ELASPIC Documentation

Release 1.0.22

kimlab

June 29, 2016

Contents

ELASPIC is a metapredictor which combines sequential features (most important being PROVEAN) with structural features (most important being FoldX). It uses the Stochastic Gradient Boosting algorithm for machine learning.

- ELASPIC is designed to work on the genome-wide scale by using homology models.
- It predicts mutation $\Delta\Delta G$ for protein folding and protein interactions.
- It is open source and can be installed and ran locally.

CHAPTER 1

Introduction

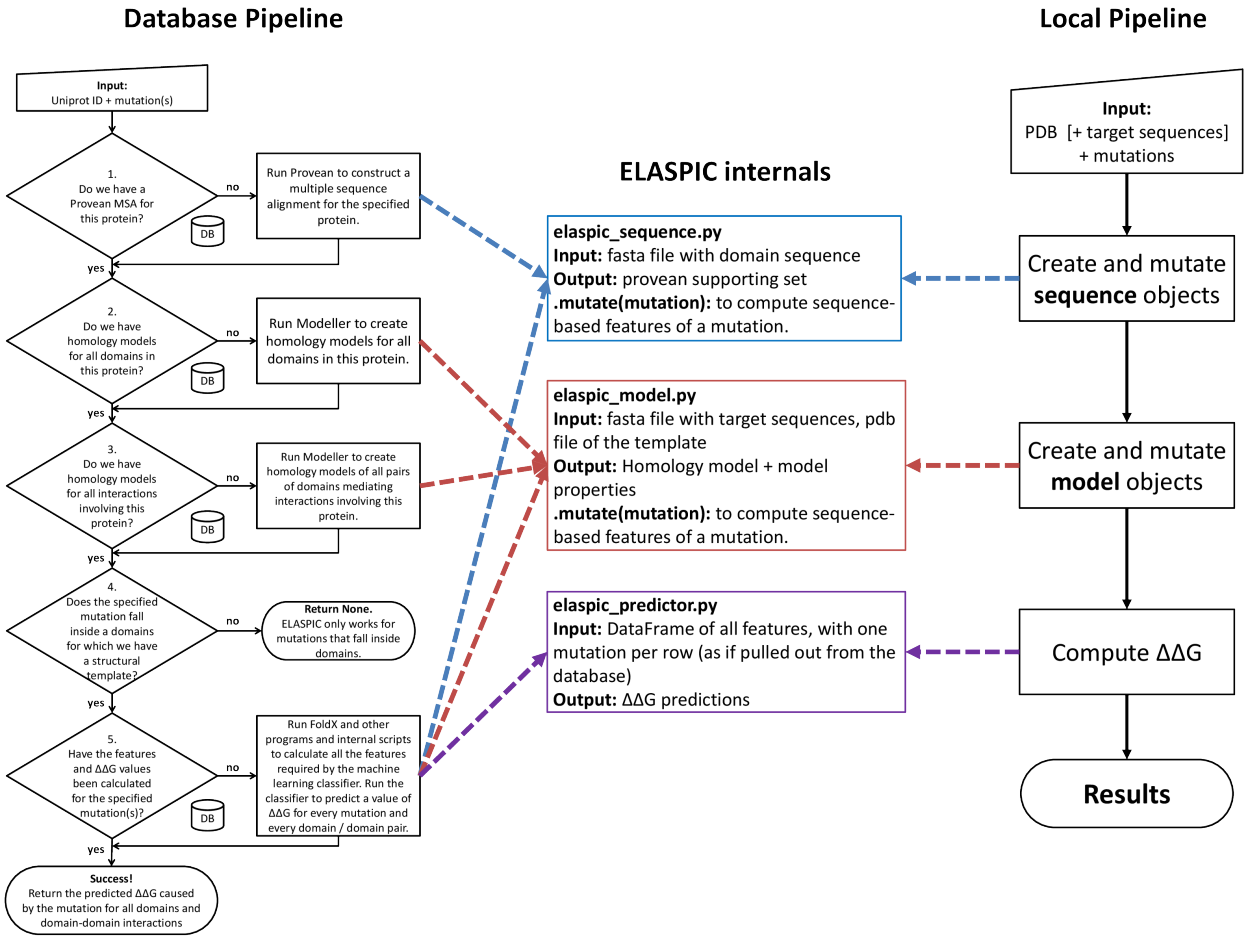
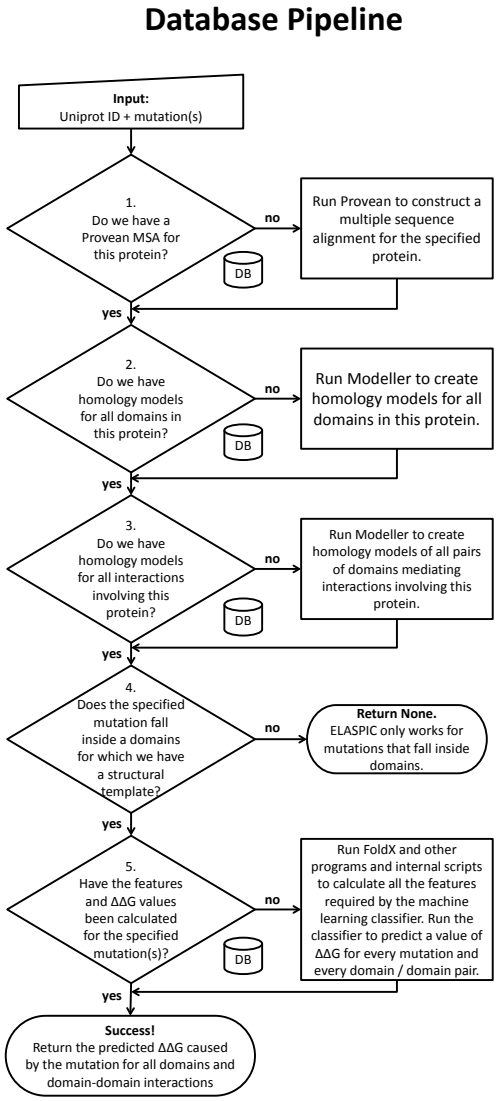


Fig. 1.1: Flowchart describing the ELASPIC pipeline .

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

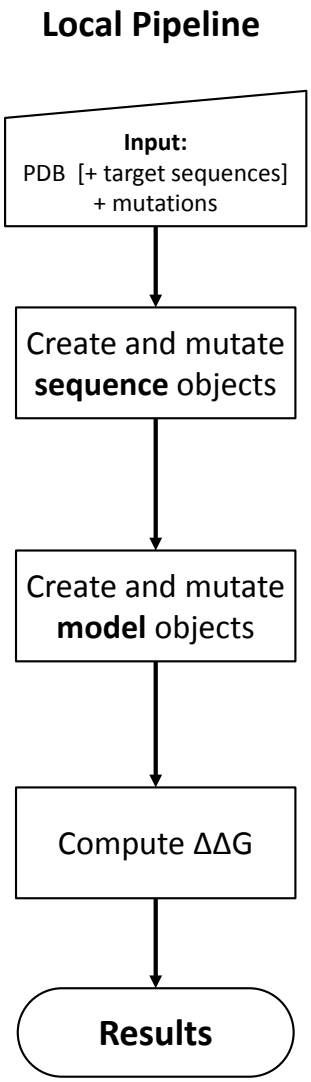
1.1 Database pipeline



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 pipleine 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 saves 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 pipleine is presented in the `figure` to the right.

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

CHAPTER 2

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 preferrably 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_MODELLER` in your ~/.bashrc file:

```
# ~/.bashrc
export KEY_MODELLER=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/mysql5/mysqld.sock` for *MySQL* and `/var/lib/postgresql` for *PostgreSQL*.

schema_version Database schema to use for storing and retreiving 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
```

CHAPTER 3

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 database maintenance tasks
    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.

CHAPTER 4

Benchmarks

Rosetta benchmarks

- https://guybrush.ucsf.edu/benchmarks/captures/DDG

4.1 Existing approaches

4.1.1 Sequence only

intogen

- https://www.intogen.org/search
- oncoDRIVE
- oncoROLE
- http://bg.upf.edu/group/index.php

mCSM: predicting the effects of mutations in proteins using graph-based signatures.

- http://www.ncbi.nlm.nih.gov/pubmed/24281696
- "To understand the roles of mutations in disease, we have evaluated their impacts not only on protein stability but also on protein-protein and protein-nucleic acid interactions".
- cite{pires_mcsm_2014}

4.1.2 Sequence and structure

Predicting Binding Free Energy Change Caused by Point Mutations with Knowledge-Modified MM/PBSA Method

- http://journals.plos.org/ploscompbiol/article?id=10.1371%2Fjournal.pcbi.1004276
- "The core of the SAAMBE method is a modified molecular mechanics Poisson-Boltzmann Surface Area (MM/PBSA) method with residue specific dielectric constant".
- cite{petukh_predicting_2015}

MAESTRO cite{laimer_maestro_2015}

- https://biwww.che.sbg.ac.at/?page_id=477
- MAESTRO implements a multi-agent machine learning system.

- Structure based tools AUTO-MUTE [7], CUPSAT [8], Dmutant [9], FoldX [10], Eris [11], PoPMuSiC [12], SDM [13] or mCSM [14] usually perform better than the sequence based counterparts. Recently, SDM and mCSM have been integrated into a new method called DUET [15].

INPS: predicting the impact of non-synonymous variations on protein stability from sequence

- http://bioinformatics.oxfordjournals.org/content/31/17/2816.long
- Here, we describe INPS, a novel approach for annotating the effect of non-synonymous mutations on the protein stability from its sequence.
- cite{fariselli_inps_2015}

FireProt: Energy- and Evolution-Based Computational Design of Thermostable Multiple-Point Mutants

- http://journals.plos.org/ploscompbiol/article?id=10.1371%2Fjournal.pcbi.1004556
- Predict the structural effect of multiple mutations.
- "Stability effects of all possible single-point mutations were estimated using the <BuildModel> module of FoldX".
-
- We demonstrate that thermostability of the model enzymes haloalkane dehalogenase DhaA and γ-hexachlorocyclohexane dehydrochlorinase LinA can be substantially increased.
- cite{bednar_fireprot_2015}

CHAPTER 5

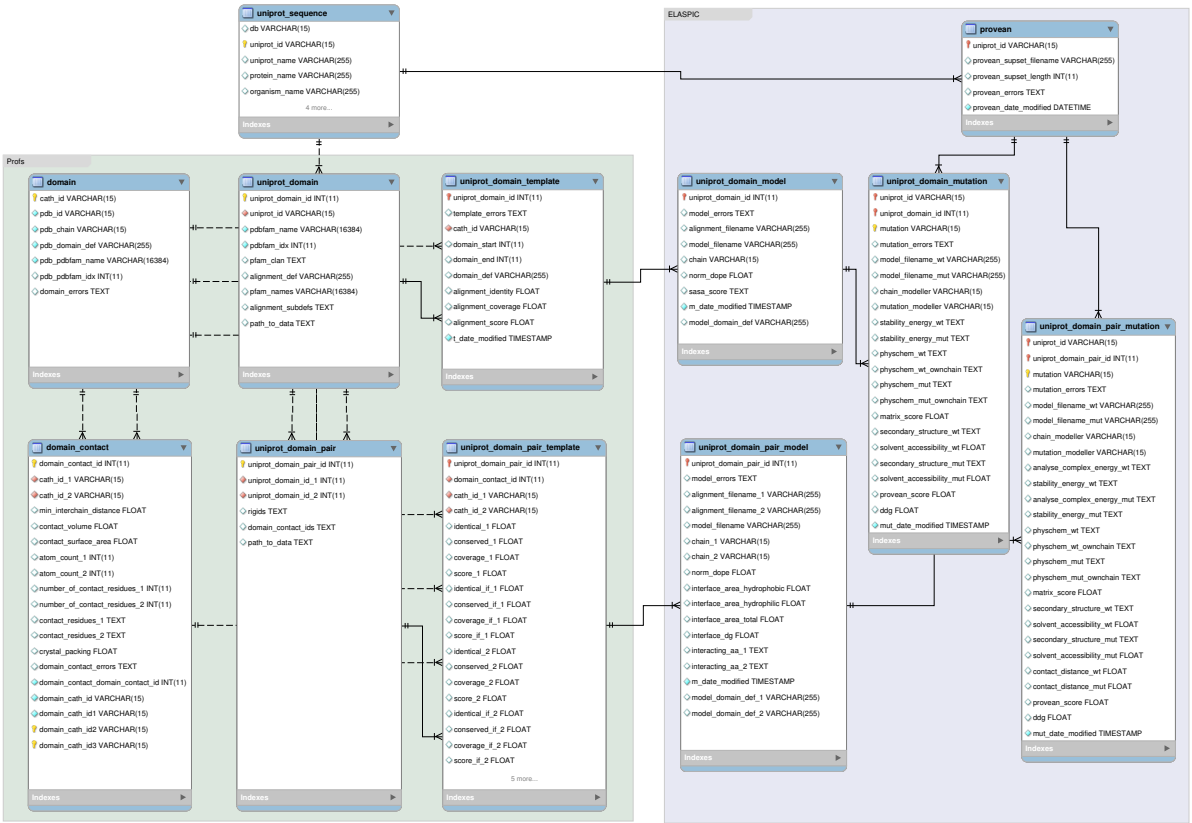
Statistics

Work in progres...

CHAPTER 6

Database

6.1 Database schema



6.2 Database tables

6.2.1 domain

6.2.2 domain_contact

6.2.3 uniprot_sequence

6.2.4 provean

6.2.5 uniprot_domain

6.2.6 uniprot_domain_template

CHAPTER 7

Modules

7.1 elaspic package

7.1.1 Submodules

7.1.2 elaspic.call_foldx module

7.1.3 elaspic.call_modeller module

7.1.4 elaspic.call_tcoffee module

7.1.5 elaspic.conf module

7.1.6 elaspic.database_pipeline module

7.1.7 elaspic.elaspic_database module

7.1.8 elaspic.elaspic_database_tables module

7.1.9 elaspic.elaspic_model module

7.1.10 elaspic.elaspic_predictor module

7.1.11 elaspic.elaspic_sequence module

7.1.12 elaspic.errors module

7.1.13 elaspic.helper module

7.1.14 elaspic.local_pipeline module

7.1.15 elaspic.machine_learning module

7.1.16 elaspic.pipeline module

7.1.17 elaspic.structure_analysis module

7.1.18 elaspic.structure_tools module

7.1.19 Module contents

7.2 Indices and tables