Pycoevol *User Guide*

Document version 1.0 - April 2012

Table of contents

1 Introduction

Pycoevol is an integrated system for studying inter-protein coevolution and interaction. It automates the identification of contact points between protein partners, extending the general coevolution workflow consisting of: homologous sequence search; multiple sequence alignment computation; and coevolution analysis; with an improved selection of organisms and contact prediction.

It generates friendly output results: matrix of scores; histograms; heat-maps; PyMOL scripts and interaction maps. Additional information for common web-services can be retrieved from SIFTS.

1.1 Availability

Pycoevol is platform independent. Packages and the source code can be downloaded from http://code.google.com/p/pycoevol/ and https://github.com/fmadeira/pycoevol, respectively.

1.2 Disclaimer

This software is provided "as is", with no explicit or implied warranties. Use this software at your own risk.

1.3 Copyright

This software is public domain, and everyone has the right to copy, distribute, reuse, modify, improve and debug it. If you want to cite this piece of software/workflow use the following:

Fábio Madeira and Ludwig Krippahl. 2012. PYCOEVOL: A Python workflow to study protein-protein coevolution. Proceedings of the International conference on Bioinformatics Models, Methods and Algorithms - BIOINFORMATICS 2012, pp.143-9.

2 Installation

Pycoevol is delivered as a command-line program called *Pycoevol.py* that can be friendly interfaced using pycoevolgui.exe (on windows) or pycoevolgui (on linux).

2.1 Dependencies

Pycoevol is a package of open source tools implemented in Python 2.7 (v2.7.2) (http://python.org/). It uses the Biopython module (v1.5.8) (http://biopython.org/wiki/Biopython) for wrapping standalone programs and for general manipulation of biological data. Numpy (v1.6.1) (http://numpy.scipy.org/) and Matplotlib (v1.1.0) (http://matplotlib.sourceforge.net/) are also used to extend the capabilities of the Python language.

Although most modern operating systems have python pre-installed, you should check its version and/or install it if not provided. To install all the dependencies you must follow the software installation instructions on the links provided above. In your terminal/shell/command prompt, type *python* (in some cases *python27)*. Inside the python shell type *import Bio* (do the same for *import numpy* and for *import matplotlib*). This is a quick way of testing if you have all the python dependencies properly installed.

As mandatory tools Pycoevol requires ClustalW (http://www.clustal.org/), for the computation of pairwise alignments, optionally for MSA computation. NCBI PSI-BLAST (ftp://ftp.ncbi.nlm.nih.gov/blast/executables/blast+/LATEST/) for use with a local sequence database, as well as MAFFT (http://mafft.cbrc.jp/alignment/software/) and MUSCLE (http://www.drive5.com/muscle/) for MSA computation, are only optionally required.

Again follow the installation instructions on the links provided above.

2.2 Quick installation

The easiest way to install Pycoevol is to start from packages. Packages include pre-compiled tools (pycoevolgui, pdbsurface and clustalw) and are available at http://code.google.com/p/pycoevol/. To install Pycoevol, you need to unpack the appropriate package (Pycoevol_i386-win32 or Pycoevol_i386-linux) to a suitable location in your system (generally where you will place your experiment files).

2.3 Installing from source

Installing Pycoevol from source is not much different but can give you a lot more work. The source code can be downloaded from https://github.com/fmadeira/pycoevol.

First make sure you add clustalw (or clustalw2), muscle, mafft and psiblast to your system/environment path (clustalw is mandatory). Alternatively, place all the applications inside the pycoevol folder in agreement with the following directory tree:

"./Pycoevol/src/tools" – place here *pdbsurface*
"./Pycoevol/src/tools/blast+" – place here *psiblast*
"./Pycoevol/src/tools/blast+/db" – place here your local database (see below)
"./Pycoevol/src/tools/clustalw" – place here *clustalw*
"./Pycoevol/src/tools/mafft" – place here *mafft*
"./Pycoevol/src/tools/muscle" – place here *muscle*
"./Pycoevol/SIFTS/" – place here all the **.lst files* from
http://www.ebi.ac.uk/pdbe/docs/sifts/quick.html

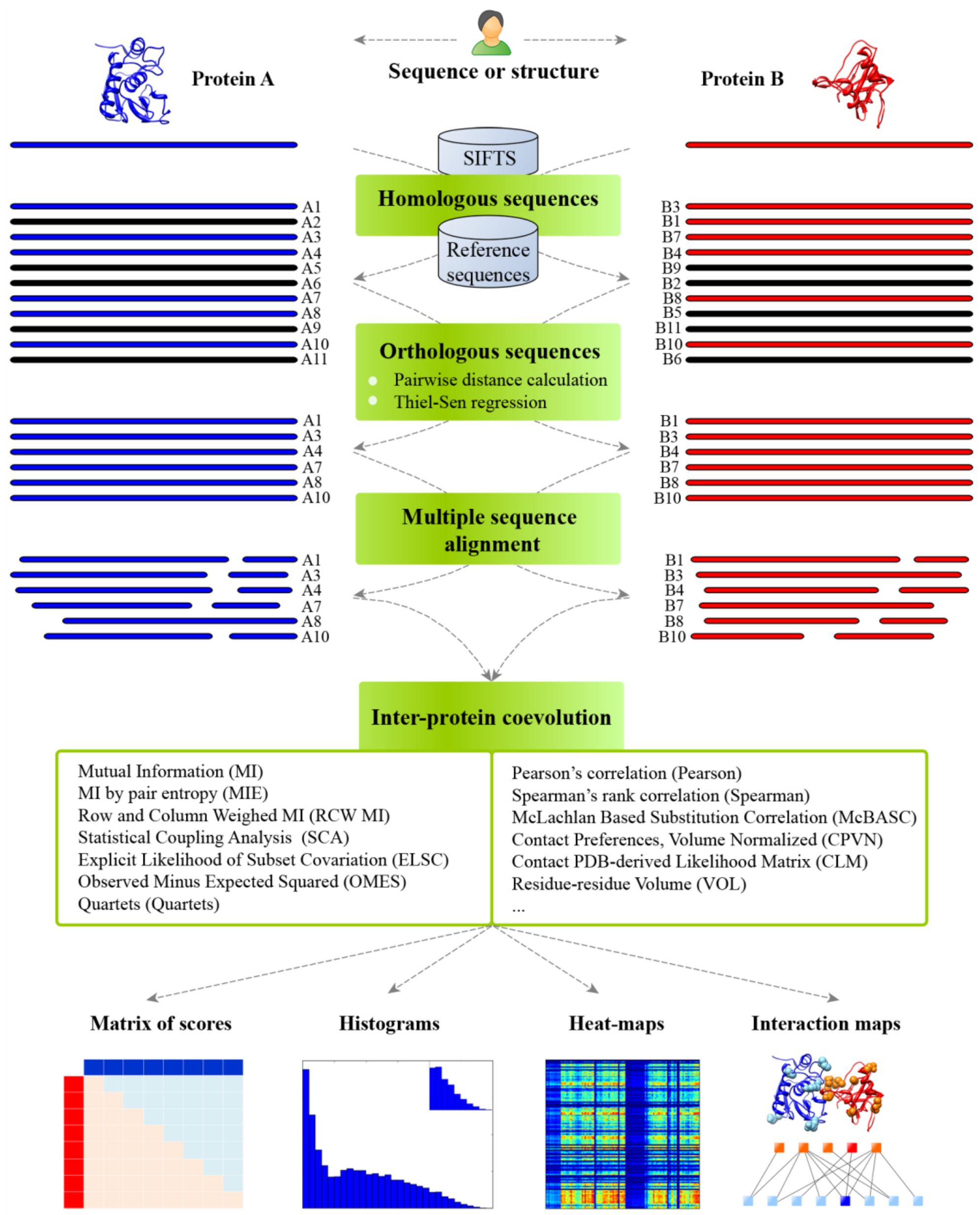
To compile *pdbsurface* you must download the Open Chemera Library (https://github.com/lkrippahl/Open-Chemera), and follow these instructions: 1) Install Lazarus and FPC, which are available here (http://www.lazarus.freepascal.org/); 2) Install the TOpenGLControl LCL control in the package lazarus/components/opengl/lazopenglcontext.lpk (in the Lazarus IDE, select "Package->Install/uninstall packages"); 3) Open the project file pdbsurface/pdbsurface.lpi; 4) Press F9; 5) Place the executable in the appropriate folder inside Pycoevol directory.

To compile *pycoevolgui* you must follow the same steps, with exception of step 2). You must open *pycoevolgui.lpi*. It can be found on the GUI folder https://github.com/fmadeira/pycoevol.

A local database (**refseq_protein**) can be installed locally for use with Pycoevol. The database can be retrieved from ftp://ftp.ncbi.nlm.nih.gov/blast/db/ and should be placed (unpacked) in the location shown above.

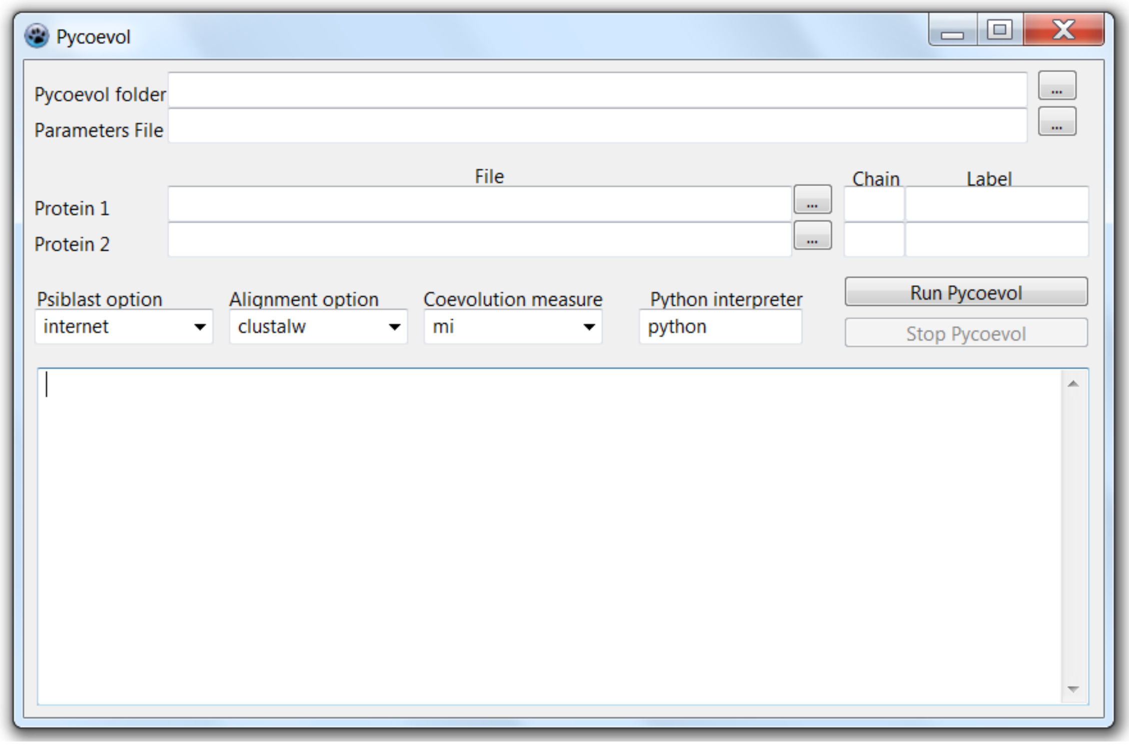
3 Using Pycoevol

Pycoevol employs a sequential workflow that enables the study of inter-protein coevolution. As shown below, several main tasks are covered: search for homologous sequences; selection of orthologous sequences; multiple sequence alignment; and coevolution analysis (scoring). In order to enable the integrated study of protein coevolution, Pycoevol contains several major functional modules for sequence handling, alignment processing and coevolution scoring.


Protein A
Sequence or structure
Protein B
SIFTS
Homologous sequences
Reference sequences
A1 A2 A3 A4 A5 A6 A7 A8 A9 A10 A11
B3 B1 B7 B4 B9 B2 B8 B5 B11 B10 B6
Orthologous sequences
Pairwise distance calculation
Thiel-Sen regression
A1 A3 A4 A7 A8 A10
B1 B3 B4 B7 B8 B10
Multiple sequence alignment
Inter-protein coevolution
Mutual Information (MI)
MI by pair entropy (MIE)
Row and Column Weighed MI (RCW MI)
Statistical Coupling Analysis (SCA)
Explicit Likelihood of Subset Covariation (ELSC)
Observed Minus Expected Squared (OMES)
Quartets (Quartets)
Pearson's correlation (Pearson)
Spearman's rank correlation (Spearman)
McLachlan Based Substitution Correlation (McBASC)
Contact Preferences, Volume Normalized (CPVN)
Contact PDB-derived Likelihood Matrix (CLM)
Residue-residue Volume (VOL)
...
Matrix of scores
Histograms
Heat-maps
Interaction maps

3.1 Quick start

The quickest way to start using Pycoevol is to use the Graphical User Interface (GUI), *pycoevolgui*. This enables you to select the Pycoevol's main folder, the parameters file (optional – if not stated default configurations will be used), the protein files (PDB files or FASTA files), chains (only for PDB files), labels (accession numbers for protein sequences (GI sequence identifier and UniProt ID) and structures (PDB ID)), psiblast options, alignment options, coevolution options and the python interpreter. The standard messages are printed out in the shell provided.



Psiblast can be performed locally or over the internet (or custom if the analysis starts from multiple alignment files). For the multiple alignments, clustalw, muscle and mafft can be used (or custom if the analysis starts from multiple alignment files). The coevolution option enables the selection of several different coevolution measures (see section 5.2).

Processed files and output results are placed in the same directory of the input files. If the analysis starts using labels (accession numbers) the results are located in the Results folder (inside Pycoevol – './Pycoevol/Results/').

3.2 Using the Command Line

Pycoevol.py is the main command line tool that enables the execution of the software, and the same input arguments can be used. The GUI is just a superior layer that intermediates the *Pycoevol.py* with the user. Pycoevol runs like another python script in the Command Line Interface (CLI), and typing "*python Pycoevol.py –h*" prints out the help message that and further instructions on how to properly input the arguments. (>> is the command line/terminal/shell prompt character)

>> *python Pycoevol.py -help*

```
 python Pycoevol.py input1 input2 [options]

-h, --help
                              show this help message and exit
 -b PSIBLAST, --psiblast=PSIBLAST
                              internet, local or custom
 -a ALIGNMENT, --alignment=ALIGNMENT
                              clustalw, muscle, mafft or custom
 -c COEVOLUTION, --coevolution=COEVOLUTION
                                mi, mie, rcwmi, cpvn, clm, vol, omes, pearson,
                                spearman, mcbasc, quartets, sca or elsc
 -i IDS, --id=IDS
 -x CHAINS, --chain=CHAINS
 -p PARAMETERFILE, --parameters=PARAMETERFILE
```

3.3 Examples

This session provides examples that can be used using either the Pycoevol GUI or the CLI.

>> python "C:\Coevolution\Pycoevol.py" "C:\1RGH.pdb" "C:\1A19.pdb" **-b**internet **-a**clustalw **-c**mi **-x**A **-x**B **-i**1RGH **-i**1A19 **-p**"C:\Pycoevol\Params.config"

In this case labels are optional since PDB files are provided.

>> *python "C:\Pycoevol\Pycoevol.py"* ***-b****internet* ***-a****clustalw* ***-c****mi* ***-x****A* ***-x****A* ***-i****1RGH* ***-i****1A19*

Here, labels are used to retrieve the PDB files, since chains are introduced. If chains were not provided these labels would be considered sequence IDs instead of PDB IDs. Note that in this case the *parameterfile* argument was not specified and default configurations were used.

>> python *"C:\Pycoevol\Pycoevol.py"* ***-b****internet* ***-a****mafft* ***-c****pearson* ***-i***P05798 ***-i***P11540

Labels are considered sequence IDs in this case.

>> *python "C:\Pycoevol\Pycoevol.py" "C:\1AY7.pdb" "C:\1AY7.pdb"* ***-b****local* ***-a****clustalw* ***-c****cpvn* ***-x****A* ***-x****B* ***-p****"C:\Pycoevol\Params.config"*

The input files are the same, but they refer to different chains in the PDB file.

>> *python "C:\Pycoevol\Pycoevol.py" "C:\1RGH.fasta" "C:\1A19.fasta"* ***-b****custom* ***-a****custom* ***-c****mi* ***-p****"C:\Pycoevol\Params.config"*

The analysis starts from alignment files in FASTA format using custom as psiblast and alignment options. In this case, the first sequence from each alignment is considered to be the query sequence. Besides, the user has to make sure that both alignments have the same number of sequences and they are matched by organism.

4 Parameters

Additional parameters can be prompt prior to coevolution analysis using Pycoevol. These parameters supplement arguments with additional options. These cover Psiblast, Alignment and Results options, as listed below:

Configuration	Description	Data type
[Global]		
surfacethreshold = 7	percentage (%)	INT
pairwisedistance = clustalw	clustalw, pdistance, kimura, jukescantor or alignscore	SRTG
matrix = BLOSUM62	BLOSUM62 or PAM250	STRG
theilsencutofft = 0.5	[0.25:1.0]	FLOAT
alignmentscore = False	sumofpairs or False	STRG
alphabetreduction = False	charge, charge_his, polarity, hydropathy or False	STRG
[Psiblast]		
evalue = 10	[10**-10:10]	FLOAT
identity = 0	[0:100]	INT
coverage = 0	[0:100]	INT
threading = False	Number of cores or False	INT
[Clustalw]		
gapopening = 10	[0:100]	INT
gapextension = 0.2	[0:10]	INT
matrix = GONNET	GONNET, BLOSUM or PAM	STRG
[Muscle]		
maxiteration = 16	[2:16]	INT
[Mafft]		
configuration = linsi	fftnsi or linsi	SRTG
threading = False	Number of cores or False	INT
[Results]		
best = 20	[1:max[	INT
histogram = True	True or False	BOOLEAN
heatmap = True	True or False	BOOLEAN
structure = False	pymol or False	STRG
sifts = False	True or False	BOOLEAN

To edit these parameters you can edit Params.config (inside Pycoevol folder) and make sure you follow the exact data type suggested above. However it is recommended to backup Params.config or copy it to a different location and work from this new version of the file.

Of particular importance are the Results's options that enable the selection of the number of best scoring interaction points, as well as, the generation of histograms, heatmaps, PyMOL scripts and additional information for common web-services from SIFTS.

5 Additional considerations

5.1 Workflow of Pycoevol

For each queried sequence or structure, a collection of homologous sequences is searched against the NCBI's reference sequence database, locally or over the internet, using PSI-BLAST.

The following steps of the workflow concern the selection of orthologous sequences. Both sets of homologous sequences are matched by organism (taxa). In order to proper select the orthologous sequences, a pairwise alignment between the query and each sequence is calculated, using one of the following methods: ClustalW pairwise distance score; p-distance; Jukes-Cantor distance; Kimura distance; pairwise alignment score using PAM or BLOSUM matrices. The most similar sequence for each taxon is then included in the analysis. Having two sets of selected sequences, and using the pairwise distances, the plot of each matrix of distances against the other is used to measure the correlation between the sequences. This is done using the Thein-Sen estimator. The points that are far from the estimation are considered non-correlating sequences (taxa), and are rejected from the analysis. The user can specify the cut-off for the Theil-Sen estimator. Adjusting it enables major or minor removal of sequences, from highly correlated sequences only to all the sequences, 0.25 to 1.0 respectively.

The orthologous sequences are then aligned using either ClustalW, MUSCLE or MAFFT. The user can specify common MSA parameters such as the gap opening penalty or the mutation matrix in use. Since a coevolution score is calculated for each pair of positions between the MSAs (inter-protein), the alignments are trimmed to cover the specific residues in analysis, matching the positions of the queried sequences (or structures). In order to enable a more flexible use of Pycoevol, the user can start the analysis with custom-made MSAs (in FASTA format). In this case, the first sequence from each alignment is considered to be the query sequence.

The analysis of inter-protein coevolution is done using one of the methods currently implemented in Pycoevol: Pearson’s correlation (Pearson); Spearman’s rank correlation (Spearman); McLachlan Based Substitution Correlation (McBASC); Quartets; Observed Minus Expected Squared (OMES); Statistical Coupling Analysis (SCA); Explicit Likelihood of Subset Covariation (ELSC); Mutual Information (MI); MI divided by pair entropy (MIE); and Row and Column Weighed MI (RCW MI). Pycoevol implements also methods based on the residue-residue contact preferences, which accounts for specific pairing preferences at the interface (matrix-based): contact preferences, volume normalized (CPVN); contact PDB-derived likelihood matrix (CLM); and a residue-residue volume derived matrix (VOL).

In order to improve the analysis of the results, Pycoevol can use structural information to estimate the solvent accessible surface area (SASA), using the classic 'rolling ball' algorithm (implemented in *pdbsurface*). This information is then used as a weighting feature on the coevolution analysis. Automatically generated output results cover the following: 1) matrices and lists of the coevolution scores; 2) histograms; 3) heat-maps; 4) interaction maps and PyMOL scripts, which plot the predicted pairs of co-evolving sites onto the structures (if the analysis starts with 3D structures); thus enabling an easier interpretation of the results. Additional information for common web-services is also fetched through the SIFTS protocol.

5.2 Coevolution measures

- Mutual Information (mi) [Gloor et al, 2005]
- MI by pair Entropy (mie) [Martin et al, 2005]
- Row and Column Weighed MI (rcwmi) [Gouveia-Oliveira et al, 2007]
- Contact Preferences, Volume Normalized (cpvn) [Glaser et al, 2001]
- Contact PDB-derived Likelihood Matrix (clm) [Singer et al, 2002]
- Residue-residue Volume Normalized (vol) [based on Esque et al, 2010]
- Observed Minus Expected Squared (omes) [Kass and Horovitz, 2002]
- Pearson's correlation (pearson) [Göbel et al, 1994]
- Spearman's rank correlation (spearman) [Pazos et al, 1997]
- McLachlan Based Substitution Correlation (mcbasc) [Fodor and Aldrich, 2004]
- Quartets (quartets) [Galitsky, 2002]
- Statistical Coupling Analysis (sca) [Lockless and Ranganathan, 1999]
- Explicit Likelihood of Subset Covariation (elsc) [Dekker et al, 2004]

5.3 Troubleshooting and Debugging

Here is some general advice on what to do if Pycoevol fails and you don't understand what happened. The code provides informative error messages in limited situations, and there will be situations we haven't anticipated (not to mention bugs). If you find some bugs, unexpected failure or strange results please contact me (see below). If you follow the guidelines provided in this document thoroughly, it is likely that proper coevolution analysis will be performed.

Please have in mind that this is open source software and you may need to make minor changes or debug some parts of the code. User requests and contributions may be implemented and included in future versions of the software. Besides, we are open to suggestions and collaborations.

5.4 Contacts

Fábio Madeira (fmadeira@campus.fct.unl.pt) and Ludwig Krippahl (ludi@di.fct.unl.pt)

CENTRIA-DI, Universidade Nova de Lisboa, Caparica, Portugal