid of the Profs domain.

**domain\_end** The Uniprot position of the last amino acid of the Profs domain.

**domain\_def** Profs domain definitions for domains with structural templates. Domain definitions in this column are different from domain definitions in the `alignment_def` column of the *uniprot\_domain* table in that they have been expanded to match domain boundaries of the Profs structural template, identified by the `cath_id`.

**alignment\_identity** Percent identity of the domain to its structural template.

**alignment\_coverage** Percent coverage of the domain to its structural template.

**alignment\_score** A score obtained by combining `alignment_identity` (*SeqId*) and `alignment_coverage` (*Cov*) using the following equation, as described by Mosca et al.:

$$Score = 0.95 \cdot \frac{SeqId}{100} \cdot \frac{Cov}{100} + 0.05 \cdot \frac{Cov}{100} \quad (6.1)$$

**t\_date\_modified** The date and time when this row was last modified.

## 6.2.7 uniprot\_domain\_model

Homology models for templates in the *uniprot\_domain\_template* table.

### Columns:

**uniprot\_domain\_id** An integer which uniquely identifies each uniprot domain in the *uniprot\_domain* table.



**model\_errors** List of errors that occurred when making the homology model.

**alignment\_filename** The name of the alignment that was given to Modeller when making the homology model.

**model\_filename** The name of the homology model that was produced by Modeller.

**chain** The chain that contains the domain in question in the homology (this is now set to 'A' in all models).

**norm\_dope** Normalized DOPE score of the model (lower is better).

**sasa\_score** Comma-separated list of the percent solvent-accessible surface area for each residue.

**m\_date\_modified** The date and time when this row was last modified.

**model\_domain\_def** Domain definitions for the region of the domain that is covered by the structural template.

In most cases, this field is identical to the `domain_def` field in the *uniprot\_domain\_template* table. However, it sometimes happens that the best Profs structural template only covers a fraction of the Pfam domain. In that case, the `alignment_def` column in the *uniprot\_domain* table, and the `domain_def` column in the *uniprot\_domain\_template* table, will contain the original Pfam domain definitions, and the `model_domain_def` column will contain domain definitions for only the region that is covered by the structural template.

## 6.2.8 uniprot\_domain\_mutation

Characterization of mutations introduced into structures in the *uniprot\_domain\_model* table.

### Columns:

**uniprot\_id** Uniprot ID of the protein that was mutated.

**uniprot\_domain\_id** Unique id which identifies the Profs domain that was mutated in the *uniprot\_domain* table.

**mutation** Mutation that was introduced into the protein, in Uniprot coordinates.

**mutation\_errors** List of errors that occurred while evaluating the mutation.

**model\_filename\_wt** The name of the file which contains the homology model of the domain after the model was relaxed with FoldX but before the mutation was introduced.

**model\_filename\_mut** The name of the file which contains the homology model of the domain after the model was relaxed with FoldX and after the mutation was introduced.

**chain\_modeller** The chain which contains the domain that was mutated in the `model_filename_wt` and the `model_filename_mut` structures.

**mutation\_modeller** The mutation that was introduced into the protein, in PDB RESNUM coordinates. This identifies the mutated residue in the `model_filename_wt` and the `model_filename_mut` structures.

**stability\_energy\_wt** Comma-separated list of scores returned by FoldX for the wildtype protein. The comma-separated list can be converted into a DataFrame with each column clearly labelled using the `elaspic.predictor.format_mutation_features()`. The FoldX energy terms are:

- dg
- backbone\_hbond
- sidechain\_hbond
- van\_der\_waals
- electrostatics
- solvation\_polar

- solvation\_hydrophobic
- van\_der\_waals\_clashes
- entropy\_sidechain
- entropy\_mainchain
- sloop\_entropy
- mloop\_entropy
- cis\_bond
- torsional\_clash
- backbone\_clash
- helix\_dipole
- water\_bridge
- disulfide
- electrostatic\_kon
- partial\_covalent\_bonds
- energy\_ionisation
- entropy\_complex
- number\_of\_residues

**stability\_energy\_mut** Comma-separated list of scores returned by FoldX for the mutant protein. FoldX energy terms are the same as in *stability\_energy\_wt*, but for the mutated amino acid rather than the wildtype.

**physchem\_wt** Physicochemical properties describing the interaction of the wildtype residue with residues on the opposite chain. The terms are:

- number of atoms in interacting residues that have the same charge.
- number of atoms in interacting residues that have an opposite charge.
- number of hydrogen bonds (very rough calculation).
- number of carbons in interacting residues within 4 Å of the mutated residue (rough measure of the van der Waals force).

**physchem\_wt\_ownchain** Physicochemical properties describing the interaction of the wildtype residue with residues on the same chain. The terms are the same as in *physchem\_wt*.

**physchem\_mut** Physicochemical properties describing the interaction of the mutant residue with residues on the opposite chain. The terms are the same as in *physchem\_wt*.

**physchem\_mut\_ownchain** Physicochemical properties describing the interaction of the mutant residue with residues on the same chain. The terms are the same as in *physchem\_wt*.

**matrix\_score** Score assigned to the wt -> mut transition by the BLOSUM substitution matrix.

**secondary\_structure\_wt** Secondary structure of the wildtype residue predicted by [stride](#).

**solvent\_accessibility\_wt** Percent solvent accessible surface area of the wildtype residue, predicted by [msms](#).

**secondary\_structure\_mut** Secondary structure of the mutated residue predicted by [stride](#).

**solvent\_accessibility\_mut** Percent solvent accessible surface area of the mutated residue, predicted by [msms](#).

**provean\_score** Score produced by [Provean](#) for this mutation.

**ddg** Change in the Gibbs free energy of folding that our classifier predicts for this mutation.

**mut\_date\_modified** Date and time that this row was last modified.

## 6.2.9 uniprot\_domain\_pair

Potentially-interacting pairs of domains for proteins that are known to interact, according to Hippie, IRefIndex, and Rolland et al. 2014.

### Columns:

**uniprot\_domain\_pair\_id** Unique id identifying each domain-domain interaction.

**uniprot\_domain\_id\_1** Unique id of the first domain.

**uniprot\_domain\_id\_2** Unique id of the second domain.

**rigids** Phased out.

**domain\_contact\_ids** List of unique ids identifying all domain-domain pairs in the PDB, where one domain belongs to the protein containing `uniprot_domain_id_1` and the other domain belongs to the protein containing `uniprot_domain_id_2`. This was used as crystallographic evidence that the two proteins interact.

**path\_to\_data** Location for storing homology models, mutation results, and all other data that is relevant to this domain pair. This path is prefixed by `archive_dir`.

## 6.2.10 uniprot\_domain\_pair\_template

Structural templates for pairs of domains in the `uniprot_domain_pair` table.

### Columns:

**uniprot\_domain\_pair\_id** Unique id identifying each domain-domain interaction.

**domain\_contact\_id** Unique id of the domain pair in the `domain_contact` table that was used as a template for the modelled domain pair.

**cath\_id\_1** Unique id of the structural template for the first domain.

**cath\_id\_2** Unique id of the structural template for the second domain.

**identical\_1** Fraction of residues in the Blast alignment of the first domain to its template that are *identical*.

**conserved\_1** Fraction of residues in the Blast alignment of the first domain to its template that are *conserved*.

**coverage\_1** Fraction of the first domain that is covered by the blast alignment.

**score\_1** Score obtained by multiplying `identical_1` by `coverage_1`.

**identical\_if\_1** Fraction of interface residues<sup>1</sup> that are *identical* in the Blast alignment of the first domain.

**conserved\_if\_1** Fraction of interface residues<sup>1</sup> that are *conserved* in the Blast alignment of the first domain.

**coverage\_if\_1** Fraction of interface residues<sup>1</sup> that are *covered* by the Blast alignment of the first domain.

**score\_if\_1** Score obtained by combining `identical_if_1` and `coverage_if_1` using (6.1).

**identical\_2** Fraction of residues in the Blast alignment of the second domain to its template that are *identical*.

**conserved\_2** Fraction of residues in the Blast alignment of the second domain to its template that are *conserved*.

**coverage\_2** Fraction of the second domain that is covered by the blast alignment.

<sup>1</sup> Interface residues are defined as residues that are within 5 of the partner domain.

**score\_2** Score obtained by multiplying `identical_2` by `coverage_2`.

**identical\_if\_2** Fraction of interface residues <sup>1</sup> that are *identical* in the Blast alignment of the second domain.

**conserved\_if\_2** Fraction of interface residues <sup>1</sup> that are *conserved* in the Blast alignment of the second domain.

**coverage\_if\_2** Fraction of interface residues <sup>1</sup> that are *covered* by the Blast alignment of the second domain.

**score\_if\_2** Score obtained by combining `identical_if_2` and `coverage_if_2` using (6.1).

**score\_total** The product of `score_1` and `score_2`.

**score\_if\_total** The product of `score_if_1` and `score_if_2`.

**score\_overall** The product of `score_total` and `score_if_total`. This is the score that was used to select the best Profs domain pair to be used as a template.

**t\_date\_modified** The date and time when this row was last updated.

**template\_errors** List of errors that occurred while looking for the structural template.

### 6.2.11 uniprot\_domain\_pair\_model

Structural models of interactions between pairs of domains in the *uniprot\_domain\_pair* table.

#### Columns:

**uniprot\_domain\_pair\_id** Unique id identifying each domain-domain interaction.

**model\_errors** List of errors that occurred while making the homology model.

**alignment\_filename\_1** Name of the file containing the alignment of the first domain with its structural template.

**alignment\_filename\_2** Name of the file containing the alignment of the second domain with its structural template.

**model\_filename** Name of the file containing the homology model of the domain-domain interaction created by Modeller.

**chain\_1** Chain containing the first domain in the model specified by `model_filename`.

**chain\_2** Chain containing the second domain in the model specified by `model_filename`.

**norm\_dope** The normalized DOPE score of the model.

**interface\_area\_hydrophobic** Hydrophobic surface area of the interface, calculated using POPS.

**interface\_area\_hydrophilic** Hydrophilic surface area of the interface, calculated using POPS.

**interface\_area\_total** Total surface area of the interface, calculated using POPS.

**interface\_dg** Gibbs free energy of binding for this domain-domain interaction, predicted using FoldX. Not implemented yet!

**interacting\_aa\_1** List of amino acid positions in the first domain that are within 5 of the second domain. Positions are specified using uniprot coordinates.

**interacting\_aa\_2** List of amino acids in the second domain that are within 5 of the first domain. Position are specified using uniprot coordinates.

**m\_date\_modified** Date and time that this row was last modified.

**model\_domain\_def\_1** Domain boundaries of the first domain that are covered by the Profs structural template.

**model\_domain\_def\_2** Domain boundaries of the second domain that are covered by the Profs structural template.

## 6.2.12 uniprot\_domain\_pair\_mutation

Characterization of interface mutations introduced into structures in the *uniprot\_domain\_pair\_model* table.

### Columns:

- uniprot\_id** Uniprot ID of the protein that is being mutated.
- uniprot\_domain\_pair\_id** Unique id identifying each domain-domain interaction.
- mutation** Mutation for which the  $\Delta\Delta G$  score is being predicted, specified in Uniprot coordinates.
- mutation\_errors** List of errors obtained when evaluating the impact of the mutation.
- model\_filename\_wt** Filename of the homology model relaxed by FoldX but containing the wildtype residue.
- model\_filename\_mut** Filename of the homology model relaxed by FoldX and containing the mutated residue.
- chain\_modeller** Chain containing the domain that was mutated, in homology models specified by `model_filename_wt` and `model_filename_mut`.
- mutation\_modeller** Mutation for which the  $\Delta\Delta G$  score is being predicted, specified in PDB RESNUM coordinates.
- analyse\_complex\_energy\_wt** Comma-separated list of FoldX scores describing the effect of the wildtype residue on the stability of the protein domain.
- stability\_energy\_wt** Comma-separated list of FoldX scores describing the effect of the wildtype residue on protein-protein interaction interface.
- analyse\_complex\_energy\_mut** Comma-separated list of FoldX scores describing the effect of the mutated residue on the stability of the protein domain.
- stability\_energy\_mut** Comma-separated list of FoldX scores describing the effect of the mutated residue on protein-protein interaction interface.
- physchem\_wt** Comma-separated list of physicochemical properties describing the interaction between the wildtype residue and other residues on the opposite chain.
- physchem\_wt\_ownchain** Comma-separated list of physicochemical properties describing the interaction between the wildtype residue and other residues on the same chain.
- physchem\_mut** Comma-separated list of physicochemical properties describing the interaction between the mutated residue and other residues on the opposite chain.
- physchem\_mut\_ownchain** Comma-separated list of physicochemical properties describing the interaction between the mutated residue and other residues on the same chain.
- matrix\_score** Score assigned to the wt -> mut transition by the BLOSUM substitution matrix.
- secondary\_structure\_wt** Secondary structure of the wildtype residue, predicted by `stride`.
- solvent\_accessibility\_wt** Percent solvent accessible surface area of the wildtype residue, predicted by `msms`.
- secondary\_structure\_mut** Secondary structure of the mutated residue, predicted by `stride`.
- solvent\_accessibility\_mut** Percent solvent accessible surface area of the mutated residue, predicted by `msms`.
- contact\_distance\_wt** Shortest distance between the wildtype residue and a residue on the opposite chain.
- contact\_distance\_mut** Shortest distance between the mutated residue and a residue on the opposite chain.
- provean\_score** `Provean` score for this mutation.
- ddg** Predicted change in Gibbs free energy of binding caused by this mutation.
- mut\_date\_modified** Date and time when this row was last modified.



## 7.1 elaspic package

### 7.1.1 Submodules

### 7.1.2 elaspic.call\_foldx module

**class** `elaspic.call_foldx.FoldX` (*pdb\_file, chain\_id, foldx\_dir=None*)

Bases: `object`

`__call__` (*whatToRun, mutCodes=[]*)

Select which action should be performed by FoldX by setting *whatToRun*.

Possible values are:

- AnalyseComplex
- Stability
- RepairPDB
- BuildModel

See the [FoldX manual](#) for an explanation on what they do.

### 7.1.3 elaspic.call\_modeller module

### 7.1.4 elaspic.call\_tcoffee module

**class** `elaspic.call_tcoffee.TCoffee` (*alignment\_fasta\_file, mode, pdb\_file=None*)

Bases: `object`

Aligns sequences using `t_coffee` in espresso mode.

`align` (*GAOPEN=-0.0, GAPEXTEND=-0.0*)

Calls `t_coffee` (make sure BLAST is installed locally!).

#### Parameters

- **alignment\_fasta\_file** (*string*) – A file containing the fasta sequences to be aligned
- **alignment\_template\_file** (*string*) – A file containing the structural templates for the fasta sequences described above

- **GAPOPEN** (*int or str*) – See `t_coffee` manual
- **GAPEXTEND** (*int or str*) – See `t_coffee` manual
- **Returns** –
- -----
- **alignment\_output\_file** (*str*) – Name of file which contains the alignment in fasta format.

### 7.1.5 elaspic.conf module

**class** `elpasic.conf.Configs`

Bases: `object`

A singleton class that keeps track of ELASPIC configuration settings.

**clear** ()

**copy** ()

**get** (*key, fallback=None*)

**items** ()

**keys** ()

**update** (\*\**kwargs*)

**values** ()

**class** `elpasic.conf.Singleton`

Bases: `type`

**instance** = `None`

`elpasic.conf.get_temp_dir` (*global\_temp\_dir='/tmp', elaspic\_foldername=''*)

If a `TMPDIR` is given as an environment variable, the `tmp` directory is created relative to that. This is useful when running on banting (the cluster in the `ccbr`) and also on `Scinet`. Make sure that it points to `/dev/shm/` on `Scinet`.

`elpasic.conf.read_configuration_file` (*config\_file, unique\_temp\_dir=None*)

`elpasic.conf.read_database_configs` (*configParser*)

[DATABASE]

`elpasic.conf.read_model_configs` (*configParser*)

[MODEL]

`elpasic.conf.read_sequence_configs` (*configParser*)

[SEQUENCE]

### 7.1.6 elaspic.database\_pipeline module

### 7.1.7 elaspic.elpasic\_database module

**class** `elpasic.elpasic_database.MyDatabase` (*echo=False*)

Bases: `object`

**add\_domain** (*d*)



**add\_domain\_errors** (*t*, *error\_string*)

**add\_uniprot\_sequence** (*uniprot\_sequence*)

Add new sequences to the database. :param uniprot\_sequence: UniprotSequence object :rtype: None

**configure\_session** ()

Configure the Session class to use the current engine.

*autocommit* and *autoflush* are enabled for the *sqlite* database in order to improve performance.

**copy\_table\_to\_db** (*table\_name*, *table\_folder*)

Copy data from a *.tsv* file to a table in the database.

**create\_database\_tables** (*clear\_schema=False*, *keep\_uniprot\_sequence=True*)

Create a new database in the schema specified by the *schema\_version* global variable. If *clear\_schema == True*, remove all the tables in the schema first.

**Warning:** Using this function with an existing database can lead to loss of data. Make sure that you know what you are doing!

#### Parameters

- **clear\_schema** (*bool*) – Whether or not to delete all tables in the database schema before creating new tables.
- **keep\_uniprot\_sequence** (*bool*) – Whether or not to keep the *uniprot\_sequence* table. Only relevant if *clear\_schema* is *True*.

**delete\_database\_tables** (*drop\_schema=False*, *keep\_uniprot\_sequence=True*)

#### Parameters

- **drop\_schema** (*bool*) – Whether or not to drop the schema after dropping the tables.
- **keep\_uniprot\_sequence** (*bool*) – Whether or not to keep the table (and schema) containing uniprot sequences.

**get\_alignment** (*model*, *path\_to\_data*)

**get\_domain** (*pfam\_names*, *subdomains=False*)

Returns pdbfam-based definitions of all pfam domains in the pdb.

**get\_domain\_contact** (*pfam\_names\_1*, *pfam\_names\_2*, *subdomains=False*)

Returns domain-domain interaction information from pdbfam. Note that the produced dataframe may not have the same order as the keys.

**get\_engine** (*echo=False*)

Get an SQLAlchemy engine that can be used to connect to the database.

**get\_rows\_by\_ids** (*row\_object*, *row\_object\_identifiers*, *row\_object\_identifier\_values*)

Get the rows from the table *row\_object* identified by keys *row\_object\_identifiers* with values *row\_object\_identifier\_values*

**get\_uniprot\_domain** (*uniprot\_id*, *copy\_data=False*)

**get\_uniprot\_domain\_pair** (*uniprot\_id*, *copy\_data=False*, *uniprot\_domain\_pair\_ids=[]*)

**get\_uniprot\_mutation** (*d*, *mutation*, *uniprot\_id=None*, *copy\_data=False*)

**get\_uniprot\_sequence** (*uniprot\_id*, *check\_external=True*)

#### Parameters

- **uniprot\_id** (*str*) – Uniprot ID of the protein

- **check\_external** (*bool*) – Whether or not to look online if the protein sequence is not found in the local database.

**Returns** Contains the sequence of the specified uniprot

**Return type** SeqRecord

**load\_db\_from\_archive** ()

TODO: In the future I should move back to using json...

**merge\_model** (*d, files\_dict={}*)

Adds MODELLER models to the database.

**merge\_mutation** (*mut, path\_to\_data=False*)

**merge\_provean** (*provean, provean\_supset\_file, path\_to\_data*)

Adds provean score to the database.

**merge\_row** (*row\_instance*)

Adds a list of rows (*row\_instances*) to the database.

**mysql\_command\_template** = "load data local infile '{table\_folder}/{table\_name}.tsv' into table {table\_db\_schema}." + {table\_name} + ".tbl"

**mysql\_load\_table\_template** = 'mysql -local-infile -host={db\_url} -user={db\_username} -password={db\_password} -u {db\_username} -D {table\_db\_schema} -e "{table\_name} < {table\_folder}/{table\_name}.tsv"'

**psql\_command\_template** = "\\copy {table\_db\_schema}.{table\_name} from '{table\_folder}/{table\_name}.tsv' with csv"

**psql\_load\_table\_template** = 'PGPASSWORD={db\_password} psql -h {db\_url} -p {db\_port} -U {db\_username} -d {table\_db\_schema} -e "{table\_name} < {table\_folder}/{table\_name}.tsv"'

**remove\_model** (*d*)

Remove a model from the database.

Do this if you realized that the model you built is incorrect or that some of the data is missing.

**Raises** `errors.ModelHasMutationsError` – The model you are trying to delete has pre-calculated mutations, so it can't be that bad. Delete those mutations and try again.

**session\_scope** ()

Provide a transactional scope around a series of operations. Enables the following construct: with `self.session_scope()` as `session`:

**sqlite\_table\_filename** = '{table\_folder}/{table\_name}.tsv'

`elaspic.elaspic_database.check_exception` (*exc, valid\_exc*)

`elaspic.elaspic_database.decorate_all_methods` (*decorator*)

Decorate all methods of a class with *decorator*.

`elaspic.elaspic_database.enable_sqlite_foreign_key_checks` (*engine*)

`elaspic.elaspic_database.get_uniprot_base_path` (*d*)

The uniprot id is cut into several chunks to create folders that will hold a manageable number of pdds.

`elaspic.elaspic_database.get_uniprot_domain_path` (*d*)

Return the path to individual domains or domain pairs.

`elaspic.elaspic_database.pickle_dump` (*obj, filename*)

`elaspic.elaspic_database.retry_archive` (*fn*)

Decorator to keep probing the database until you succeed.

`elaspic.elaspic_database.retry_database` (*fn*)

Decorator to keep probing the database until you succeed.

`elaspic.elaspic_database.scinet_cleanup` (*folder, destination, name=None*)

zip and copy the results from the ramdisk to /scratch

## 7.1.8 elaspic.elaspic\_database\_tables module

Created on Thu Jun 11 16:52:31 2015

@author: ostrokach

**class** elaspic.elaspic\_database\_tables.**Domain** (\*\*kwargs)

Bases: sqlalchemy.ext.declarative.api.Base

Profs domain definitions for all proteins in the PDB.

**Columns:**

**cath\_id** Unique id identifying each domain in the PDB. Constructed by concatenating the `pdb_id`, `pdb_chain`, and an index specifying the order of the domain in the chain.

**pdb\_id** The PDB id in which the domain is found.

**pdb\_chain** The PDB chain in which the domain is found.

**pdb\_domain\_def** Domain definitions of the domain, in PDB RESNUM coordinates.

**pdb\_pdbfam\_name** The Profs name of the domain.

**pdb\_pdbfam\_idx** An integer specifying the number of times a domain with domain name `pdb_pdbfam_name` has occurred in this chain up to this point. It is used to make every `(pdb_id, pdb_chain, pdb_pdbfam_name, pdb_pdbfam_idx)` tuple unique.

**domain\_errors** List of errors that occurred when annotating this domain, or when using this domain to make structural homology models.

`cath_id`

`domain_errors`

`pdb_chain`

`pdb_domain_def`

`pdb_id`

`pdb_pdbfam_idx`

`pdb_pdbfam_name`

**class** elaspic.elaspic\_database\_tables.**DomainContact** (\*\*kwargs)

Bases: sqlalchemy.ext.declarative.api.Base

Interactions between Profs domains in the PDB. Only interactions that were predicted to be biologically relevant by `NOXclass` are included in this table.

**Columns:**

**domain\_contact\_id** A unique integer identifying each domain pair.

**cath\_id\_1** Unique id identifying the first interacting domain in the *domain* table.

**cath\_id\_2** Unique id identifying the second interacting domain in the *domain* table.

**min\_interchain\_distance** The closest that any residue in domain one comes to any residue in domain two.

**contact\_volume** The volume covered by contacting residues.

**contact\_surface\_area** The surface area of the contacting regions of the first and second domains.

**atom\_count\_1** The number of atoms in the first domain.

**atom\_count\_2** The number of atoms in the second domain.

**number\_of\_contact\_residues\_1** The number of residues in the first domain that come within 5 of the second domain.

**number\_of\_contact\_residues\_2** The number of residues in the second domain that come withing 5 of the first domain.

**contact\_residues\_1** A list of all residues in the first domain that come within 5 of the second domain. The residue number corresponds to the position of the residue in the domain.

**contact\_residues\_2** A list of all residues in the second domain that come within 5 of the first domain. The residue number corresponds to the position of the residue in the domain.

**crystal\_packing** The probability that the interaction is a crystallization artifacts, as defined by [NOXclass](#).

**domain\_contact\_errors** List of errors that occurred when annotating this domain pair, or when using this domain as a template for making structural homology models.

`atom_count_1`

`atom_count_2`

`cath_id_1`

`cath_id_2`

`contact_residues_1`

`contact_residues_2`

`contact_surface_area`

`contact_volume`

`crystal_packing`

`domain_1`

`domain_2`

`domain_contact_errors`

`domain_contact_id`

`min_interchain_distance`

`number_of_contact_residues_1`

`number_of_contact_residues_2`

**class** `elaspic.elaspic_database_tables.Provean` (\*\*kwargs)

Bases: `sqlalchemy.ext.declarative.api.Base`

Description of the [Provean](#) supporting set calculated for a protein sequence. The construction of a supporting set is the most lengthy step in running Provean. Therefore, the supporting set is precalculated and stored for every protein sequence.

**Columns:**

**uniprot\_id** The uniprot id of the protein.

**provean\_supset\_filename** The filename of the Provean supporting set. The supporting set contains the ids and sequences of all proteins in the NCBI nr database that are used by Provean to construct a multiple sequence alignment for the given protein.

**provean\_supset\_length** The number of sequences in Provean supporting set.

**provean\_errors** List of errors that occurred while the Provean supporting set was being calculated.

**provean\_date\_modified** Date and time that this row was last modified.

**provean\_date\_modified**

**provean\_errors**

**provean\_supset\_filename**

**provean\_supset\_length**

**uniprot\_id**

**uniprot\_sequence**

**class** `elaspic.elaspic_database_tables.UniprotDomain (**kwargs)`

Bases: `sqlalchemy.ext.declarative.api.Base`

Pfam domain definitions for proteins in the *uniprot\_sequence* table. This table was obtained by downloading Pfam domain definitions for all known proteins from the [SIMAP](#) website, and mapping the protein sequence to uniprot using the MD5 hash of each sequence.

**Columns:**

**uniprot\_domain\_id** Unique id identifying each domain.

**uniprot\_id** The uniprot id of the protein containing the domain.

**pdbfam\_name** The Profs name of the domain. In most cases this will be equivalent to the Pfam name of the domain.

**pdbfam\_idx** The index of the Profs domain. `pdbfam_idx` ranges from 1 to the number of domains with the name `pdbfam_name` in the given protein. The `(pdbfam_name, pdbfam_idx)` tuple uniquely identifies each domain.

**pfam\_clan** The Pfam clan to which this Profs domain belongs.

**alignment\_def** Alignment domain definitions of the Profs domain. This field is obtained by removing gaps in the `alignment_subdefs` column.

**pfam\_names** Pfam names of all Pfam domains that were combined to create the given Profs domain.

**alignment\_subdefs** Comma-separated list of domain definitions for all Pfam domains that were merged to create the given Profs domain.

**path\_to\_data** Location for storing homology models, mutation results, and all other data that are relevant to this domain. This path is prefixed by *archive\_dir*.

**IS\_TRAINING\_SCHEMA = False**

**alignment\_def**

**alignment\_subdefs**

**path\_to\_data**

**pdbfam\_idx**

**pdbfam\_name**

**pfam\_clan**

**pfam\_names**

**uniprot\_domain\_id**

**uniprot\_id**

**uniprot\_sequence**

**class** `elaspic.elaspic_database_tables.UniprotDomainModel` (\*\*kwargs)

Bases: `sqlalchemy.ext.declarative.api.Base`

Homology models for templates in the *uniprot\_domain\_template* table.

**Columns:**

**uniprot\_domain\_id** An integer which uniquely identifies each uniprot domain in the *uniprot\_domain* table.

**model\_errors** List of errors that occurred when making the homology model.

**alignment\_filename** The name of the alignment that was given to Modeller when making the homology model.

**model\_filename** The name of the homology model that was produced by Modeller.

**chain** The chain that contains the domain in question in the homology (this is now set to 'A' in all models).

**norm\_dope** Normalized DOPE score of the model (lower is better).

**sasa\_score** Comma-separated list of the percent solvent-accessible surface area for each residue.

**m\_date\_modified** The date and time when this row was last modified.

**model\_domain\_def** Domain definitions for the region of the domain that is covered by the structural template.

In most cases, this field is identical to the `domain_def` field in the *uniprot\_domain\_template* table. However, it sometimes happens that the best Profs structural template only covers a fraction of the Pfam domain. In that case, the `alignment_def` column in the *uniprot\_domain* table, and the `domain_def` column in the *uniprot\_domain\_template* table, will contain the original Pfam domain definitions, and the `model_domain_def` column will contain domain definitions for only the region that is covered by the structural template.

**alignment\_filename**

**chain**

**m\_date\_modified**

**model\_domain\_def**

**model\_errors**

**model\_filename**

**norm\_dope**

**sasa\_score**

**template**

**uniprot\_domain\_id**

**class** `elaspic.elaspic_database_tables.UniprotDomainMutation` (\*\*kwargs)

Bases: `sqlalchemy.ext.declarative.api.Base`

Characterization of mutations introduced into structures in the *uniprot\_domain\_model* table.

**Columns:**

**uniprot\_id** Uniprot ID of the protein that was mutated.

**uniprot\_domain\_id** Unique id which identifies the Profs domain that was mutated in the *uniprot\_domain* table.

**mutation** Mutation that was introduced into the protein, in Uniprot coordinates.

**mutation\_errors** List of errors that occurred while evaluating the mutation.

**model\_filename\_wt** The name of the file which contains the homology model of the domain after the model was relaxed with FoldX but before the mutation was introduced.

**model\_filename\_mut** The name of the file which contains the homology model of the domain after the model was relaxed with FoldX and after the mutation was introduced.

**chain\_modeller** The chain which contains the domain that was mutated in the `model_filename_wt` and the `model_filename_mut` structures.

**mutation\_modeller** The mutation that was introduced into the protein, in PDB RESNUM coordinates. This identifies the mutated residue in the `model_filename_wt` and the `model_filename_mut` structures.

**stability\_energy\_wt** Comma-separated list of scores returned by FoldX for the wildtype protein. The comma-separated list can be converted into a DataFrame with each column clearly labelled using the `elaspic.predictor.format_mutation_features()`. The FoldX energy terms are:

- dg
- backbone\_hbond
- sidechain\_hbond
- van\_der\_waals
- electrostatics
- solvation\_polar
- solvation\_hydrophobic
- van\_der\_waals\_clashes
- entropy\_sidechain
- entropy\_mainchain
- sloop\_entropy
- mloop\_entropy
- cis\_bond
- torsional\_clash
- backbone\_clash
- helix\_dipole
- water\_bridge
- disulfide
- electrostatic\_kon
- partial\_covalent\_bonds
- energy\_ionisation
- entropy\_complex
- number\_of\_residues

**stability\_energy\_mut** Comma-separated list of scores returned by FoldX for the mutant protein. FoldX energy terms are the same as in *stability\_energy\_wt*, but for the mutated amino acid rather than the wildtype.

**physchem\_wt** Physicochemical properties describing the interaction of the wildtype residue with residues on the opposite chain. The terms are:

- number of atoms in interacting residues that have the same charge.
- number of atoms in interacting residues that have an opposite charge.
- number of hydrogen bonds (very rough calculation).
- number of carbons in interacting residues within 4 Å of the mutated residue (rough measure of the van der Waals force).

**physchem\_wt\_ownchain** Physicochemical properties describing the interaction of the wildtype residue with residues on the same chain. The terms are the same as in *physchem\_wt*.

**physchem\_mut** Physicochemical properties describing the interaction of the mutant residue with residues on the opposite chain. The terms are the same as in *physchem\_wt*.

**physchem\_mut\_ownchain** Physicochemical properties describing the interaction of the mutant residue with residues on the same chain. The terms are the same as in *physchem\_wt*.

**matrix\_score** Score assigned to the wt -> mut transition by the BLOSUM substitution matrix.

**secondary\_structure\_wt** Secondary structure of the wildtype residue predicted by *stride*.

**solvent\_accessibility\_wt** Percent solvent accessible surface area of the wildtype residue, predicted by *msms*.

**secondary\_structure\_mut** Secondary structure of the mutated residue predicted by *stride*.

**solvent\_accessibility\_mut** Percent solvent accessible surface area of the mutated residue, predicted by *msms*.

**provean\_score** Score produced by *Provean* for this mutation.

**ddg** Change in the Gibbs free energy of folding that our classifier predicts for this mutation.

**mut\_date\_modified** Date and time that this row was last modified.

`chain_modeller`

`ddg`

`matrix_score`

`model`

`model_filename_mut`

`model_filename_wt`

`mut_date_modified`

`mutation`

`mutation_errors`

`mutation_modeller`

`physchem_mut`

`physchem_mut_ownchain`

`physchem_wt`



**physchem\_wt\_owchain**  
**provean\_score**  
**secondary\_structure\_mut**  
**secondary\_structure\_wt**  
**solvent\_accessibility\_mut**  
**solvent\_accessibility\_wt**  
**stability\_energy\_mut**  
**stability\_energy\_wt**  
**uniprot\_domain\_id**  
**uniprot\_id**

**class** `elaspic.elaspic_database_tables.UniprotDomainPair` (\*\*kwargs)

Bases: `sqlalchemy.ext.declarative.api.Base`

Potentially-interacting pairs of domains for proteins that are known to interact, according to [Hippie](#), [IRefIndex](#), and [Rolland et al. 2014](#).

**Columns:**

**uniprot\_domain\_pair\_id** Unique id identifying each domain-domain interaction.

**uniprot\_domain\_id\_1** Unique id of the first domain.

**uniprot\_domain\_id\_2** Unique id of the second domain.

**rigids** Phased out.

**domain\_contact\_ids** List of unique ids identifying all domain-domain pairs in the PDB, where one domain belongs to the protein containing `uniprot_domain_id_1` and the other domain belongs to the protein containing `uniprot_domain_id_2`. This was used as crystallographic evidence that the two proteins interact.

**path\_to\_data** Location for storing homology models, mutation results, and all other data that is relevant to this domain pair. This path is prefixed by `archive_dir`.

**domain\_contact\_ids**

**path\_to\_data**

**rigids**

**uniprot\_domain\_1**

**uniprot\_domain\_2**

**uniprot\_domain\_id\_1**

**uniprot\_domain\_id\_2**

**uniprot\_domain\_pair\_id**

**uniprot\_id\_1**

**uniprot\_id\_2**

**class** `elaspic.elaspic_database_tables.UniprotDomainPairModel` (\*\*kwargs)

Bases: `sqlalchemy.ext.declarative.api.Base`

Structural models of interactions between pairs of domains in the `uniprot_domain_pair` table.

**Columns:**

**uniprot\_domain\_pair\_id** Unique id identifying each domain-domain interaction.

**model\_errors** List of errors that occurred while making the homology model.

**alignment\_filename\_1** Name of the file containing the alignment of the first domain with its structural template.

**alignment\_filename\_2** Name of the file containing the alignment of the second domain with its structural template.

**model\_filename** Name of the file containing the homology model of the domain-domain interaction created by Modeller.

**chain\_1** Chain containing the first domain in the model specified by `model_filename`.

**chain\_2** Chain containing the second domain in the model specified by `model_filename`.

**norm\_dope** The normalized DOPE score of the model.

**interface\_area\_hydrophobic** Hydrophobic surface area of the interface, calculated using [POPS](#).

**interface\_area\_hydrophilic** Hydrophilic surface area of the interface, calculated using [POPS](#).

**interface\_area\_total** Total surface area of the interface, calculated using [POPS](#).

**interface\_dg** Gibbs free energy of binding for this domain-domain interaction, predicted using [FoldX](#).  
Not implemented yet!

**interacting\_aa\_1** List of amino acid positions in the first domain that are within 5 of the second domain. Positions are specified using uniprot coordinates.

**interacting\_aa\_2** List of amino acids in the second domain that are within 5 of the first domain. Position are specified using uniprot coordinates.

**m\_date\_modified** Date and time that this row was last modified.

**model\_domain\_def\_1** Domain boundaries of the first domain that are covered by the Profs structural template.

**model\_domain\_def\_2** Domain boundaries of the second domain that are covered by the Profs structural template.

`alignment_filename_1`

`alignment_filename_2`

`chain_1`

`chain_2`

`interacting_aa_1`

`interacting_aa_2`

`interface_area_hydrophilic`

`interface_area_hydrophobic`

`interface_area_total`

`interface_dg`

`m_date_modified`

`model_domain_def_1`

`model_domain_def_2`

**model\_errors**  
**model\_filename**  
**norm\_dope**  
**template**  
**uniprot\_domain\_pair\_id**

**class** `elaspic.elaspic_database_tables.UniprotDomainPairMutation` (\*\*kwargs)  
 Bases: `sqlalchemy.ext.declarative.api.Base`

Characterization of interface mutations introduced into structures in the *uniprot\_domain\_pair\_model* table.

**Columns:**

**uniprot\_id** Uniprot ID of the protein that is being mutated.  
**uniprot\_domain\_pair\_id** Unique id identifying each domain-domain interaction.  
**mutation** Mutation for which the  $\Delta\Delta G$  score is being predicted, specified in Uniprot coordinates.  
**mutation\_errors** List of errors obtained when evaluating the impact of the mutation.  
**model\_filename\_wt** Filename of the homology model relaxed by FoldX but containing the wildtype residue.  
**model\_filename\_mut** Filename of the homology model relaxed by FoldX and containing the mutated residue.  
**chain\_modeller** Chain containing the domain that was mutated, in homology models specified by `model_filename_wt` and `model_filename_mut`.  
**mutation\_modeller** Mutation for which the  $\Delta\Delta G$  score is being predicted, specified in PDB RESNUM coordinates.  
**analyse\_complex\_energy\_wt** Comma-separated list of FoldX scores describing the effect of the wildtype residue on the stability of the protein domain.  
**stability\_energy\_wt** Comma-separated list of FoldX scores describing the effect of the wildtype residue on protein-protein interaction interface.  
**analyse\_complex\_energy\_mut** Comma-separated list of FoldX scores describing the effect of the mutated residue on the stability of the protein domain.  
**stability\_energy\_mut** Comma-separated list of FoldX scores describing the effect of the mutated residue on protein-protein interaction interface.  
**physchem\_wt** Comma-separated list of physicochemical properties describing the interaction between the wildtype residue and other residues on the opposite chain.  
**physchem\_wt\_ownchain** Comma-separated list of physicochemical properties describing the interaction between the wildtype residue and other residues on the same chain.  
**physchem\_mut** Comma-separated list of physicochemical properties describing the interaction between the mutated residue and other residues on the opposite chain.  
**physchem\_mut\_ownchain** Comma-separated list of physicochemical properties describing the interaction between the mutated residue and other residues on the same chain.  
**matrix\_score** Score assigned to the wt -> mut transition by the BLOSUM substitution matrix.  
**secondary\_structure\_wt** Secondary structure of the wildtype residue, predicted by `stride`.  
**solvent\_accessibility\_wt** Percent solvent accessible surface area of the wildtype residue, predicted by `msms`.

**secondary\_structure\_mut** Secondary structure of the mutated residue, predicted by [stride](#).

**solvent\_accessibility\_mut** Percent solvent accessible surface area of the mutated residue, predicted by [msms](#).

**contact\_distance\_wt** Shortest distance between the wildtype residue and a residue on the opposite chain.

**contact\_distance\_mut** Shortest distance between the mutated residue and a residue on the opposite chain.

**provean\_score** [Provean](#) score for this mutation.

**ddg** Predicted change in Gibbs free energy of binding caused by this mutation.

**mut\_date\_modified** Date and time when this row was last modified.

**analyse\_complex\_energy\_mut**  
**analyse\_complex\_energy\_wt**  
**chain\_modeller**  
**contact\_distance\_mut**  
**contact\_distance\_wt**  
**ddg**  
**matrix\_score**  
**model**  
**model\_filename\_mut**  
**model\_filename\_wt**  
**mut\_date\_modified**  
**mutation**  
**mutation\_errors**  
**mutation\_modeller**  
**physchem\_mut**  
**physchem\_mut\_ownchain**  
**physchem\_wt**  
**physchem\_wt\_ownchain**  
**provean\_score**  
**secondary\_structure\_mut**  
**secondary\_structure\_wt**  
**solvent\_accessibility\_mut**  
**solvent\_accessibility\_wt**  
**stability\_energy\_mut**  
**stability\_energy\_wt**  
**uniprot\_domain\_pair\_id**  
**uniprot\_id**

`class` `elaspic.elaspic_database_tables.UniprotDomainPairTemplate` (\*\*kwargs)  
 Bases: `sqlalchemy.ext.declarative.api.Base`

Structural templates for pairs of domains in the `uniprot_domain_pair` table.

**Columns:**

**uniprot\_domain\_pair\_id** Unique id identifying each domain-domain interaction.

**domain\_contact\_id** Unique id of the domain pair in the `domain_contact` table that was used as a template for the modelled domain pair.

**cath\_id\_1** Unique id of the structural template for the first domain.

**cath\_id\_2** Unique id of the structural template for the second domain.

**identical\_1** Fraction of residues in the Blast alignment of the first domain to its template that are *identical*.

**conserved\_1** Fraction of residues in the Blast alignment of the first domain to its template that are *conserved*.

**coverage\_1** Fraction of the first domain that is covered by the blast alignment.

**score\_1** Score obtained by multiplying `identical_1` by `coverage_1`.

**identical\_if\_1** Fraction of interface residues<sup>1</sup> that are *identical* in the Blast alignment of the first domain.

**conserved\_if\_1** Fraction of interface residues<sup>1</sup> that are *conserved* in the Blast alignment of the first domain.

**coverage\_if\_1** Fraction of interface residues<sup>1</sup> that are *covered* by the Blast alignment of the first domain.

**score\_if\_1** Score obtained by combining `identical_if_1` and `coverage_if_1` using (7.1).

**identical\_2** Fraction of residues in the Blast alignment of the second domain to its template that are *identical*.

**conserved\_2** Fraction of residues in the Blast alignment of the second domain to its template that are *conserved*.

**coverage\_2** Fraction of the second domain that is covered by the blast alignment.

**score\_2** Score obtained by multiplying `identical_2` by `coverage_2`.

**identical\_if\_2** Fraction of interface residues<sup>1</sup> that are *identical* in the Blast alignment of the second domain.

**conserved\_if\_2** Fraction of interface residues<sup>1</sup> that are *conserved* in the Blast alignment of the second domain.

**coverage\_if\_2** Fraction of interface residues<sup>1</sup> that are *covered* by the Blast alignment of the second domain.

**score\_if\_2** Score obtained by combining `identical_if_2` and `coverage_if_2` using (7.1).

**score\_total** The product of `score_1` and `score_2`.

**score\_if\_total** The product of `score_if_1` and `score_if_2`.

**score\_overall** The product of `score_total` and `score_if_total`. This is the score that was used to select the best Profs domain pair to be used as a template.

**t\_date\_modified** The date and time when this row was last updated.

**template\_errors** List of errors that occurred while looking for the structural template.

<sup>1</sup> Interface residues are defined as residues that are within 5 of the partner domain.

`cath_id_1`  
`cath_id_2`  
`conserved_1`  
`conserved_2`  
`conserved_if_1`  
`conserved_if_2`  
`coverage_1`  
`coverage_2`  
`coverage_if_1`  
`coverage_if_2`  
`domain_1`  
`domain_2`  
`domain_contact`  
`domain_contact_id`  
`domain_pair`  
`identical_1`  
`identical_2`  
`identical_if_1`  
`identical_if_2`  
`score_1`  
`score_2`  
`score_if_1`  
`score_if_2`  
`score_if_total`  
`score_overall`  
`score_total`  
`t_date_modified`  
`template_errors`  
`uniprot_domain_pair_id`

**class** `elaspic.elaspic_database_tables.UniprotDomainTemplate` (\*\*kwargs)

Bases: `sqlalchemy.ext.declarative.api.Base`

Structural templates for domains in the *uniprot\_domain* table. Lists PDB crystal structures that will be used for making homology models.

**Columns:**

**uniprot\_domain\_id** An integer which uniquely identifies each uniprot domain in the *uniprot\_domain* table.

**template\_errors** List of errors that occurred during the process for finding the template.

**cath\_id** The unique id identifying the structural template of the domain.

**domain\_start** The Uniprot position of the first amino acid of the Profs domain.

**domain\_end** The Uniprot position of the last amino acid of the Profs domain.

**domain\_def** Profs domain definitions for domains with structural templates. Domain definitions in this column are different from domain definitions in the `alignment_def` column of the *uniprot\_domain* table in that they have been expanded to match domain boundaries of the Profs structural template, identified by the `cath_id`.

**alignment\_identity** Percent identity of the domain to its structural template.

**alignment\_coverage** Percent coverage of the domain to its structural template.

**alignment\_score** A score obtained by combining `alignment_identity` (*SeqId*) and `alignment_coverage` (*Cov*) using the following equation, as described by Mosca et al.:

$$Score = 0.95 \cdot \frac{SeqId}{100} \cdot \frac{Cov}{100} + 0.05 \cdot \frac{Cov}{100} \quad (7.1)$$

**t\_date\_modified** The date and time when this row was last modified.

`alignment_coverage`

`alignment_identity`

`alignment_score`

`cath_id`

`domain`

`domain_def`

`domain_end`

`domain_start`

`t_date_modified`

`template_errors`

`uniprot_domain`

`uniprot_domain_id`

**class** `elaspic.elaspic_database_tables.UniprotSequence` (\*\*kwargs)

Bases: `sqlalchemy.ext.declarative.api.Base`

Protein sequences from the Uniprot KB, obtained by parsing *uniprot\_sprot\_fasta.gz*, *uniprot\_trembl\_fasta.gz*, and *homo\_sapiens\_variation.txt* files from the [Uniprot ftp site](#).

**Columns:**

**db** The database to which the protein sequence belongs. Possible values are *sp* for SwissProt and *tr* for TrEMBL.

**uniprot\_id** The uniprot id of the protein.

**uniprot\_name** The uniprot name of the protein.

**protein\_name** The protein name.

**organism\_name** Name of the organism in which this protein is found.

**gene\_name** Name of the gene that codes for this protein sequence.

**protein\_existence** Evidence for the existence of the protein:

1. Experimental evidence at protein level
2. Experimental evidence at transcript level
3. Protein inferred from homology
4. Protein predicted
5. Protein uncertain

**sequence\_version** Version of the protein amino acid sequence.

**uniprot\_sequence** Amino acid sequence of the protein.

**db**

**gene\_name**

**organism\_name**

**protein\_existence**

**protein\_name**

**sequence\_version**

**uniprot\_id**

**uniprot\_name**

**uniprot\_sequence**

`elaspic.elaspic_database_tables.get_db_specific_param(key)`

`elaspic.elaspic_database_tables.get_table_args(table_name, index_columns=[], db_specific_params=[])`

Returns a tuple of additional table arguments.

## 7.1.9 elaspic.elaspic\_model module

### 7.1.10 elaspic.elaspic\_predictor module

Created on Wed Sep 30 16:54:21 2015

@author: strokach

**class** `elaspic.elaspic_predictor.Predictor`

Bases: `object`

**feature\_name\_conversion** = {'normDOPE': 'norm\_dope', 'seq\_id\_avg': 'alignment\_identity'}

**score** (*df, core\_or\_interface*)

**Parameters** *df* (*DataFrame*) – One or more rows with all data required to predict  $\Delta\Delta G$  score. Like something that you would get when you join the appropriate rows in the database.

**Returns** *df* – Same as the input dataframe, except with one additional column: *ddg*.

**Return type** `Dataframe`



`elaspic.elaspic_predictor.convert_features_to_differences` (*df*, *keep\_mut=False*)

Creates a new set of features (ending in *\_change*) that describe the difference between values of the wildtype (features ending in *\_wt*) and mutant (features ending in *\_mut*) features. If *keep\_mut* is *False*, removes all mutant features (features ending in *\_mut*).

`elaspic.elaspic_predictor.format_mutation_features` (*feature\_df*, *core\_or\_interface*)

Converts columns containing comma-separated lists of FoldX features and physicochemical features into a DataFrame where each feature has its own column.

#### Parameters

- **feature\_df** (*DataFrame*) – A pandas DataFrame containing a subset of rows from the *uniprot\_domain\_mutation* or the *uniprot\_domain\_pair\_mutation* tables.
- **core\_or\_interface** (*int or str*) – If 0 or ‘core’, the *feature\_df* DataFrame contains columns from the *uniprot\_domain\_mutation* table. If 1 or ‘interface’, the *feature\_df* DataFrame contains columns from the *uniprot\_domain\_pair\_mutation* table.

**Returns** Contains the same data as *feature\_df*, but with columns containing comma-separated lists of features converted to columns containing a single feature each.

**Return type** DataFrame

### 7.1.11 elaspic.elaspic\_sequence module

**class** `elaspic.elaspic_sequence.Sequence` (*sequence\_file*, *provean\_supset\_file=None*)

Bases: `object`

Class for calculating sequence level features.

**mutate** (*mutation*)

**provean\_supset\_exists**

**provean\_supset\_file**

**result**

**score\_pairwise** (*seq1*, *seq2*, *matrix=None*, *gap\_s=None*, *gap\_e=None*)

Get the BLOSUM (or what ever matrix is given) score.

`elaspic.elaspic_sequence.convert_basestring_to_seqrecord` (*sequence*, *sequence\_id='id'*)

`elaspic.elaspic_sequence.download_uniport_sequence` (*uniprot\_id*, *output\_dir*)

### 7.1.12 elaspic.errors module

**exception** `elaspic.errors.AlignmentNotFoundError` (*save\_path*, *alignment\_filename*)

Bases: `Exception`

**exception** `elaspic.errors.Archive7zipError` (*result*, *error\_message*, *return\_code*)

Bases: `Exception`

**exception** `elaspic.errors.Archive7zipFileNotFoundError` (*result*, *error\_message*, *return\_code*)

Bases: `elaspic.errors.Archive7zipError`

**exception** `elaspic.errors.ChainsNotInteractingError`

Bases: `Exception`

**exception** `elaspic.errors.DataError`

Bases: Exception

**exception** `elaspic.errors.FoldXAAMismatchError`

Bases: Exception

**exception** `elaspic.errors.FoldxError`

Bases: Exception

**exception** `elaspic.errors.InterfaceMismatchError`

Bases: Exception

**exception** `elaspic.errors.LowIdentity`

Bases: Exception

**exception** `elaspic.errors.MSMSError`

Bases: Exception

**exception** `elaspic.errors.ModelHasMutationsError`

Bases: Exception

Don't delete a model that has precalculated mutations!

**exception** `elaspic.errors.ModellerError`

Bases: Exception

**exception** `elaspic.errors.MutationMismatchError`

Bases: Exception

**exception** `elaspic.errors.MutationOutsideDomainError`

Bases: Exception

**exception** `elaspic.errors.MutationOutsideInterfaceError`

Bases: Exception

**exception** `elaspic.errors.NoModelFoundError`

Bases: Exception

**exception** `elaspic.errors.NoSequenceFound`

Bases: Exception

**exception** `elaspic.errors.NoTemplatesFoundError`

Bases: Exception

**exception** `elaspic.errors.PDBChainError`

Bases: Exception

**exception** `elaspic.errors.PDBDomainDefsError`

Bases: Exception

PDB domain definitions not found in the pdb file

**exception** `elaspic.errors.PDBEmptySequenceError`

Bases: Exception

One of the sequences is missing from the alignment. The most likely cause is that the alignment domain definitions were incorrect.

**exception** `elaspic.errors.PDBError`

Bases: Exception

**exception** `elaspic.errors.PDBNotFoundError`

Bases: Exception

**exception** `elaspic.errors.PopsError` (*message, pdb, chains*)

Bases: Exception

**exception** `elaspic.errors.ProteinDefinitionError`

Bases: Exception

**exception** `elaspic.errors.ProveanError`

Bases: Exception

**exception** `elaspic.errors.ProveanResourceError` (*message, child\_process\_group\_id*)

Bases: Exception

**exception** `elaspic.errors.ResourceError`

Bases: Exception

**exception** `elaspic.errors.TcoffeeBlastError` (*result, error, alignInFile, system\_command*)

Bases: Exception

**exception** `elaspic.errors.TcoffeeError` (*result, error, alignInFile, system\_command*)

Bases: Exception

**exception** `elaspic.errors.TcoffeePDBidError` (*result, error, alignInFile, system\_command*)

Bases: Exception

**exception** `elaspic.errors.TemplateCoreError`

Bases: Exception

**exception** `elaspic.errors.TemplateInterfaceError`

Bases: Exception

**exception** `elaspic.errors.WrongConfigKeyError`

Bases: Exception

### 7.1.13 elaspic.helper module

**class** `elaspic.helper.WritableObject` (*logger*)

Bases: object

A class for collecting all the print statements from modeller in order to redirect them to the logger later on

**write** (*string*)

**class** `elaspic.helper.color`

Bases: object

**BLUE** = '\x1b[94m'

**BOLD** = '\x1b[1m'

**CYAN** = '\x1b[96m'

**DARKCYAN** = '\x1b[36m'

**END** = '\x1b[0m'

**FAIL** = '\x1b[91m'

**GREEN** = '\x1b[92m'

**HEADER** = '\x1b[95m'

**OKBLUE** = '\x1b[94m'

**OKGREEN** = '\x1b[92m'

**PURPLE** = '\x1b[95m'

**RED** = '\x1b[91m'

**UNDERLINE** = '\x1b[4m'

**WARNING** = '\x1b[93m'

**YELLOW** = '\x1b[93m'

`elaspic.helper.decode_aa_list` (*interface\_aa*)

`elaspic.helper.decode_domain_def` (*domains*, *merge=True*, *return\_string=False*)

Unlike `split_domain()`, this function returns a tuple of tuples of strings, preserving letter numbering (e.g. 10B)

`elaspic.helper.decode_text_as_list` (*list\_string*)

Uses the database convention to decode a string, describing domain boundaries of multiple domains, as a list of lists.

`elaspic.helper.encode_domain` (*domains*, *merged=True*)

`elaspic.helper.encode_list_as_text` (*list\_of\_lists*)

Uses the database convention to encode a list of lists, describing domain boundaries of multiple domains, as a string.

`elaspic.helper.get_echo` (*system\_constant*)

`elaspic.helper.get_hostname` ()

`elaspic.helper.get_lock` (*name*)

`elaspic.helper.get_logger` (*do\_debug=True*, *logger\_filename=None*)

`elaspic.helper.get_path_to_current_file` ()

Find the location of the file that is being executed

`elaspic.helper.get_username` ()

`elaspic.helper.get_which` (*bin\_name*)

`elaspic.helper.kill_child_process` (*child\_process*)

`elaspic.helper.lock` (*fn*)

Allow only a single instance of function *fn*, and save results to a lock file.

`elaspic.helper.log_print_statements` (*logger*)

Channel print statements to the debug logger

`elaspic.helper.make_tarfile` (*output\_filename*, *source\_dir*)

`elaspic.helper.open_exclusively` (*filename*, *mode='a'*)

`elaspic.helper.print_heartbeats` ()

`elaspic.helper.row2dict` (*row*)

`elaspic.helper.run_subprocess` (*system\_command*, *\*\*popen\_argvars*)

`elaspic.helper.run_subprocess_locally` (*working\_path*, *system\_command*, *\*\*popen\_argvars*)

`elaspic.helper.slugify` (*filename\_string*)

`elaspic.helper.subprocess_check_output` (*system\_command*, *\*\*popen\_argvars*)

`elaspic.helper.subprocess_check_output_locally` (*working\_path*, *system\_command*, *\*\*popen\_argvars*)

`elaspic.helper.subprocess_communicate` (*child\_process*)

`elaspic.helper.switch_paths` (*working\_path*)

`elaspic.helper.underline` (*print\_string*)

### 7.1.14 `elaspic.local_pipeline` module

### 7.1.15 `elaspic.machine_learning` module

`elaspic.machine_learning.cross_validate_predictor` (*data, features, clf\_options, output\_filename=None*)

`elaspic.machine_learning.get_final_predictor` (*data, features, options*)  
Train a predictor using the entire dataset.

`elaspic.machine_learning.write_row_to_file` (*results, output\_filename*)  
TODO: Add a datetime column to each written row.

### 7.1.16 `elaspic.pipeline` module

**class** `elaspic.pipeline.Foo`  
Bases: `object`

`run()`

**class** `elaspic.pipeline.Pipeline` (*configurations*)  
Bases: `object`

`elaspic.pipeline.execute_and_remember` (*f*)  
Some basic memoizer.

`elaspic.pipeline.lock` (*fn*)  
Allow only a single instance of function *fn*, and save results to a lock file.

### 7.1.17 `elaspic.structure_analysis` module

**class** `elaspic.structure_analysis.AnalyzeStructure` (*pdb\_file, working\_dir, vdw\_distance=5.0, min\_contact\_distance=4.0*)  
Bases: `object`

Runs the program pops to calculate the interface size of the complexes This is done by calculating the surface of the complex and the seperated parts. The interface is then given by the subtracting.

`__call__` (*chain\_id, mutation, chain\_id\_other=None*)  
Calculate all properties.

`get_dssp()`  
Not used because crashes on server.

`get_interchain_distances` (*pdb\_chain=None, pdb\_mutation=None, cutoff=None*)

`get_interface_area` (*chain\_ids*)

`get_physi_chem` (*chain\_id, mutation*)  
Return the atomic contact vector, that is, counting how many interactions between charged, polar or “carbon” residues there are. The “carbon” interactions give you information about the Van der Waals packing of the residues. Comparing the wildtype vs. the mutant values is used in the machine learning algorithm.

'mutation' is of the form: 'A16' where A is the chain identifier and 16 the residue number (in pdb numbering) of the mutation chainIDs is a list of strings with the chain identifiers to be used if more than two chains are given, the chains not containing the mutation are considered as "opposing" chain

**get\_sasa** (*program\_to\_use='pops'*)  
Get Solvent Accessible Surface Area scores.

---

**Note:** deprecated

Use python:fn:*get\_seasa* instead.

---

**get\_seasa** ()

**get\_secondary\_structure** ()

**get\_stride** ()

**get\_structure\_file** (*chains, ext='.pdb'*)

**working\_dir** = None

Folder with all the binaries (i.e. *.analyze\_structure*)

### 7.1.18 elaspic.structure\_tools module

**class** elaspic.structure\_tools.**MMCIFParserMod** (*temp\_dir*)

Bases: Bio.PDB.MMCIFParser.MMCIFParser

**get\_structure** (*structure\_id, gzip\_fh*)

Altered *get\_structure* method which accepts gzip file handles as input.

**class** elaspic.structure\_tools.**SelectChains** (*chain\_letters, ns\_chain\_letters=None, ns=None, r\_cutoff=None*)

Bases: Bio.PDB.PDBIO.Select

Only accept the specified chains when saving.

**accept\_residue** (*residue*)

**class** elaspic.structure\_tools.**StructureParser** (*pdb\_file, chain\_ids=None, domain\_defs=[]*)

Bases: *object*

**pdb\_id**

—

**domain\_boundaries**

*list of lists of lists*

Elements in the outer list correspond to domains in each chain of the pdb. Elements of the inner list contain the start and end of each fragment of each domain. For example, if there is only one chain with pdb domain boundaries 1-10:20-45, this would correspond to domain\_boundaries [[[1,10],[20,45]]].

**extract** ()

Extract the wanted chains out of the PDB file. Removes water atoms and selects the domain regions (i.e. selects only those parts of the domain that are within the domain boundaries specified).

**get\_chain\_seqres\_sequence** (*chain\_id, \*args, \*\*varargs*)

Call *get\_chain\_seqres\_sequence* using chain with id *chain\_id*

**get\_chain\_sequence\_and\_numbering** (*chain\_id, \*args, \*\*varargs*)

Call *get\_chain\_sequence\_and\_numbering* using chain with id *chain\_id*

**save\_sequences** (*output\_dir=''*)

**save\_structure** (*output\_dir*='', *remove\_disordered*=False)

`elaspic.structure_tools.calculate_distance` (*atom\_1*, *atom\_2*, *cutoff*=None)

Calculate the distance between two points in 3D space.

**Parameters** *cutoff* (*float, optional*) – The maximum distance allowable between two points.

`elaspic.structure_tools.chain_is_hetatm` (*chain*)

Return True if the chain is made up entirely of HETATMs.

`elaspic.structure_tools.convert_aa` (*aa*, *quiet*=False)

Convert amino acids from three letter code to one letter code or vice versa

**Note:** Deprecated!

Use `''.join(AAA_DICT[aaa] for aaa in aa)` and `''.join(A_DICT[a] for a in aa)`.

`elaspic.structure_tools.convert_position_to_resid` (*chain*, *positions*, *do-main\_def\_tuple*=None)

Convert *mutation\_domain* to *mutation\_modeller*. In *mutation\_modeller*, the first amino acid in a chain may start with something other than 1.

`elaspic.structure_tools.convert_resnum_alphanumeric_to_numeric` (*resnum*)

Convert residue numbering that has letters (i.e. 1A, 1B, 1C...) to residue numbering without letters (i.e. 1, 2, 3...).

**Note:** Deprecated!

Use `get_chain_sequence_and_numbering()`.

`elaspic.structure_tools.download_pdb_file` (*pdb\_id*, *output\_dir*)

Move PDB structure to the local working directory.

`elaspic.structure_tools.euclidean_distance` (*a*, *b*)

Calculate the Euclidean distance between two lists or tuples of arbitrary length.

`elaspic.structure_tools.get_aa_residues` (*chain*)

`elaspic.structure_tools.get_chain_seqres_sequence` (*chain*, *aa\_only*=False)

Get the amino acid sequence for the construct coding for the given chain.

Extracts a sequence from a PDB file. Usefull when interested in the sequence that was used for crystallization and not the ATOM sequence.

**Parameters** *aa\_only* (*bool*) – If *aa\_only* is set to *False*, selenomethionines will be included in the sequence. See: <http://biopython.org/DIST/docs/api/Bio.PDB.Polypeptide-module.html>.

`elaspic.structure_tools.get_chain_sequence_and_numbering` (*chain*, *do-main\_def\_tuple*=None, *include\_hetatms*=False)

Get the amino acid sequence and a list of residue ids for the given chain.

**Parameters** *chain* (*Bio.PDB.Chain.Chain*) – The chain for which to get the amino acid sequence and numbering.

`elaspic.structure_tools.get_interacting_residues` (*model*, *r\_cutoff*=5, *skip\_hetatm\_chains*=True)

Returns all interactions between residues on different chains in *model*.

**Returns** A dictionary of interactions between chains *i* (0..n-1) and *j* (i+1..n). Keys are (chain\_idx, chain\_id, residue\_idx, residue\_resnum, residue\_amino\_acid) tuples. (e.g. (0, 'A', 0, 'O', 'M'), (0, 1, '2', 'K'), ...) Values are a list of tuples having the same format as the keys.

**Return type** dict

You can reverse the order of keys and values like this:

```
complement = dict()
for key, values in get_interacting_chains(model):
    for value in values:
        complement.setdefault(value, set()).add(key)
```

You can get a list of all interacting chains using this command:

```
{(key[0], value[0])
 for (key, values) in get_interacting_chains(model).items()
 for value in values}
```

`elaspic.structure_tools.get_interactions(model, chain_id, r_cutoff=6)`

`elaspic.structure_tools.get_interactions_between_chains(model, chain_id_1, chain_id_2, r_cutoff=6)`

Calculate interactions between residues in `pdb_chain_1` and `pdb_chain_2`. An interaction is defines as a pair of residues where at least one pair of atom is closer than `r_cutoff`. The default value for `r_cutoff` is 5 Angstroms.

Deprecated since version 1.0: Use `python:fn:get_interacting_residues` instead. It gives you both the residue index and the resnum.

**Returns** Keys are (residue\_number, residue\_amino\_acid) tuples (e.g. ('0', 'M'), ('1', 'Q'), ...). Values are lists of (residue\_number, residue\_amino\_acid) tuples. (e.g. [('0', 'M'), ('1', 'Q'), ...]).

**Return type** OrderedDict

`elaspic.structure_tools.get_interactions_between_chains_slow(model, pdb_chain_1, pdb_chain_2, r_cutoff=5)`

Calculate interactions between residues in `pdb_chain_1` and `pdb_chain_2`. An interaction is defines as a pair of residues where at least one pair of atom is closer than `r_cutoff`. The default value for `r_cutoff` is 5 Angstroms.

Deprecated since version 1.0: Use `get_interacting_residues()` instead. It gives you both the residue index and the resnum.

`elaspic.structure_tools.get_pdb`

Parse a pdb file with biopythons PDBParser() and return the structure.

**Parameters**

- **pdb\_code** (*str*) – Four letter code of the PDB file
- **pdb\_path** (*str*) – Biopython pdb structure
- **temp\_dir** (*str, optional, default='/tmp/'*) – Path to the folder for storing temporary files
- **pdb\_type** (*'ent'/'pdb'/'cif', optional, default='ent'*) – The extension of the pdb to use

**Raises** PDBNotFoundError – If the pdb file could not be retrieved from the local (and remote) databases

`elaspic.structure_tools.get_pdb_file(pdb_id, pdb_database_dir, pdb_type='ent')`

Get PDB file from a local mirror of the PDB database.

`elaspic.structure_tools.get_pdb_id(pdb_file)`

`elaspic.structure_tools.get_pdb_parser(pdb_type, temp_dir='/tmp')`

Get PDB parser that can work with structures of the specified type.



`elaspic.structure_tools.get_pdb_structure` (*pdb\_file*, *pdb\_id=None*)

Set QUIET to False to output warnings like incomplete chains etc.

`elaspic.structure_tools.get_structure_sequences` (*file\_or\_structure*, *seqs\_sequence=False*)

Convenience function returning a dictionary of sequences for a given file or a Biopython Structure, Model or Chain.

`elaspic.structure_tools.suppress_logger` (*fn*)

### 7.1.19 Module contents

## 7.2 Indices and tables

- [modindex](#)



## e

elaspic, 53  
elaspic.call\_foldx, 27  
elaspic.call\_tcoffee, 27  
elaspic.conf, 28  
elaspic.elaspic\_database, 28  
elaspic.elaspic\_database\_tables, 31  
elaspic.elaspic\_predictor, 44  
elaspic.elaspic\_sequence, 45  
elaspic.errors, 45  
elaspic.helper, 47  
elaspic.machine\_learning, 49  
elaspic.pipeline, 49  
elaspic.structure\_analysis, 49  
elaspic.structure\_tools, 50



## Symbols

`__call__()` (elaspic.call\_foldx.FoldX method), 27

`__call__()` (elaspic.structure\_analysis.AnalyzeStructure method), 49

## A

`accept_residue()` (elaspic.structure\_tools.SelectChains method), 50

`add_domain()` (elaspic.elaspic\_database.MyDatabase method), 28

`add_domain_errors()` (elaspic.elaspic\_database.MyDatabase method), 28

`add_uniprot_sequence()` (elaspic.elaspic\_database.MyDatabase method), 29

`align()` (elaspic.call\_tcoffee.TCoffee method), 27

`alignment_coverage` (elaspic.elaspic\_database\_tables.UniprotDomainTemplate attribute), 43

`alignment_def` (elaspic.elaspic\_database\_tables.UniprotDomainTemplate attribute), 33

`alignment_filename` (elaspic.elaspic\_database\_tables.UniprotDomainTemplate attribute), 34

`alignment_filename_1` (elaspic.elaspic\_database\_tables.UniprotDomainPairModel attribute), 38

`alignment_filename_2` (elaspic.elaspic\_database\_tables.UniprotDomainPairModel attribute), 38

`alignment_identity` (elaspic.elaspic\_database\_tables.UniprotDomainTemplate attribute), 43

`alignment_score` (elaspic.elaspic\_database\_tables.UniprotDomainPairModel attribute), 43

`alignment_subdefs` (elaspic.elaspic\_database\_tables.UniprotDomainTemplate attribute), 33

AlignmentNotFoundError, 45

`analyse_complex_energy_mut`

(elaspic.elaspic\_database\_tables.UniprotDomainPairMutation attribute), 40

`analyse_complex_energy_wt`

(elaspic.elaspic\_database\_tables.UniprotDomainPairMutation attribute), 40

AnalyzeStructure (class in elaspic.structure\_analysis), 49

Archive7zipError, 45

Archive7zipFileNotFoundError, 45

`archive_dir`, 8

`archive_type`, 7

`atom_count_1` (elaspic.elaspic\_database\_tables.DomainContact attribute), 32

`atom_count_2` (elaspic.elaspic\_database\_tables.DomainContact attribute), 32

## B

`blast_db_dir`, 7

`blast_db_dir_fallback`, 7

BLUE (elaspic.helper.color attribute), 47

BOLD (elaspic.helper.color attribute), 47

## C

`calculate_templates()` (in module elaspic.structure\_tools), 51

`chain_id` (elaspic.elaspic\_database\_tables.DomainTemplate attribute), 31

`chain_is_hetatm` (elaspic.elaspic\_database\_tables.UniprotDomainTemplate attribute), 43

`chain_is_hetatm_1` (elaspic.elaspic\_database\_tables.DomainContact attribute), 32

`chain_is_hetatm_2` (elaspic.elaspic\_database\_tables.UniprotDomainPairTemplate attribute), 41

`chain_is_hetatm_3` (elaspic.elaspic\_database\_tables.DomainContact attribute), 32

`chain_is_hetatm_4` (elaspic.elaspic\_database\_tables.UniprotDomainPairTemplate attribute), 42

`chain_is_hetatm_5` (elaspic.elaspic\_database\_tables.UniprotDomainModel attribute), 34

`chain_1` (elaspic.elaspic\_database\_tables.UniprotDomainPairModel attribute), 38

`chain_2` (elaspic.elaspic\_database\_tables.UniprotDomainPairModel attribute), 38

`chain_is_hetatm()` (in module elaspic.structure\_tools), 51

`chain_modeller` (elaspic.elaspic\_database\_tables.UniprotDomainMutation attribute), 36

`chain_modeller` (elaspic.elaspic\_database\_tables.UniprotDomainPairMutation attribute), 40

- ChainsNotInteractingError, 45
  - check\_exception() (in module elaspic.elaspic\_database), 30
  - clear() (elaspic.conf.Configs method), 28
  - color (class in elaspic.helper), 47
  - Configs (class in elaspic.conf), 28
  - configure\_session() (elaspic.elaspic\_database.MyDatabase method), 29
  - conserved\_1 (elaspic.elaspic\_database\_tables.UniprotDomainPairTemplate attribute), 42
  - conserved\_2 (elaspic.elaspic\_database\_tables.UniprotDomainPairTemplate attribute), 42
  - conserved\_if\_1 (elaspic.elaspic\_database\_tables.UniprotDomainPairTemplate attribute), 42
  - conserved\_if\_2 (elaspic.elaspic\_database\_tables.UniprotDomainPairTemplate attribute), 42
  - contact\_distance\_mut (elaspic.elaspic\_database\_tables.UniprotDomainPairMutation attribute), 40
  - contact\_distance\_wt (elaspic.elaspic\_database\_tables.UniprotDomainPairMutation attribute), 40
  - contact\_residues\_1 (elaspic.elaspic\_database\_tables.DomainContact attribute), 32
  - contact\_residues\_2 (elaspic.elaspic\_database\_tables.DomainContact attribute), 32
  - contact\_surface\_area (elaspic.elaspic\_database\_tables.DomainContact attribute), 32
  - contact\_volume (elaspic.elaspic\_database\_tables.DomainContact attribute), 32
  - convert\_aa() (in module elaspic.structure\_tools), 51
  - convert\_basestring\_to\_seqrecord() (in module elaspic.elaspic\_sequence), 45
  - convert\_features\_to\_differences() (in module elaspic.elaspic\_predictor), 44
  - convert\_position\_to\_resid() (in module elaspic.structure\_tools), 51
  - convert\_resnum\_alphanumeric\_to\_numeric() (in module elaspic.structure\_tools), 51
  - copy() (elaspic.conf.Configs method), 28
  - copy\_data, 6
  - copy\_table\_to\_db() (elaspic.elaspic\_database.MyDatabase method), 29
  - coverage\_1 (elaspic.elaspic\_database\_tables.UniprotDomainPairTemplate attribute), 42
  - coverage\_2 (elaspic.elaspic\_database\_tables.UniprotDomainPairTemplate attribute), 42
  - coverage\_if\_1 (elaspic.elaspic\_database\_tables.UniprotDomainPairTemplate attribute), 42
  - coverage\_if\_2 (elaspic.elaspic\_database\_tables.UniprotDomainPairTemplate attribute), 42
  - create\_database\_tables() (elaspic.elaspic\_database.MyDatabase method), 29
  - cross\_validate\_predictor() (in module elaspic.machine\_learning), 49
  - crystal\_packing (elaspic.elaspic\_database\_tables.DomainContact attribute), 32
  - CYAN (elaspic.helper.color attribute), 47
- ## D
- DARKCYAN (elaspic.helper.color attribute), 47
  - DataError, 45
  - db (elaspic.elaspic\_database\_tables.UniprotSequence attribute), 44
  - db\_database, 7
  - db\_password, 7
  - db\_port, 7
  - db\_schema, 7
  - db\_schema\_uniprot, 7
  - db\_socket, 7
  - db\_type, 7
  - db\_url, 7
  - db\_username, 7
  - ddg (elaspic.elaspic\_database\_tables.UniprotDomainMutation attribute), 40
  - ddg (elaspic.elaspic\_database\_tables.UniprotDomainPairMutation attribute), 40
  - debug, 6
  - decode\_aa\_list() (in module elaspic.helper), 48
  - decode\_domain\_def() (in module elaspic.helper), 48
  - decode\_text\_as\_list() (in module elaspic.helper), 48
  - decorate\_all\_methods() (in module elaspic.elaspic\_database), 30
  - delete\_database\_tables() (elaspic.elaspic\_database.MyDatabase method), 29
  - Domain (class in elaspic.elaspic\_database\_tables), 31
  - domain (elaspic.elaspic\_database\_tables.UniprotDomainTemplate attribute), 43
  - domain\_1 (elaspic.elaspic\_database\_tables.DomainContact attribute), 32
  - domain\_1 (elaspic.elaspic\_database\_tables.UniprotDomainPairTemplate attribute), 42
  - domain\_2 (elaspic.elaspic\_database\_tables.DomainContact attribute), 32
  - domain\_2 (elaspic.elaspic\_database\_tables.UniprotDomainPairTemplate attribute), 42
  - domain\_boundaries (elaspic.structure\_tools.StructureParser attribute), 50
  - domain\_contact (elaspic.elaspic\_database\_tables.UniprotDomainPairTemplate attribute), 42
  - domain\_contact\_errors (elaspic.elaspic\_database\_tables.DomainContact attribute), 32
  - domain\_contact\_id (elaspic.elaspic\_database\_tables.DomainContact attribute), 32
  - domain\_contact\_id (elaspic.elaspic\_database\_tables.UniprotDomainPairTemplate attribute), 42
  - domain\_contact\_ids (elaspic.elaspic\_database\_tables.UniprotDomainPair attribute), 37
  - domain\_def (elaspic.elaspic\_database\_tables.UniprotDomainTemplate attribute), 43

- domain\_end (elaspic.elaspic\_database\_tables.UniprotDomainContact attribute), 43
- domain\_errors (elaspic.elaspic\_database\_tables.DomainContact attribute), 31
- domain\_pair (elaspic.elaspic\_database\_tables.UniprotDomainContact attribute), 42
- domain\_start (elaspic.elaspic\_database\_tables.UniprotDomainContact attribute), 43
- DomainContact (class in elaspic.elaspic\_database\_tables), 31
- download\_pdb\_file() (in module elaspic.structure\_tools), 51
- download\_uniprot\_sequence() (in module elaspic.elaspic\_sequence), 45
- ## E
- elaspic (module), 27
- elaspic.call\_foldx (module), 27
- elaspic.call\_tcoffee (module), 27
- elaspic.conf (module), 28
- elaspic.elaspic\_database (module), 28
- elaspic.elaspic\_database\_tables (module), 31
- elaspic.elaspic\_predictor (module), 44
- elaspic.elaspic\_sequence (module), 45
- elaspic.errors (module), 45
- elaspic.helper (module), 47
- elaspic.machine\_learning (module), 49
- elaspic.pipeline (module), 49
- elaspic.structure\_analysis (module), 49
- elaspic.structure\_tools (module), 50
- enable\_sqlite\_foreign\_key\_checks() (in module elaspic.elaspic\_database), 30
- encode\_domain() (in module elaspic.helper), 48
- encode\_list\_as\_text() (in module elaspic.helper), 48
- END (elaspic.helper.color attribute), 47
- environment variable
- PATH, 8
  - TMPDIR, 6, 8, 28
- euclidean\_distance() (in module elaspic.structure\_tools), 51
- execute\_and\_remember() (in module elaspic.pipeline), 49
- extract() (elaspic.structure\_tools.StructureParser method), 50
- ## F
- FAIL (elaspic.helper.color attribute), 47
- feature\_name\_conversion (elaspic.elaspic\_predictor.Predictor attribute), 44
- FoldX (class in elaspic.call\_foldx), 27
- foldx\_num\_of\_runs, 7
- foldx\_water, 7
- FoldXAAMismatchError, 46
- FoldxError, 46
- File (class in elaspic.pipeline), 49
- format\_mutation\_features() (in module elaspic.elaspic\_predictor), 45
- ## G
- GapTemplate
- gap\_extend, 7
  - gap\_start, 7
- gene\_name (elaspic.elaspic\_database\_tables.UniprotSequence attribute), 44
- get() (elaspic.conf.Configs method), 28
- get\_aa\_residues() (in module elaspic.structure\_tools), 51
- get\_alignment() (elaspic.elaspic\_database.MyDatabase method), 29
- get\_chain\_seqres\_sequence() (elaspic.structure\_tools.StructureParser method), 50
- get\_chain\_seqres\_sequence() (in module elaspic.structure\_tools), 51
- get\_chain\_sequence\_and\_numbering() (elaspic.structure\_tools.StructureParser method), 50
- get\_chain\_sequence\_and\_numbering() (in module elaspic.structure\_tools), 51
- get\_db\_specific\_param() (in module elaspic.elaspic\_database\_tables), 44
- get\_domain() (elaspic.elaspic\_database.MyDatabase method), 29
- get\_domain\_contact() (elaspic.elaspic\_database.MyDatabase method), 29
- get\_dssp() (elaspic.structure\_analysis.AnalyzeStructure method), 49
- get\_echo() (in module elaspic.helper), 48
- get\_engine() (elaspic.elaspic\_database.MyDatabase method), 29
- get\_final\_predictor() (in module elaspic.machine\_learning), 49
- get\_hostname() (in module elaspic.helper), 48
- get\_interacting\_residues() (in module elaspic.structure\_tools), 51
- get\_interactions() (in module elaspic.structure\_tools), 52
- get\_interactions\_between\_chains() (in module elaspic.structure\_tools), 52
- get\_interactions\_between\_chains\_slow() (in module elaspic.structure\_tools), 52
- get\_interchain\_distances() (elaspic.structure\_analysis.AnalyzeStructure method), 49
- get\_interface\_area() (elaspic.structure\_analysis.AnalyzeStructure method), 49
- get\_lock() (in module elaspic.helper), 48
- get\_logger() (in module elaspic.helper), 48
- get\_path\_to\_current\_file() (in module elaspic.helper), 48
- get\_pdb (in module elaspic.structure\_tools), 52
- get\_pdb\_file() (in module elaspic.structure\_tools), 52

[get\\_pdb\\_id\(\)](#) (in module `elaspic.structure_tools`), 52  
[get\\_pdb\\_parser\(\)](#) (in module `elaspic.structure_tools`), 52  
[get\\_pdb\\_structure\(\)](#) (in module `elaspic.structure_tools`), 52  
[get\\_physi\\_chem\(\)](#) (`elaspic.structure_analysis.AnalyzeStructure` method), 49  
[get\\_rows\\_by\\_ids\(\)](#) (`elaspic.elaspic_database.MyDatabase` method), 29  
[get\\_sasa\(\)](#) (`elaspic.structure_analysis.AnalyzeStructure` method), 50  
[get\\_seasa\(\)](#) (`elaspic.structure_analysis.AnalyzeStructure` method), 50  
[get\\_secondary\\_structure\(\)](#) (`elaspic.structure_analysis.AnalyzeStructure` method), 50  
[get\\_stride\(\)](#) (`elaspic.structure_analysis.AnalyzeStructure` method), 50  
[get\\_structure\(\)](#) (`elaspic.structure_tools.MMCIFParserMod` method), 50  
[get\\_structure\\_file\(\)](#) (`elaspic.structure_analysis.AnalyzeStructure` method), 50  
[get\\_structure\\_sequences\(\)](#) (in module `elaspic.structure_tools`), 53  
[get\\_table\\_args\(\)](#) (in module `elaspic.elaspic_database_tables`), 44  
[get\\_temp\\_dir\(\)](#) (in module `elaspic.conf`), 28  
[get\\_uniprot\\_base\\_path\(\)](#) (in module `elaspic.elaspic_database`), 30  
[get\\_uniprot\\_domain\(\)](#) (`elaspic.elaspic_database.MyDatabase` method), 29  
[get\\_uniprot\\_domain\\_pair\(\)](#) (`elaspic.elaspic_database.MyDatabase` method), 29  
[get\\_uniprot\\_domain\\_path\(\)](#) (in module `elaspic.elaspic_database`), 30  
[get\\_uniprot\\_mutation\(\)](#) (`elaspic.elaspic_database.MyDatabase` method), 29  
[get\\_uniprot\\_sequence\(\)](#) (`elaspic.elaspic_database.MyDatabase` method), 29  
[get\\_username\(\)](#) (in module `elaspic.helper`), 48  
[get\\_which\(\)](#) (in module `elaspic.helper`), 48  
[global\\_temp\\_dir](#), 6  
[GREEN](#) (`elaspic.helper.color` attribute), 47

**H**

[HEADER](#) (`elaspic.helper.color` attribute), 47

**I**

[identical\\_1](#) (`elaspic.elaspic_database_tables.UniprotDomainPairTemplate` attribute), 42  
[identical\\_2](#) (`elaspic.elaspic_database_tables.UniprotDomainPairTemplate` attribute), 42  
[identical\\_if\\_1](#) (`elaspic.elaspic_database_tables.UniprotDomainPairTemplate` attribute), 42  
[identical\\_if\\_2](#) (`elaspic.elaspic_database_tables.UniprotDomainPairTemplate` attribute), 42  
[instance](#) (`elaspic.conf.Singleton` attribute), 28  
[interacting\\_aa\\_1](#) (`elaspic.elaspic_database_tables.UniprotDomainPairModel` attribute), 38  
[interacting\\_aa\\_2](#) (`elaspic.elaspic_database_tables.UniprotDomainPairModel` attribute), 38  
[interface\\_area\\_hydrophilic](#) (`elaspic.elaspic_database_tables.UniprotDomainPairModel` attribute), 38  
[interface\\_area\\_hydrophobic](#) (`elaspic.elaspic_database_tables.UniprotDomainPairModel` attribute), 38  
[interface\\_area\\_total](#) (`elaspic.elaspic_database_tables.UniprotDomainPairModel` attribute), 38  
[interface\\_dg](#) (`elaspic.elaspic_database_tables.UniprotDomainPairModel` attribute), 38  
[InterfaceMismatchError](#), 46  
[IS\\_TRAINING\\_SCHEMA](#) (`elaspic.elaspic_database_tables.UniprotDomain` attribute), 33  
[items\(\)](#) (`elaspic.conf.Configs` method), 28

**K**

[keys\(\)](#) (`elaspic.conf.Configs` method), 28  
[kill\\_child\\_process\(\)](#) (in module `elaspic.helper`), 48

**L**

[load\\_db\\_from\\_archive\(\)](#) (`elaspic.elaspic_database.MyDatabase` method), 30  
[lock\(\)](#) (in module `elaspic.helper`), 48  
[lock\(\)](#) (in module `elaspic.pipeline`), 49  
[log\\_print\\_statements\(\)](#) (in module `elaspic.helper`), 48  
[look\\_for\\_interactions](#), 6  
[LowIdentity](#), 46

**M**

[m\\_date\\_modified](#) (`elaspic.elaspic_database_tables.UniprotDomainModel` attribute), 34  
[m\\_date\\_modified](#) (`elaspic.elaspic_database_tables.UniprotDomainPairModel` attribute), 38  
[make\\_tarfile\(\)](#) (in module `elaspic.helper`), 48  
[matrix\\_score](#) (`elaspic.elaspic_database_tables.UniprotDomainMutation` attribute), 36  
[matrix\\_score](#) (`elaspic.elaspic_database_tables.UniprotDomainPairMutation` attribute), 40  
[matrix\\_type](#), 7  
[merge\\_model\(\)](#) (`elaspic.elaspic_database.MyDatabase` method), 30  
[merge\\_mutation\(\)](#) (`elaspic.elaspic_database.MyDatabase` method), 30  
[merge\\_provean\(\)](#) (`elaspic.elaspic_database.MyDatabase` method), 30



merge\_row() (elaspic.elaspic\_database.MyDatabase method), 30  
 MutationOutsideInterfaceError, 46  
 MyDatabase (class in elaspic.elaspic\_database), 28  
 min\_interchain\_distance (elaspic.elaspic\_database\_tables.DomainContact attribute), 32  
 MySQLContact and template (elaspic.elaspic\_database.MyDatabase attribute), 30  
 MMCIFParserMod (class in elaspic.structure\_tools), 50  
 mysql\_load\_table\_template (elaspic.elaspic\_database.MyDatabase attribute), 30  
 model (elaspic.elaspic\_database\_tables.UniprotDomainMutation attribute), 36  
 model (elaspic.elaspic\_database\_tables.UniprotDomainPairMutation attribute), 30  
 model\_domain\_def (elaspic.elaspic\_database\_tables.UniprotDomainModel attribute), 34  
 model\_domain\_def\_1 (elaspic.elaspic\_database\_tables.UniprotDomainPairModel attribute), 38  
 model\_domain\_def\_2 (elaspic.elaspic\_database\_tables.UniprotDomainPairModel attribute), 38  
 model\_errors (elaspic.elaspic\_database\_tables.UniprotDomainModel attribute), 39  
 model\_errors (elaspic.elaspic\_database\_tables.UniprotDomainPairModel attribute), 38  
 model\_filename (elaspic.elaspic\_database\_tables.UniprotDomainModel attribute), 34  
 model\_filename (elaspic.elaspic\_database\_tables.UniprotDomainPairModel attribute), 39  
 model\_filename\_mut (elaspic.elaspic\_database\_tables.UniprotDomainMutation attribute), 36  
 model\_filename\_mut (elaspic.elaspic\_database\_tables.UniprotDomainPairMutation attribute), 40  
 model\_filename\_wt (elaspic.elaspic\_database\_tables.UniprotDomainMutation attribute), 36  
 model\_filename\_wt (elaspic.elaspic\_database\_tables.UniprotDomainPairMutation attribute), 40  
 ModelHasMutationsError, 46  
 modeller\_runs, 7  
 ModellerError, 46  
 MSMSError, 46  
 mut\_date\_modified (elaspic.elaspic\_database\_tables.UniprotDomainMutation attribute), 36  
 mut\_date\_modified (elaspic.elaspic\_database\_tables.UniprotDomainPairMutation attribute), 40  
 mutate() (elaspic.elaspic\_sequence.Sequence method), 45  
 mutation (elaspic.elaspic\_database\_tables.UniprotDomainMutation attribute), 36  
 mutation (elaspic.elaspic\_database\_tables.UniprotDomainPairMutation attribute), 40  
 mutation\_errors (elaspic.elaspic\_database\_tables.UniprotDomainMutation attribute), 36  
 mutation\_errors (elaspic.elaspic\_database\_tables.UniprotDomainPairMutation attribute), 40  
 mutation\_modeller (elaspic.elaspic\_database\_tables.UniprotDomainMutation attribute), 36  
 mutation\_modeller (elaspic.elaspic\_database\_tables.UniprotDomainPairMutation attribute), 40  
 MutationMismatchError, 46  
 MutationOutsideDomainError, 46  
 n\_cores, 6  
 norm\_dope (elaspic.elaspic\_database\_tables.UniprotDomainModel attribute), 34  
 norm\_dope (elaspic.elaspic\_database\_tables.UniprotDomainPairModel attribute), 34  
 NoSequenceFound, 46  
 NoPairModelNotFoundError, 46  
 number\_of\_contact\_residues\_1 (elaspic.elaspic\_database\_tables.DomainContact attribute), 32  
 number\_of\_contact\_residues\_2 (elaspic.elaspic\_database\_tables.DomainContact attribute), 32  
 OKBLUE (elaspic.helper.color attribute), 47  
 OKGREEN (elaspic.helper.color attribute), 47  
 open\_exclusively() (in module elaspic.helper), 48  
 organism\_name (elaspic.elaspic\_database\_tables.UniprotSequence attribute), 44  
**P**  
 path\_to\_data (elaspic.elaspic\_database\_tables.UniprotDomain attribute), 33  
 path\_to\_data (elaspic.elaspic\_database\_tables.UniprotDomainPair attribute), 37  
 pdb\_chain (elaspic.elaspic\_database\_tables.Domain attribute), 31  
 pdb\_dir, 8  
 pdb\_domain\_def (elaspic.elaspic\_database\_tables.Domain attribute), 31  
 pdb\_id (elaspic.elaspic\_database\_tables.Domain attribute), 31  
 pdb\_id (elaspic.structure\_tools.StructureParser attribute), 50  
 pdb\_pdbfam\_idx (elaspic.elaspic\_database\_tables.Domain attribute), 31  
 pdb\_pdbfam\_name (elaspic.elaspic\_database\_tables.Domain attribute), 31  
 PDBChainError, 46  
 PDBDomainDefsError, 46  
 PDBEmptySequenceError, 46  
 PDBError, 46

pdbfam\_idx (elaspic.elaspic\_database\_tables.UniprotDomain attribute), 33  
 attribute), 33  
 provean\_temp\_dir, 6  
 pdbfam\_name (elaspic.elaspic\_database\_tables.UniprotDomain attribute), 33  
 ProveanError, 47  
 ProveanResourceError, 47  
 PDBNotFound, 46  
 psql\_command\_template  
 pfam\_clan (elaspic.elaspic\_database\_tables.UniprotDomain attribute), 33  
 (elaspic.elaspic\_database.MyDatabase attribute), 30  
 pfam\_names (elaspic.elaspic\_database\_tables.UniprotDomain attribute), 33  
 psql\_load\_table\_template  
 (elaspic.elaspic\_database.MyDatabase attribute)  
 physchem\_mut (elaspic.elaspic\_database\_tables.UniprotDomainMutation attribute), 36  
 attribute), 30  
 PURPLE (elaspic.helper.color attribute), 47  
 physchem\_mut (elaspic.elaspic\_database\_tables.UniprotDomainPairMutation attribute), 40  
**R**  
 physchem\_mut\_ownchain  
 read\_configuration\_file() (in module elaspic.conf), 28  
 (elaspic.elaspic\_database\_tables.UniprotDomainMutation attribute), 36  
 read\_database\_configs() (in module elaspic.conf), 28  
 physchem\_mut\_ownchain  
 read\_model\_configs() (in module elaspic.conf), 28  
 (elaspic.elaspic\_database\_tables.UniprotDomainPairMutation attribute), 40  
 read\_sequence\_configs() (in module elaspic.conf), 28  
 physchem\_wt (elaspic.elaspic\_database\_tables.UniprotDomainMutation attribute), 36  
 read\_table() (elaspic.helper.color attribute), 48  
 physchem\_wt (elaspic.elaspic\_database\_tables.UniprotDomainPairMutation attribute), 40  
 remake\_provean\_supset, 6  
 physchem\_wt\_ownchain (elaspic.elaspic\_database\_tables.UniprotDomainPairMutation attribute), 36  
 remove\_model() (elaspic.elaspic\_database.MyDatabase method), 30  
 physchem\_wt\_ownchain (elaspic.elaspic\_database\_tables.UniprotDomainPairMutation attribute), 36  
 resource\_error, 47  
 result (elaspic.elaspic\_sequence.Sequence attribute), 45  
 physchem\_wt\_ownchain (elaspic.elaspic\_database\_tables.UniprotDomainPairMutation attribute), 40  
 retry\_database() (in module elaspic.elaspic\_database), 30  
 pickledump() (in module elaspic.elaspic\_database), 30  
 riggs (elaspic.elaspic\_database\_tables.UniprotDomainPairMutation attribute), 37  
 Pipeline (class in elaspic.pipeline), 49  
 row2dict() (in module elaspic.helper), 48  
 PopsError, 46  
 run() (elaspic.pipeline.Foo method), 49  
 Predictor (class in elaspic.elaspic\_predictor), 44  
 run\_subprocess() (in module elaspic.helper), 48  
 print\_heartbeats() (in module elaspic.helper), 48  
 run\_subprocess\_locally() (in module elaspic.helper), 48  
 protein\_existence (elaspic.elaspic\_database\_tables.UniprotSequence attribute), 44  
**S**  
 protein\_name (elaspic.elaspic\_database\_tables.UniprotSequence attribute), 44  
 sasa\_score (elaspic.elaspic\_database\_tables.UniprotDomainModel attribute), 34  
 ProteinDefinitionError, 47  
 save\_sequences() (elaspic.structure\_tools.StructureParser method), 50  
 Provean (class in elaspic.elaspic\_database\_tables), 32  
 save\_structure() (elaspic.structure\_tools.StructureParser method), 50  
 provean\_date\_modified (elaspic.elaspic\_database\_tables.Provean attribute), 33  
 schema\_version, 7  
 provean\_errors (elaspic.elaspic\_database\_tables.Provean attribute), 33  
 scinet\_cleanup() (in module elaspic.elaspic\_database), 30  
 provean\_score (elaspic.elaspic\_database\_tables.UniprotDomainMutation attribute), 37  
 score() (elaspic.elaspic\_predictor.Predictor method), 44  
 provean\_score (elaspic.elaspic\_database\_tables.UniprotDomainPairMutation attribute), 40  
 score\_1 (elaspic.elaspic\_database\_tables.UniprotDomainPairTemplate attribute), 42  
 provean\_supset\_exists (elaspic.elaspic\_sequence.Sequence attribute), 45  
 score\_2 (elaspic.elaspic\_database\_tables.UniprotDomainPairTemplate attribute), 42  
 provean\_supset\_file (elaspic.elaspic\_sequence.Sequence attribute), 45  
 score\_if\_1 (elaspic.elaspic\_database\_tables.UniprotDomainPairTemplate attribute), 42  
 provean\_supset\_filename  
 (elaspic.elaspic\_database\_tables.Provean attribute), 33  
 score\_if\_2 (elaspic.elaspic\_database\_tables.UniprotDomainPairTemplate attribute), 42  
 provean\_supset\_length (elaspic.elaspic\_database\_tables.Provean attribute), 33  
 score\_if\_total (elaspic.elaspic\_database\_tables.UniprotDomainPairTemplate attribute), 42  
 score\_overall (elaspic.elaspic\_database\_tables.UniprotDomainPairTemplate attribute), 42

- score\_pairwise() (elaspic.elaspic\_sequence.Sequence method), 45
- score\_total (elaspic.elaspic\_database\_tables.UniprotDomainTemplate attribute), 42
- secondary\_structure\_mut (elaspic.elaspic\_database\_tables.UniprotDomainMutation attribute), 37
- secondary\_structure\_mut (elaspic.elaspic\_database\_tables.UniprotDomainPairMutation attribute), 40
- secondary\_structure\_wt (elaspic.elaspic\_database\_tables.UniprotDomainMutation attribute), 37
- secondary\_structure\_wt (elaspic.elaspic\_database\_tables.UniprotDomainPairMutation attribute), 40
- SelectChains (class in elaspic.structure\_tools), 50
- Sequence (class in elaspic.elaspic\_sequence), 45
- sequence\_version (elaspic.elaspic\_database\_tables.UniprotDomainTemplate attribute), 44
- session\_scope() (elaspic.elaspic\_database.MyDatabase method), 30
- Singleton (class in elaspic.conf), 28
- slugify() (in module elaspic.helper), 48
- solvent\_accessibility\_mut (elaspic.elaspic\_database\_tables.UniprotDomainMutation attribute), 37
- solvent\_accessibility\_mut (elaspic.elaspic\_database\_tables.UniprotDomainPairMutation attribute), 40
- solvent\_accessibility\_wt (elaspic.elaspic\_database\_tables.UniprotDomainMutation attribute), 37
- solvent\_accessibility\_wt (elaspic.elaspic\_database\_tables.UniprotDomainPairMutation attribute), 40
- sqlite\_db\_dir, 7
- sqlite\_table\_filename (elaspic.elaspic\_database.MyDatabase attribute), 30
- stability\_energy\_mut (elaspic.elaspic\_database\_tables.UniprotDomainMutation attribute), 37
- stability\_energy\_mut (elaspic.elaspic\_database\_tables.UniprotDomainPairMutation attribute), 40
- stability\_energy\_wt (elaspic.elaspic\_database\_tables.UniprotDomainMutation attribute), 37
- stability\_energy\_wt (elaspic.elaspic\_database\_tables.UniprotDomainPairMutation attribute), 40
- StructureParser (class in elaspic.structure\_tools), 50
- subprocess\_check\_output() (in module elaspic.helper), 48
- subprocess\_check\_output\_locally() (in module elaspic.helper), 48
- subprocess\_communicate() (in module elaspic.helper), 48
- suppress\_logger() (in module elaspic.structure\_tools), 53
- switch\_paths() (in module elaspic.helper), 48
- T**
- t\_date\_modified (elaspic.elaspic\_database\_tables.UniprotDomainTemplate attribute), 42
- t\_date\_modified (elaspic.elaspic\_database\_tables.UniprotDomainMutation attribute), 43
- TCoffee (class in elaspic.call\_tcoffee), 27
- TcoffeeBlastError, 47
- TcoffeeError, 47
- TcoffeePDBidError, 47
- temp\_dir string, 6
- template (elaspic.elaspic\_database\_tables.UniprotDomainModel attribute), 34
- template (elaspic.elaspic\_database\_tables.UniprotDomainPairModel attribute), 39
- template\_errors (elaspic.elaspic\_database\_tables.UniprotDomainPairTemplate attribute), 40
- template\_errors (elaspic.elaspic\_database\_tables.UniprotDomainTemplate attribute), 43
- TemplateCoreError, 47
- TemplateInterfaceError, 47
- TMPDIR, 6, 28
- U**
- UNDERLINE (elaspic.helper.color attribute), 48
- underline() (in module elaspic.helper), 49
- uniprot\_domain (elaspic.elaspic\_database\_tables.UniprotDomainTemplate attribute), 43
- uniprot\_domain\_1 (elaspic.elaspic\_database\_tables.UniprotDomainPairMutation attribute), 37
- uniprot\_domain\_2 (elaspic.elaspic\_database\_tables.UniprotDomainPairMutation attribute), 37
- uniprot\_domain\_id (elaspic.elaspic\_database\_tables.UniprotDomainMutation attribute), 33
- uniprot\_domain\_id (elaspic.elaspic\_database\_tables.UniprotDomainPairMutation attribute), 34
- uniprot\_domain\_id (elaspic.elaspic\_database\_tables.UniprotDomainMutation attribute), 37
- uniprot\_domain\_id (elaspic.elaspic\_database\_tables.UniprotDomainTemplate attribute), 43
- uniprot\_domain\_id\_1 (elaspic.elaspic\_database\_tables.UniprotDomainPairMutation attribute), 39
- uniprot\_domain\_id\_2 (elaspic.elaspic\_database\_tables.UniprotDomainPairMutation attribute), 37
- uniprot\_domain\_pair\_id (elaspic.elaspic\_database\_tables.UniprotDomainPairMutation attribute), 37
- uniprot\_domain\_pair\_id (elaspic.elaspic\_database\_tables.UniprotDomainPairMutation attribute), 39
- uniprot\_domain\_pair\_id (elaspic.elaspic\_database\_tables.UniprotDomainPairMutation attribute), 40
- uniprot\_domain\_pair\_id (elaspic.elaspic\_database\_tables.UniprotDomainPairMutation attribute), 42
- uniprot\_id (elaspic.elaspic\_database\_tables.Provean attribute), 33
- uniprot\_id (elaspic.elaspic\_database\_tables.UniprotDomainMutation attribute), 33
- uniprot\_id (elaspic.elaspic\_database\_tables.UniprotDomainMutation attribute), 37

uniprot\_id (elaspic.elaspic\_database\_tables.UniprotDomainPairMutation attribute), 40

uniprot\_id (elaspic.elaspic\_database\_tables.UniprotSequence attribute), 44

uniprot\_id\_1 (elaspic.elaspic\_database\_tables.UniprotDomainPair attribute), 37

uniprot\_id\_2 (elaspic.elaspic\_database\_tables.UniprotDomainPair attribute), 37

uniprot\_name (elaspic.elaspic\_database\_tables.UniprotSequence attribute), 44

uniprot\_sequence (elaspic.elaspic\_database\_tables.Provean attribute), 33

uniprot\_sequence (elaspic.elaspic\_database\_tables.UniprotDomain attribute), 33

uniprot\_sequence (elaspic.elaspic\_database\_tables.UniprotSequence attribute), 44

UniprotDomain (class in elaspic.elaspic\_database\_tables), 33

UniprotDomainModel (class in elaspic.elaspic\_database\_tables), 34

UniprotDomainMutation (class in elaspic.elaspic\_database\_tables), 34

UniprotDomainPair (class in elaspic.elaspic\_database\_tables), 37

UniprotDomainPairModel (class in elaspic.elaspic\_database\_tables), 37

UniprotDomainPairMutation (class in elaspic.elaspic\_database\_tables), 39

UniprotDomainPairTemplate (class in elaspic.elaspic\_database\_tables), 40

UniprotDomainTemplate (class in elaspic.elaspic\_database\_tables), 42

UniprotSequence (class in elaspic.elaspic\_database\_tables), 43

update() (elaspic.conf.Configs method), 28

## V

values() (elaspic.conf.Configs method), 28

## W

WARNING (elaspic.helper.color attribute), 48

web\_server, 6

working\_dir (elaspic.structure\_analysis.AnalyzeStructure attribute), 50

WritableObject (class in elaspic.helper), 47

write() (elaspic.helper.WritableObject method), 47

write\_row\_to\_file() (in elaspic.machine\_learning module), 49

WrongConfigKeyError, 47

## Y

YELLOW (elaspic.helper.color attribute), 48