# LabBlouinTools

# API Documentation

# November 28, 2014

# Contents

C	onter	nts	1
1	<b>Pac</b> 1.1	ekage labblouin  Modules	2
2	Mo	dule labblouin.FASTAnet	3
	2.1	Variables	3
	2.2	Class FASTAsequence	3
		2.2.1 Methods	3
	2.3	Class FASTAstructure	4
		2.3.1 Methods	4
3	Mo	dule labblouin.IO	6
	3.1	Functions	6
	3.2	Variables	6
4	Mo	dule labblouin.Ordination	8
-	4.1	Variables	8
	4.2	Class ORD	8
		4.2.1 Methods	8
5	Mo	dule labblouin.PDBS2VMDstate	10
	5.1		10
	5.2	Variables	10
6	Mo	dule labblouin.PDBnet	11
_	6.1		11
	6.2	Class PDBatom	11
		6.2.1 Methods	11
		6.2.2 Properties	12
	6.3	Class PDBterminator	12
		6.3.1 Methods	13
		6.3.2 Properties	13
	6.4	Class PDBresidue	14
		6.4.1 Methods	14
	6.5	Class PDBchain	14
		6.5.1 Methods	15
		6.5.2 Properties	17

CONTENTS

	6.6	Class PDBmodel	17 17 19
	6.7	Class PDBstructure	19 19
	6.8	6.7.2 Properties	24 24 25
		6.8.2 Properties	26
7	<b>Moo</b> 7.1 7.2	Iule labblouin.RegExpress       Functions	27 27 27
3	Mod	lule labblouin.SeqMask	28
	8.1 8.2	Functions	28 28
9	Mod	lule labblouin.expresso	30
	9.1 9.2	Variables	30 30 30
10	Mod	lule labblouin.homology	31
	10.2	Functions	31 32
		Class manualModeller	33 33 33
		10.4.1 Methods	33
11		lule labblouin.homstrad Functions	<b>34</b> 34
		Variables	34 34
	11.4	11.3.1 Methods	34 35 35
12	Μοι	lule labblouin.loadingBar	37
	12.1	Variables	37 37
		12.2.1 Methods	37
13	13.1	lule labblouin.logfile Functions	<b>38</b> 38
		Variables	38 38 38
	13.4	Class logfile	38 38
14	Mod	lule labblouin.mdsa	40

CONTENTS

	14.2 (	Variables	 40
		4.2.1 Methods	
		4.3.1 Methods	
		lle labblouin.memEfficient	42
		Variables	
		5.2.1 Methods	
		5.2.2 Properties	
		5.2.3 Class Variables	
		le labblouin.passToqsub	46
		functions	
	16.2 V	Variables	 46
		lle labblouin.pdbCompare	47
		functions	
		Tariables	
		le labblouin.pfam	48
		functions	
		Variables	
		Slass pfamFile	
19	Modi		
		alle labblouin.plotGM	50
	19.1 V	le labblouin.plotGM	50
20	19.1 V <b>Mod</b> u	ale labblouin.plotGM  Variables	 <b>50</b> 50 <b>51</b>
20	19.1 V <b>Modu</b> 20.1 F	lle labblouin.plotGM Variables	 50 50 51 51
20	19.1 V Modu 20.1 F 20.2 V Modu	lle labblouin.plotGM  Tariables  Unctions Tariables  Tariables  Tariables  Tariables	 50 50 51 51 51
20	19.1 V Modu 20.1 F 20.2 V Modu 21.1 V	ale labblouin.plotGM Variables  Ule labblouin.psub Variables  Variables  Ule labblouin.sabmark Variables	 50 51 51 51 52 52
20	19.1 V Modu 20.1 H 20.2 V Modu 21.1 V 21.2 (	Ale labblouin.plotGM Variables  Ale labblouin.psub Variables  Variables  Ariables  Variables  Variables  Variables  Variables	 50 51 51 52 52 52 53
20	19.1 V Modu 20.1 F 20.2 V Modu 21.1 V 21.2 Q	de labblouin.plotGM Variables  de labblouin.psub Variables	 50 51 51 52 52 52 52 53
20	19.1 V Modu 20.1 F 20.2 V Modu 21.1 V 21.2 Q 21.3 Q	de labblouin.plotGM Variables  de labblouin.psub Variables  Variables  de labblouin.sabmark Variables  Class sabmarkFile  1.2.1 Methods  Class sabmarkSummary	 50 51 51 52 52 52 53
20 21	19.1 V Modu 20.1 F 20.2 V Modu 21.1 V 21.2 ( 21.3 (	lle labblouin.plotGM Variables  Cle labblouin.psub Variables  Variables  Class sabmarkFile  1.2.1 Methods  Class sabmarkSummary  1.3.1 Methods	 50 51 51 52 52 52 52 52 52 52
20 21	19.1 V Modu 20.1 F 20.2 V Modu 21.1 V 21.2 Q 21.3 Q 21.4 Q	de labblouin.plotGM Variables  de labblouin.psub Unctions Variables  de labblouin.sabmark Variables  Class sabmarkFile 1.2.1 Methods Class sabmarkSummary 1.3.1 Methods Class sabmarkFolder	 50 51 51 52 52 52 52 52 52 52 52
20 21	19.1 V Modu 20.1 F 20.2 V Modu 21.1 V 21.2 Q 21.3 Q 21.4 Q	de labblouin.plotGM Fariables  de labblouin.psub Functions Fariables  de labblouin.sabmark Fariables  Class sabmarkFile 1.2.1 Methods Class sabmarkSummary 1.3.1 Methods Class sabmarkFolder 1.4.1 Methods	50 51 51 52 52 52 52 52 52 52 52 52
20 21	19.1 V Modu 20.1 F 20.2 V Modu 21.1 V 21.2 (22) 21.4 (22) 21.5 (22)	de labblouin.plotGM Variables  de labblouin.psub Variables  de labblouin.sabmark Variables  Class sabmarkFile 1.2.1 Methods Class sabmarkSummary 1.3.1 Methods Class sabmarkFolder 1.4.1 Methods Class sabmarkDatabase	50 51 51 52 52 52 52 52 52 52 52 52 52 52 52 52
20 21	19.1 V Modu 20.1 H 20.2 V Modu 21.1 V 21.2 Q 21.3 Q 21.4 Q 21.5 Q 2	de labblouin.plotGM Fariables  de labblouin.psub Functions Fariables  de labblouin.sabmark Fariables  Class sabmarkFile 1.2.1 Methods Class sabmarkSummary 1.3.1 Methods Class sabmarkFolder 1.4.1 Methods Class sabmarkDatabase 1.5.1 Methods	50 51 51 52 52 52 52 52 52 52 52 52 52 52 52 52
20 21 22	19.1 Modu 20.1 H 20.2 M Modu 21.1 M 21.2 Q 21.3 Q 21.4 Q 21.5 Q Modu Modu	de labblouin.plotGM Fariables  de labblouin.psub Functions Fariables  de labblouin.sabmark Fariables  de labblouin.sabmark Fariables  de lass sabmarkFile  1.2.1 Methods  de lass sabmarkSummary  1.3.1 Methods  de lass sabmarkFolder  1.4.1 Methods  de lass sabmarkDatabase  1.5.1 Methods  de labblouin.scop	50 51 51 52 52 52 52 52 52 52 52 54 54
20 21 22	19.1 Modu 20.1 F 20.2 Modu 21.1 V 21.2 Q 21.3 Q 21.4 Q 21.5 Q Modu 22.1 Modu 22.1 V	de labblouin.plotGM  fariables  de labblouin.psub functions fariables  de labblouin.sabmark fariables  flass sabmarkFile  1.2.1 Methods  flass sabmarkSummary  1.3.1 Methods  flass sabmarkFolder  1.4.1 Methods  flass sabmarkDatabase  1.5.1 Methods  flass sabmarkDatabase  1.5.1 Methods  flass sabmarkDatabase  1.5.1 Methods  flass sabmarkDatabase  1.5.1 Methods	50 51 51 52 52 52 52 52 52 52 52 54 54
20 21 22	19.1 Modu 20.1 H 20.2 Modu 21.1 M 21.2 Q 21.3 Q 21.4 Q 21.5 Q 22.1 Modu 22.1 Modu 22.1 M	de labblouin.plotGM Variables  de labblouin.psub Variables  de labblouin.sabmark Variables  Class sabmarkFile 1.2.1 Methods Class sabmarkSummary 1.3.1 Methods Class sabmarkFolder 1.4.1 Methods Class sabmarkDatabase 1.5.1 Methods Class sabmarkDatabase 1.5.1 Methods Class sabmarkDatabase 1.5.1 Methods Class sabmarkDatabase 1.5.2 Methods Class sabmarkDatabase 1.5.3 Methods Class sabmarkDatabase	50 51 51 52 52 52 52 52 52 52 54 54 54
20 21 22	19.1 Modu 20.1 H 20.2 Modu 21.1 M 21.2 Q 21.3 Q 21.4 Q 21.5 Q 22.1 Modu 22.1 Modu 22.2 Q	de labblouin.plotGM  fariables  de labblouin.psub functions fariables  de labblouin.sabmark fariables  flass sabmarkFile  1.2.1 Methods  flass sabmarkSummary  1.3.1 Methods  flass sabmarkFolder  1.4.1 Methods  flass sabmarkDatabase  1.5.1 Methods  flass sabmarkDatabase  1.5.1 Methods  flass sabmarkDatabase  1.5.1 Methods  flass sabmarkDatabase  1.5.1 Methods	50 51 51 52 52 52 52 52 52 52 54 54 54 54
20 21 22	19.1 V Modu 20.1 H 20.2 V Modu 21.1 V 21.2 G 21.3 G 21.4 G 22.1 V 22.2 G 22.2 G	de labblouin.plotGM Variables  de labblouin.psub Variables  de labblouin.sabmark Variables  Class sabmarkFile 1.2.1 Methods Class sabmarkSummary 1.3.1 Methods Class sabmarkFolder 1.4.1 Methods Class sabmarkDatabase 1.5.1 Methods Class sabmarkDatabase 1.5.1 Methods Class scopHierarchy Variables Class scopHierarchy Value Labblouin.scop	50 51 51 52 52 52 52 52 52 52 52 54 54 55 54

CONTENTS

	dule labble																		56
	Variables																		
23.2	$Class\ svm_{\underline{\ }}$																		
	23.2.1 Me	thods		 															56
23.3	Class cross																		
	23.3.1 Me	thods		 															56
	dule labble																		57
24.1	Functions			 											 				57
24.2	Variables			 															57

# 1 Package labblouin

Bioinformatic python utilities modules, and libraries for use with Lab Blouin applications.

#### 1.1 Modules

- **FASTAnet**: A simple parser and writer for FASTA sequence files that uses a rich object hierarchy. (Section 2, p. 3)
- **IO**: Different shortcut functions for dealing with file systems. (Section 3, p. 6)
- Ordination: Ordination is a class designed to compute and plot ordination methods such as PCA and MDS.

(Section 4, p. 8)

- PDBS2VMDstate: PDB(s) to VMD state Giving a list of structures (can be a single one), return a VMD state, with the same color scale (from the minimum of all to the maximum of all), white background, and other visual hacks (Section 5, p. 10)
- **PDBnet**: PDBnet is a collection of Python objects intended to model and contain PDB protein data. (Section 6, p. 11)
- RegExpress: Prosite pattern generator. (Section 7, p. 27)
- SeqMask: Get the amino acid sequence from a landmark file, align this profile to more sequences if required using muscle, and mask the membership vector into this alignment. (Section 8, p. 28)
- expresso: An (incomplete) parsing library for output from the T-Coffee Expresso executable. (Section 9, p. 30)
- homology (Section 10, p. 31)
- homstrad: A library to manage interfacing with the Homstrad database raw files. (Section 11, p. 34)
- loadingBar (Section 12, p. 37)
- logfile (Section 13, p. 38)
- mdsa: A library to manage interfacing with the MDSA database raw files (http://dna.cs.byu.edu/mdsas/index.shtml). (Section 14, p. 40)
- memEfficient (Section 15, p. 42)
- passToqsub (Section 16, p. 46)
- pdbCompare (Section 17, p. 47)
- pfam (Section 18, p. 48)
- plotGM: this will plot a gm of a 2D shape (Section 19, p. 50)
- **psub**: Run as many of a input set of jobs/commands as there are cores at any given time (i.e. (Section 20, p. 51)
- sabmark (Section 21, p. 52)
- scop (Section 22, p. 54)
- svm (Section 23, p. 56)
- timer (Section 24, p. 57)

# 2 Module labblouin.FASTAnet

A simple parser and writer for FASTA sequence files that uses a rich object hierarchy.

# 2.1 Variables

Name	Description
package	Value: None

# 2.2 Class FASTAsequence

#### 2.2.1 Methods

init(self, name, seq)
Initialize this object. Provide a name for the sequence and the sequence itself as parameters.
iter(self)
Iterate through a sequence in a pseudo-line-by-line manner as if it was read in a FASTA file.
hash(self)
$\phantom{aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa$
ne(self, o)
count(self, item)
Return the number of characters in the sequence equal to input string.
$\boxed{ \mathbf{removeGaps}(\mathit{self}) }$
Modify the sequence so gaps are removed and return it.
toUpper(self)
Modify the sequence so it is uppercase and return it.
toLower(self)
Return a lowercase version of the sequence, also changing it in the structure.
len(self)
$\mathbf{str}$ (self)

### 2.3 Class FASTAstructure

#### 2.3.1 Methods

\_init\_\_\_(self, filein=',', uniqueOnly=True, curate=False)

A file to be read is optional. If uniqueOnly is set to false, multiple duplicate sequences are allowed; otherwise, duplicates are ignored and their aliases are recorded in sequence Names. If curate is triggered, will remove special characters from names.

getSequenceNames(self)

getSequences(self)

getSequenceByName(self, n)

getStrictlyUngappedPositions(self, seqInds=None)

Acquire the positions of all strictly ungapped sites. If parameter is set, expects a list of what sequences (by index) you are checking. Defaults to all sequences.

readFile(self, fin)

Read a file in. Return this FASTA object.

read(self, fast)

Read the contents of a FASTA file.

writeFile(self, fout)

Write the information currently contained in the FASTA structure to a file as a FASTA-formatted file.

addSequence(self, name, seq)

Add a sequence to the FASTA object.

renameSequence(self, oldname, newname)

Renames a given sequence.

removeSequence(self, name)

Remove a sequence from the FASTA object and return it; or return None if it was not found.

reorderSequences(self, iterable)

Reorder all sequences by an iterable sequence of their names.

rer	$\mathbf{noveGaps}(\mathit{self})$
-----	------------------------------------

Remove the gaps for all sequences.

# $\mathbf{allUpper}(\mathit{self})$

Change all sequences to uppercase.

# $\mathbf{allLower}(self)$

Change all sequences to lowercase.

 $\_\_iter\_\_(self)$ 

Iterate through the FASTA by going through its sequences.

 $\underline{\phantom{a}}$  len $\underline{\phantom{a}}$  (self)

Return the number of sequences in the FASTA object.

\_\_\_str\_\_\_(self)

Return the FASTA object as FASTA file text content.

# 3 Module labblouin.IO

Different shortcut functions for dealing with file systems.

IO Python Library / 2012 / Alex Safatli

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <a href="http://www.gnu.org/licenses/">http://www.gnu.org/licenses/</a>.

E-mail: safatli@cs.dal.ca Dependencies: -

#### 3.1 Functions

$\boxed{\mathbf{getFileName}(path)}$
Returns a filename of a given path.
$\boxed{\mathbf{getFolderName}(\mathit{path})}$
Returns the directory name of a given path.
makeFolder(folder_name)
Makes a folder (if does not already exist). Returns path.
moveFile(origin, target)
Moves a file. Returns path.
copyFile(origin, target)
Copies a file. Returns path.
removeFile(origin)
Removes a file.

#### 3.2 Variables

getFilesInFolder(folder, ext)

Returns a list of all full file paths for a given extension in a given folder.

Variables Module labblouin.IO

Name	Description
package	Value: 'labblouin'

# 4 Module labblouin.Ordination

Ordination is a class designed to compute and plot ordination methods such as PCA and MDS. It is intended as a helper function to PDBnet, but have the functionality to work with gm files.

This file is based on Nelle Varoquaux <nelle.varoquaux@gmail.com> code plotmds.py, available at http://scikit-learn.org/stable/auto\_examples/manifold/plot\_mds.html, and recomendations in stackoverflow by Jaime Fernandez (http://numericalrecipes.wordpress.com/)

Dependencies: SKlearn, PDBnet, Scipy, matplotlib

Author: Jose Sergio Hleap email: jshleap@dal.ca

### 4.1 Variables

Name	Description
colors	Value: ['b', 'r', 'k', 'g', 'y', 'm', 'c',
	'#778899', '#EEC900',
hexa	Value: ['#778899', '#EEC900', '#CDC9A5',
	'#BF3EFF', '#CD69C9', '
package	Value: 'labblouin'

#### 4.2 Class ORD

A class for the most popular ordination methods using PDBnet instaces or gm files.

#### 4.2.1 Methods

$c, data, fastafile = None, n\_comp$	2=2
--------------------------------------	-----

PrepData4PDBnet(self, data)
Load the data to the class assuming is a PDBnet instance file

LoadDataGMfile(self, data)	
Load the data to the class assuming is a GM file	

$\mathbf{dict2array2matrix}(self,\ dict)$
Giving an upper-triangle distance matrix in a dictionary, returns a distance-like array

( 10)		
I Storol colt		
LOUDICISCALI		
1()		

MDS(self, options, typeof='classic', dist=False, groups=None)

Perform Multidimensional Scaling wither classic (PCoA) or non-metric. If you have the upper triangle of a distance matrix as a dictionary, pass the dictionary as dist.

**PCA**(self, options, groups=None)

Performs a principal coordinate analysis of the data

Plot(self, options, groups=None)

Plot the components from an ordination method of the class ORD. If the number of components is greater than 3, it will plot the first three components. Components has to be a n x k numpy array of eigenvectors, where n is the observations/individuals and k the components. The option groups allow to pass a list (of the same length of the arrar, that is a length of n).

PlotXDA(self, membership, options, group\_labels=None)

Plots a Linear Discriminant Analysis (LDA) or a Quadratic Discriminan Analysis (QDA) with confidence ellipses at std (standard deviations)

ellipse(self, singlegroupx, singlegroupy, std=2, color='k')

Create an ellipse given points with x coordinates in singlegroup and singlegroup

pointsInEllipse(self, Xs, Ys, ellipse)

Tests which set of points are within the boundaries defined by the ellipse. The set of points are defined in two arrays Xs and Ys for the x-coordinates and y-coordinates respectively

getEllipses(self, stds)

will populate the ellipses attribute

LDA(self, membership, options, group\_labels=None)

Perform a Linear discriminant analysis of the data and plot it. Membership must be an array of integers of the same length of the number of observations in the data.

**QDA**(self, membership, options, group\_labels=None)

# 5 Module labblouin.PDBS2VMDstate

PDB(s) to VMD state Giving a list of structures (can be a single one), return a VMD state, with the same color scale (from the minimum of all to the maximum of all), white background, and other visual hacks

# 5.1 Functions

$\mathbf{GetFDs}(\mathit{files})$
-----------------------------------

### 5.2 Variables

Name	Description			
InvariantLine	Value: '#!/usr/local/bin/vmd\n# VMD script			
	written by save_state			
molLine	Value: 'mol new PATHNPDBNAME type pdb first 0			
	last -1 step 1 fil			
lastline	Value: 'foreach v \$viewplist {\n molinfo \$v set			
	{center_matrix			
fout	Value: open(sys.argv [1]+ '.vmd', 'w')			
files	Value: sys.argv [2:]			
scale	Value: GetFDs(files)			
package	Value: 'labblouin'			

# 6 Module labblouin.PDBnet

PDBnet is a collection of Python objects intended to model and contain PDB protein data.

PDBnet Copyright (C) 2012 Christian Blouin Contributions by Alex Safatli and Jose Sergio Hleap Major Refactoring (2014) done by Alex Safatli and Jose Sergio Hleap

 $\label{lem:contained} \begin{tabular}{ll} E-mail: cblouin@cs.dal.ca, safatli@cs.dal.ca, jshleap@dal.ca Dependencies: Scipy, BioPython, FASTAnet (contained in LabBlouinTools) \\ \end{tabular}$ 

Version: 1.0.5

Authors: Christian Blouin, Jose Sergio Hleap, Alexander Safatli

#### 6.1 Variables

Name	Description
PDB_LARGE_FILE_SIZE	Value: 100000000
aa	Value: {'ALA': 'A', 'ARG': 'R', 'ASN': 'N',
	'ASP': 'D', 'CYS': '
aa_names	Value: {'A': 'ALA', 'C': 'CYS', 'D': 'ASP',
	'E': 'GLU', 'F': 'PH
aa_lists	Value: {'ALA': ['N', 'CA', 'C', 'O', 'CB'],
	'ARG': ['N', 'CA', '
package	Value: 'labblouin'

### 6.2 Class PDBatom

object | labblouin.PDBnet.PDBatom

Known Subclasses: labblouin.PDBnet.PDBterminator

ATOM in a PDB protein structure.

#### 6.2.1 Methods

init(self, serial, name, x, y, z, oc, b, symbol, charge)
Construct an atom in a PDB protein structure.
Overrides: objectinit

Express tills	atom as a single line in a PDB file (see PDB format).
Overrides: o	bjectstr
GetPositio	$\mathbf{n}(self)$
Get the 3-di	mensional coordinate of this atom in space.
DistanceTo	(self, atom)
Acquire the	distance from this atom to another atom.
erited from	n object
	(),format(),getattribute(),hash(),nev (),reduceex(),repr(),setattr(),sizeof

# 6.2.2 Properties

Name	Description
charge	
name	
occupancy	
parent	
serial	
symbol	
tempFactor	
X	
У	
Z	
Inherited from object	
class	

# 6.3 Class PDBterminator

object — labblouin.PDBnet.PDBatom — labblouin.PDBnet.PDBterminator

A placeholder class that represents a terminating ATOM-like line in the PDB file.

# 6.3.1 Methods

init(self, chaininst)			
Construct an atom in a PDB protein structure.			
Overrides: objectinit extit(inherited documentation)			
$\_\_\_str\_\_\_(self)$			
Express this atom as a single line in a PDB file (see PDB format).			
Overrides: objectstr extit(inherited documentation)			

# $Inherited\ from\ labblouin.PDBnet.PDBatom(Section\ 6.2)$

DistanceTo(), GetPosition(), fixname()

# $Inherited\ from\ object$

delattr(),	$\_{ m format}$	(),	_getattrib	$ute_{}()$	$, _{}$ hash	ı(), _	new_	():
reduce(),	_reduce_	_ex()	),repr_	(),	_setattr	_(),	_sizeof	_(),
subclasshook	_()							

# 6.3.2 Properties

Name	Description
charge	
lastatom	
lastreschain	
lastresind	
lastresname	
name	
occupancy	
parent	
serial	
symbol	
tempFactor	
X	
У	
Z	
Inherited from object	
class	

### 6.4 Class PDBresidue

A residue (collection of ATOM fields) in a PDB protein structure.

#### 6.4.1 Methods

\_init\_\_\_(*self*, *index*=None, *name*=',')

Construct a residue (collection of atoms) in a PDB protein structure.

GetAtoms(self)

AddAtom(self, atom)

Add a PDBatom structure to this residue.

GetCA(self)

Get the alpha-carbon found in this residue as a PDBatom.

Centroid(self)

Calculate the centroid of this residue. Return this as a PDBatom.

**InContactWith**(*self*, *other*, *thres*=4.5)

Determine if in contact with another residue.

\_\_\_int\_\_\_(self)

str (self)

#### 6.5 Class PDBchain

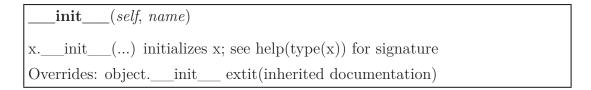
object —

labblouin.PDBnet.PDBchain

Known Subclasses: labblouin.PDBnet.PDBmodel

A PDB chain (collection of protein residues).

#### 6.5.1 Methods



# GetResidues(self)

Acquire all of the residues in this chain. Note that if this is a large file, this will force the object to load all of these residues into memory from the file.

# IterResidues(self)

Iteratively yield all of the residues in this chain. Note that if this is a large file, this will force the object to load all of these residues into memory from the file.

# GetResidueByIndex(self, index)

Alias for acquiring a residue from this instance using [] operator.

### GetAtoms(self)

Get all comprising atoms in this chain in order of residues.

### GetIndices(self)

Acquire all of the indices for residues present in this chain.

### AddIndexOfResidue(self, index)

Add an index for a residue to this chain. Will generate own residue information if called upon (forces lazy evaluation).

# AddResidue(self, resid)

Add a residue to this chain.

#### AddResidueByIndex(self, index)

Add a residue to this chain by its index in the PDB. The residue object will automatically be constructed from the file.

#### RemoveResidue(self, resid)

Remove a residue from this chain.

ContactMap(self, thres=4.5) Compute the contact map of this chain. GetPrimaryPropertiesFromBioPython(self)Use BioPython to populate this class with attributes. AsFASTA(self)Return the string representing this chain's FASTA sequence. WriteAsPDB(self, filename) Write this single chain to a file as a PDB. SortByNumericalIndices(self) Sort all internal items by a numerical index.  $\mathbf{pop}(self, s)$ Pop a residue.  $\mathbf{update}(\mathit{self}, o)$ Add residues from another chain. (self) $\operatorname{str}$ str(x)Overrides: object. str extit(inherited documentation)  $_{
m getitem}$ (self, i)Force the object to load the residue if not present in this structure. len (self)iter (self)

### Inherited from object

 $\label{eq:condition} $$ \_ delattr_{()}, \_ format_{()}, \_ getattribute_{()}, \_ hash_{()}, \_ new_{()}, \_ reduce_{()}, \_ repr_{()}, \_ setattr_{()}, \_ sizeof_{()}, $$$ subclasshook ()

#### 6.5.2 Properties

Name	Description
indices	
name	
parent	
residues	
structure	
Inherited from object	
class	

### 6.6 Class PDBmodel

object —	
labblouin.PDBnet.PDBchain	
	labblouin.PDBnet.PDBmodel

A PDB model (a special kind of chain).

#### 6.6.1 Methods

init	_(self, name)
$x_{\text{init}}$	_() initializes x; see help(type(x)) for signature
Overrides:	objectinit extit(inherited documentation)

### GetResidues(self)

Acquire all of the residues in this model. Note that if this is a large file, this will force the object to load all of these residues into memory from the file.

 $Overrides:\ labblouin. PDB net. PDB chain. Get Residues$ 

### IterResidues(self)

Iteratively yield all of the residues in this model. Note that if this is a large file, this will force the object to load all of these residues into memory from the file.

Overrides: labblouin.PDBnet.PDBchain.IterResidues

# $\mathbf{GetChains}(self)$

Get	tChain(self, name)
Ge	tChainByName(self, name)
Ge	tChainNames $(self)$
Ne	wChain(self, ch)
Cre	ate a new PDBchain, add it to this model, and return it.
Ad	dChain(self, chain)
Ada	d a PDBchain (chain) to this model.
Ad	$\mathbf{dResidue}(\mathit{self}, \mathit{resid})$
Ado	d a residue to this model.
Ove	errides: labblouin.PDBnet.PDBchain.AddResidue
	$getitem_{\underline{\hspace{1cm}}}(self, i)$
For	ce the object to load the residue if not present in this structure.
1	errides: labblouin.PDBnet.PDBchaingetitem extit(inherited umentation)
	$len_{\underline{\hspace{1cm}}}(self)$
Ove	errides: labblouin.PDBnet.PDBchainlen
	_str(self)
str(	$\mathbf{x}$ )
`	errides: objectstr extit(inherited documentation)
erite	$ed\ from\ labblouin.PDBnet.PDBchain(Section\ 6.5)$
Add Get:	IIndexOfResidue(), AddResidueByIndex(), AsFASTA(), ContactMap(), GetAton Indices(), GetPrimaryPropertiesFromBioPython(), GetResidueByIndex(), ReveResidue(), SortByNumericalIndices(), WriteAsPDB(),iter(), pop(), ate()
ierite	ed from object
	delattr(),format(),getattribute(),hash(),new(), reduce(),reduce_ex(),repr(),setattr(),sizeof(),

\_\_subclasshook\_\_\_()

# 6.6.2 Properties

Name	Description
chainNames	
chains	
indices	
name	
residues	
structure	
Inherited from labblouin.PDBnet.PDBchain (Section 6.5)	
parent	
Inherited from object	
class	

# 6.7 Class PDBstructure

object — labblouin.PDBnet.PDBstructure

A PDB protein structure (a collection of chains/models).

#### 6.7.1 Methods

init(self, filein=',')
xinit() initializes $x$ ; see $help(type(x))$ for signature
Overrides: objectinit extit(inherited documentation)

 $\mathbf{GetChainNames}(\mathit{self})$ 

 $\mathbf{GetModelNames}(self)$ 

Get Chain (self, ch)
Get a chain by name.

GetModel(self, mod)

Get a model by name.

**NewChain**(self, name)

Construct and add a new chain by name to the PDB. Returns the chain.

NewModel(self, name)

Construct and add a new model by name to the PDB. Returns the model.

RemoveChain(self, name)

Remove a chain from the structure (by name). Returns the chain.

RemoveModel(self, name)

Remove a model from the structure (by name). Returns the chain.

AddChain(self, chainname, chain)

Add a chain as a list of residues to the PDB.

AddModel(self, modelname, model)

Add a model as a list of residues to the PDB.

AddResidueToChain(self, chain, res)

Add a residue to a chain. Deprecated; use chain class function.

AddResidueToModel(self, model, res)

Add a residue to a model. Deprecated; use model class function.

AddRemark(self, remark)

Add a remark (note/comment) to the structure/PDB file.

GetRemarks(self)

Return all remarks from the PDB as a list of strings.

CheckComplete(self)

For every chain, check to see if every residue is complete (see aa\_list dictionary).

view(self, istrajectory=False)

View the structure in a Pymol window. Requires an installation of Pymol.

ViewStructure(self)

WriteFile(self, filename)

Write this PDB structure as a single PDB file.

**read**(self, filename)

Alias for ReadFile().

**ReadFile**(self, filename)

Read a PDB file. Populate this PDBstructure.

ChainAsFASTA(self, chain)

Return the chain as a FASTA. Deprecated; use chain class function.

ModelAsFASTA(self, model)

Return the model as a FASTA. Deprecated; use chain class function.

tmscore(self, fasta, chains=None, native=None, CA=True)

Get the TMscore between two chains. Requires a FASTA alignment and a value for the length of the native structure (e.g., for a pairwise alignment, the length of the structure used as a reference before alignment was done). The latter is computed by assuming the first of both provided chains is the native structure; otherwise, uses a provided chain name (native input).

gdt(self, fasta, chains=None, distcutoffs=[1, 2, 4, 8], CA=True)

Get the GDT score between two chains. Requires a FASTA alignment.

rmsd(self, fasta, chains=None, CA=True)

Get the RMSD between chains. Requires a FASTA alignment.

rrmsd(self, fasta, chains=None, CA=True)

Get the RRMSD between chains. Requires a FASTA alignment. See Betancourt & Skolnick, "Universal Similarity Measure for Comparison Protein Structures".

# GetAverage(self, chains=None, newname=None)

Acquire a new chain or model corresponding to an average of all present chains or models specified.

# RadiusOfGyration(self, chains=None)

Acquire the radius of the gyration of the entire, or a portion of, the PDB protein molecule.

# GetAllCentroid(self, chain)

Populates the centroids of all residues.

# IndexSeq(self, chain, fst)

Store in residues the correct index to the fasta. Requires a 1-to-1 correspondence at least a portion of the way through. Deprecated; use GetFASTAIndices().

### GetFASTAIndices(self, thing, fst)

Given a PDBchain, find 1-to-1 correspondances between it and a FASTA sequence.

# IterResiduesFor(self, chains=None)

Produce an iterator to allow one to iterate over all residues for a subset of the structure.

#### IterAllResidues(self)

Produce an iterator to allow one to iterate over all possible residues.

### gm(self, fasta, chains=None, CA=False, typeof='str')

Acquire Geometric Morphometric data corresponding to the (x,y,z) coordinates between all homologous residue positions. Requires a FASTA alignment. Options include using alpha-carbon positions. By default, uses centroids of residues. Returns a list of labels and a list of coordinates as raw GM data. The typeof option provides an option for coordinate output; they are returned as a semicolon-delimited string (str) or as a numpy 2d array (matrix).

# $\mathbf{WriteGM}(self, fasta, gm, chains=\mathtt{None}, CA=\mathtt{False})$

Write the information present in this PDB between multiple chains as a Geometric Morphometric text file. This file will be formatted such that individual lines correspond to chains and semi-colons separate the (x,y,z) coordinates between all homologous residue positions. Requires a FASTA alignment. Options include using alpha-carbon positions. By default, uses centroids of residues.

#### WriteLandmarks(self, fasta, lm, chains=None)

Write the information present in this PDB between multiple chains as a landmark text file. This file will be formatted such that the file is partitioned in sections starting with chain names and individual lines in these correspond to homologous residue positions denoted by homologous position, residue number, and residue name tab-delimited. Requires a FASTA file.

## FDmatrix(self, fasta, chains=None, scaled=True)

Compute the form difference matrix (FDM) as explained in Claude 2008. It relates to the identification of the most influential residue, with respect to the overall shape/structure. If the scaled option is True, will return an scaled list (from -1 to 1) of the of length equal to the number of residues. Otherwise will return the raw FDM, rounded so it can be included in a PDB. The scaled version is better for vizualization. By default the FDM is computed far all chains, but a subset can be passed to the chains option.

## Map2Protein(self, outname, lis, chain, fasta)

Map a list of values (lis), that must have a length equal to that of the number of residues in the PDB to be mapped (chain). If a list of list is provided, the first list will be mapped as the beta factor and the second as occupancy

#### Contacts(self, chain=None, thres=4.5)

Compute the contact map of all chains or a chain.

:param chain: A list of chain or model names or a single string or integer. By default, entire structure. :param thres: A threshold for distinguishing contact in Angstroms. :returns: A list of tuples of indices (integers) which correspond to chains or models and their residues.

#### WriteContacts(self, filename)

Write contact map.

iter(self)
Returns all PDBchains as an iterator.
$\underline{\underline{\hspace{1cm}}}$ len $\underline{\hspace{1cm}}$ (self)
Returns the length of the protein.
$\_\_str\_\_(self)$
As a string, outputs structure in the PDB format.
Overrides: objectstr
Inherited from object
delattr(),format(),getattribute(),hash(),new reduce(),reduce_ex(),repr(),setattr(),sizeof subclasshook()

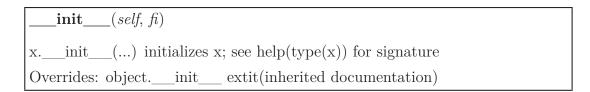
# 6.7.2 Properties

Name	Description
chains	
contactmap	
filepath	
handle	
ismodel	
models	
mutation	
orderofchains	
orderofmodels	
organism	
remarks	
taxid	
Inherited from object	
class	

# 6.8 Class PDBfile

 $\begin{array}{c} \text{object} \ \ \, \\ \text{labblouin.PDBnet.PDBfile} \end{array}$ 

#### 6.8.1 Methods



# isLargeFile(self)

Return whether or on the PDB file this object represents is incredibly large.

 $\mathbf{close}(\mathit{self})$ 

Close the file.

# read(self)

Acquire all remarks, and the indices of all models and residues. Returns remarks, and biological source information as a tuple (remarks, organism, taxid, mutant).

readResidue(self, chain, res, model=-1)

Parse a residue from the PDB file and return a PDB residue.

 $|\mathbf{hasModels}(self)|$ 

getModelNames(self)

 $\mathbf{getChainNames}(\mathit{self})$ 

# $\mathbf{iterResidueData}(\mathit{self})$

Yield the model number, chain name, and residue number for all residues present in this PDB file, not necessarily in order.

 ${\bf getResidueNamesInChain}(\mathit{self}, \mathit{ch})$ 

# $Inherited\ from\ object$

```
___delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce__ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()
```

# 6.8.2 Properties

Name	Description
chains	
fileHandle	
filePath	
memHandle	
modelIndices	
residueIndices	
size	
Inherited from object	
class	

# 7 Module labblouin.RegExpress

Prosite pattern generator. Will turn an alignment into a regular expression

# 7.1 Functions

 ${\bf check\_if\_is\_align}(\mathit{dictionary\_fasta})$ 

will check the read fasta to see if is an alignment

# read\_fasta(prefix)

Wll read the fasta, place sequences in a dictionary. Will break if not an alignment

 $strip\_gaps(matrix)$ 

given an alignment matrix, strip out the gap columns

# $reg\_express\_col(col)$

given a column of an alignment, return the string corresponding to the regular expression of such column

### $reg\_express(fastas, l)$

will create a matrix with the alignment, discard columns with gaps, and create a regular expression similar to prosite

#### 7.2 Variables

Name	Description
aa	Value: {'A': 'Ala', 'C': 'Cys', 'D':
	'Asp', 'E': 'Glu', 'F': 'Ph
allaa	Value: set(['A', 'C', 'D', 'E', 'F',
	'G', 'H', 'I', 'K', 'L', 'M
package	Value: 'labblouin'

# 8 Module labblouin.SeqMask

Get the amino acid sequence from a landmark file, align this profile to more sequences if required using muscle, and mask the membership vector into this alignment.

#### 8.1 Functions

# InferSingleLettterCode(threlettercode)

Convert the amino acid three letter code, into a single letter code

# InferSeqs\_landmarks(prefix)

Given a landmark file, return a dictionary with the name as key and the sequence as value

## dict2fasta(outfilename, dic)

Convert the dictionary return by InferSeqs\_landmarks fuction and writes a fasta file. The dictionary should have the seq name as key and a string as value containing the sequence

## **fasta2dict**(fastafilename)

convert a fasta file into a python dictionary, being the keys the name of the sequence and the values the sequence itself, in a list

#### Mem2List2tuple(prefix)

Read the membership vector, and returns a list and a dictionary with tupples for each module, indicating the indexes

#### str2list(string)

returns a list of a string that have no spaces between elements

### MaskAln(fastafilename, modulename, mem, tuples)

Strip the module in an alignment (fasta) file

# modseq2Fasta(prefix, modseq, names)

#### 8.2 Variables

Name	Description
package	Value: 'labblouin'

# 9 Module labblouin.expresso

An (incomplete) parsing library for output from the T-Coffee Expresso executable. expresso Python Library / Summer 2013 / Alex Safatli

E-mail: safatli@cs.dal.ca Dependencies: -

#### 9.1 Variables

Name	Description
package	Value: None

# 9.2 Class expressoParser

Parses Expresso output.

#### 9.2.1 Methods

init(self, fin)
-----------------

 $\_\_read\_\_\_(self)$ 

Read an Expresso output file; store the data inside this object.

getPDBcodes(self)

Return PDB code and chain as a tuple for each line.

 $\mathbf{writeNonEmpty}(\mathit{self},\mathit{fout})$ 

Writes non-empty PDB IDs to file.

 ${f getNonEmptySequenceNames}(self)$ 

Gets non-empty PDB names.

# 10 Module labblouin.homology

#### 10.1 Functions

# cleanFastaFile(fname, targetfolder=',')

Removes gaps from target FASTA file. Allows for BLAST input using this file. Creates new files for every sequence entry. Returns the filename(s) of the new file(s).

# writeModellerPIR(pdb\_files, seq, dest\_file)

# writeAlignmentFromFasta(fasta\_file, dest\_file)

Reads a FASTA file and outputs it to a Clustal-like alignment.

# writeSOAPSSAlignment(fasta\_file, pdb1, pdb2, dest\_file)

Reads a FASTA file and outputs to a SOAPSS alignment file.

# writeFirstFromFasta(fasta file, dest file)

Extracts first sequence from a FASTA and writes to a new one.

# extractRandomFasta(fasta\_file, numRandom, dest\_file)

Extracts a random number of sequences from a FASTA file. Writes to new FASTA.

### completePDB(pdbin, pdbout)

Given a PDB, clean/complete the PDB using Modeller's complete\_pdb function.

### getRandomPDBFragment(pdb, k=100)

Given a PDB, get a random fragment of length k.

#### getFirstResidueNumber(pdb)

Gets the FASTA residue number of the first ATOM in a PDB file.

### checkForFileCollision(path, ext)

Ensures a file is not being overwritten. Will recursively keep adding a '0' to end of base filename until a unique filename is found. Returns the path.

#### system(instruction)

Executes a system command line instruction. Returns list of stdout, stderr.

# cleanModellerFolder(path)

Moves intermediate PDBs to another subfolder in the Modeller folder output from Biskit.

# **getModellerPDB**(workflowfolder, targetfile)

Copies the model\_00.pdb file from Modeller if present in given workflow directory from homologyWorkflow.

# compareStructures(foldername1, foldername2, outpath, verbose=False)

Given 2 folders with PDB files, calculates the RMSDs with all possible pairwise combinations of PDBs; writes to file located at path determined by outpath.

# startPymol()

Ensures Pymol has been launched.

# radiusOfGyration(pdbFile, chain, pdb=None)

Returns the approximate radius of gyration of a given PDB molecule. Deprecated; now in PDBnet.

### tmscore(alignPDB, alignFASTA)

Returns the TMscore and associated P-value. See Xu & Zhang, "How significant is a protein structure similarity with TM-score = 0.5?".

### rrmsd(alignPDB, alignFASTA, rmsd=False, chains=None)

Returns the rRMSD of an aligned pairwise PDB and FASTA file. See Betancourt & Skolnick, "Universal Similarity Measure for Comparison Protein Structures". Deprecated.

### rmsd(pdbFile1, pdbFile2)

Returns the RMSD of two PDB files. Requires: Pymol.

#### 10.2 Variables

continued on next page

Name	Description
Name	Description
package	Value: 'labblouin'

# 10.3 Class manualModeller

Set up and run Modeller using a given folder of PDB files and a target FASTA sequence.

# 10.3.1 Methods

init(self, outfolder, targetfile, templatefildr)
$\_\_$ buildList $\_\_$ (self)
Build a string list of all PDB files.
$\_\_prepare\_\_(self)$
Prepare input for Modeller.
$\mathbf{run}(self)$
Run, call Modeller.
$\mathbf{postProcess}(\mathit{self})$
Make a copy of all found models and rename.

# 10.4 Class fasta

Open, read, and organize a FASTA file as an object.

#### 10.4.1 Methods

init(self, fname, removeGaps=True, allUpper=True)	
$\mathbf{read}(self, rgaps, allup)$	

### 11 Module labblouin.homstrad

A library to manage interfacing with the Homstrad database raw files.

homstrad Python Library / May 22, 2013 / Alex Safatli

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WAR-RANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <a href="http://www.gnu.org/licenses/">http://www.gnu.org/licenses/</a>>.

E-mail: safatli@cs.dal.ca Dependencies: IO (LabBlouinTools)

#### 11.1 Functions

$\boxed{\mathbf{has NoFolders}(\mathit{pa}}$	eth)
--	------

#### 11.2 Variables

Name	Description
package	Value: 'labblouin'

#### 11.3 Class homstradFolder

Model a single Homstrad folder or set.

#### 11.3.1 Methods

init(self, foldrname)	
manifest(self)	
getNames(self)	

${\bf getNumSequences}(self)$	
$\mathbf{getSequences}(self)$	
$\mathbf{getFiles}(\mathit{self})$	
$\mathbf{getPath}(self)$	
$\mathbf{getFASTA}(self)$	
writeFASTA(self, fi, names=None)	
getFASTAfor(self, names)	
${\bf getAlignedPDB}(\mathit{self})$	
${f getPDBs}(self)$	
$\mathbf{getPDBfor}(\mathit{self}, \mathit{name})$	
${\bf get Sequence Length}(\mathit{self})$	
${\bf getAlignmentLength}(self)$	
1.4 Class homstradDatabase	
1.4.1 Methods	
init(self, dbpath, traverse=True)	
iter(self)	
$\mathbf{traverse}(\mathit{self})$	
${f getPath}(\mathit{self})$	
${f getFolders}(self)$	

${\bf getFailedCount}(\mathit{self})$	
${\bf getSucceededCount}(self)$	

# 12 Module labblouin.loadingBar

## 12.1 Variables

Name	Description
package	Value: 'labblouin'

## 12.2 Class loadingBar

#### 12.2.1 Methods

init(self, title, width=300, height=25)
$\mathbf{open}(\mathit{self})$
open(seg)
$\mathbf{close}(\mathit{self})$
update(self, ratio)
update(seij, 1aiio)
settitle(self, title)

## 13 Module labblouin.logfile

### 13.1 Functions

$\mathbf{parseTime}(time)$	

| calculate Time(timevector)|

#### 13.2 Variables

Name	Description
package	Value: 'labblouin'

### 13.3 Class XMLfile

#### 13.3.1 Methods

\_\_\_init\_\_\_(self, xmlf, root='root')

 $\mathbf{compile}(\mathit{self})$ 

 $\mathbf{read}(self)$ 

 $\mathbf{write}(\mathit{self})$ 

add(self, ele, it, \*args)

### 13.4 Class logfile

#### 13.4.1 Methods

 $\_\_init\_\_\_(self, logf, enable = True, silent = False)$ 

 $\mathbf{setTotalNum}(\mathit{self}, \mathit{tn})$ 

incrementTimer(self)

updateTimer(self, numat)

\_\_\_fs\_\_\_(self, tot)

writeTemporary(self, msg, silent=None)

write(self, msg, silent=None)

writeElapsedTime(self)

### 14 Module labblouin.mdsa

A library to manage interfacing with the MDSA database raw files (http://dna.cs.byu.edu/mdsas/index.shtmdsa Python Library / Oct 8, 2014 / Alex Safatli

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WAR-RANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <a href="http://www.gnu.org/licenses/">http://www.gnu.org/licenses/</a>>.

E-mail: safatli@cs.dal.ca Dependencies: IO, FASTAnet (LabBlouinTools)

#### 14.1 Variables

Name	Description
mdsaTypes	Value: ['smart', 'balibase', 'oxbench',
	'prefab']
package	Value: 'labblouin'

#### 14.2 Class mdsaAlignment

Model a single MDSA alignment.

#### 14.2.1 Methods

init(self, finame)
$\mathbf{getObject}(self)$
getNames(self)
getNumSequences(self)
getSequences(self)

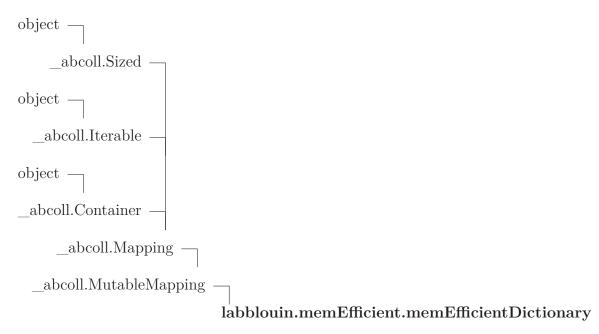
$\mathbf{getPath}(self)$
writeFASTA(self, fi, names=None)
getFASTAfor(self, names)
${\bf getSequenceLength}(self)$
${\bf getAlignmentLength}(self)$
14.3 Class mdsaDatabase
14.3 Class musaDatabase
14.3.1 Methods
init(self, dbpath, traverse=True)
iter(self)
$\mathbf{traverse}(self)$
$\mathbf{getPath}(self)$
$\mathbf{getFiles}(self)$

### 15 Module labblouin.memEfficient

#### 15.1 Variables

Name	Description
DICT_START_NUM	Value: 8
DICT_FREE_SLOT	Value: -1
DICT_DUMMY_SLOT	Value: -2
DICT_SHIFT	Value: 5
package	Value: 'labblouin'

#### 15.2 Class memEfficientDictionary



A memory-efficient alternative to the built-in Python dictionary with a trade-off for (insertion) performance. Based on recipe found by Raymond Hettinger (http://code.activestate.com/recipes/578 proof-of-concept-for-a-more-space-efficient -faster/).

#### 15.2.1 Methods

```
___init___(self, *args, **kwds)
x.___init___(...) initializes x; see help(type(x)) for signature
Overrides: object.___init___ extit(inherited documentation)
```

\_\_\_len\_\_\_(self)
Overrides: \_abcoll.Sized.\_\_\_len\_\_\_

\_\_\_iter\_\_\_(self)

Overrides: abcoll.Iterable. iter

iterkeys(self)

#### Return Value

an iterator over the keys of D

Overrides: \_abcoll.Mapping.iterkeys extit(inherited documentation)

 $\mathbf{keys}(self)$ 

#### Return Value

list of D's keys

Overrides: abcoll.Mapping.keys extit(inherited documentation)

itervalues(self)

#### Return Value

an iterator over the values of D

Overrides: \_abcoll.Mapping.itervalues extit(inherited documentation)

values(self)

#### Return Value

list of D's values

Overrides: \_abcoll.Mapping.values extit(inherited documentation)

iteritems(self)

#### Return Value

an iterator over the (key, value) items of D

Overrides: \_abcoll.Mapping.iteritems extit(inherited documentation)

items(self)

#### Return Value

list of D's (key, value) pairs, as 2-tuples

Overrides: \_abcoll.Mapping.items extit(inherited documentation)

contains (self, key)Overrides: abcoll.Container. contains get(self, key, default=None) d defaults to None. Return Value D[k] if k in D, else d Overrides: \_abcoll.Mapping.get extit(inherited documentation) popitem(self)as a 2-tuple; but raise KeyError if D is empty. Return Value (k, v), remove and return some (key, value) pair Overrides: abcoll.MutableMapping.popitem extit(inherited documentation) clear(self)Clear the data structure. Return Value None Overrides: abcoll.MutableMapping.clear  $_{
m getitem}$ (self, key)Access an item through the [] notation. Overrides: \_abcoll.Mapping.\_\_getitem\_ setitem (self, key, value) Overrides: \_abcoll.MutableMapping.\_ setitem  $_{\underline{\phantom{a}}}$  delitem\_\_\_(self, key) Overrides: \_abcoll.MutableMapping.\_\_ delitem  $Inherited\ from\ \_abcoll.Mutable Mapping$ pop(), setdefault(), update()  $Inherited\ from\ \_abcoll.Mapping$ \_\_\_eq\_\_\_(), \_\_\_ne\_\_\_()

Inh	$erited\ from\ \_abcoll.Sized$		
	$\_\_subclasshook\_\_()$		
Inh	erited from object		
		_(),getattribute(),new(),reduce (),setattr(),sizeof(),str()	_(),
15.2.	2 Properties		
	Name	Description	
	Inherited from object		
	class		

#### 15.2.3 Class Variables

Name	Description
abstractmethods	Value: frozenset([])
Inherited from _abcoll.Mapping	
hash	

## 16 Module labblouin.passToqsub

## 16.1 Functions

returnScript(command, jobname='default\_job', np=1, npmin=1,
nomail=True, sync=False)

## 16.2 Variables

Name	Description
package	Value: 'labblouin'

## 17 Module labblouin.pdbCompare

### 17.1 Functions

$\boxed{\mathbf{getChecksum}(\mathit{pdb\_file})}$
Given a PDB file, retrieve a unique checksum.

# $\mathbf{pdbCrosscheck}(folder1,\,folder2)$

Given two folder paths, determine what PDBs are unique amongst the two

outputResults(pdbcrosscheck\_results, fpath=None)

### 17.2 Variables

Name	Description	
package	Value: 'labblouin'	
warningregistry	Value: {('the md5 module is deprecated;	
	use hashlib instead', <t< th=""></t<>	

## 18 Module labblouin.pfam

#### 18.1 Functions

#### extractPDBSequences(pdb\_file)

Extracts the Amino Acid sequences for a protein structure defined by a PDB file and returns as a dictionary.

#### extractPDBChain(pdb\_file, ch, dest\_file)

Extracts a PDB file from another that has only information for a single chain.

#### writeSequencesToFile(seq\_list, targetfolder, fasta=True)

Takes in a dictionary of sequences and writes them to seperate .seq or .fasta files.

#### writeSequenceToFasta(seq, seq\_id, targetpath)

Creates FASTA file from given sequence string.

#### writeSequencesToFasta(seq\_list, dest\_file)

Creates single FASTA file from given sequence tuple list where first item is sequence ID, second item is the sequence.

#### doSequenceSearch(seq\_file)

Performs the PFAM search for a sequence file.

#### processSequenceSearch(search\_results)

Takes in a XML result returned from a sequence search and parses it in order to determine what the most significant results are for related families. Returns a list of the family PFAM IDs.

### downloadFamilySequences(pfam\_family\_id, dest\_folder, atype='seed')

Takes in PFAM family ID and acquires the gzipped flat file filled with seed or full sequence allignments.

#### decompressGzipFile(gzip\_file\_name, destfolder, file\_ext='ann')

Takes in GZIP file from PFAM or PDB and decompresses it to an .ann or specified file extension.

readPfamFile(file\_name)

Takes in PFAM Stockholm file and reads it, returning a pfamFile object.

 ${f grab}{f NCBIAccessionMetadata}(accession\_id)$ 

Takes in an accession ID and returns NCBI metadata from GenBank.

grabPDBFile(pdb\_code, dest\_folder)

Takes in a PDB code and acquires the PDB file from the PDB database, placing it in the specified destination folder.

printListToFile(list\_in, dest)

Prints a list to a file.

list2txttable(list\_in, title)

Converts a list to a string table.

removeListDuplicates(list\_in)

Removes all duplicates in a list.

#### 18.2 Variables

Name	Description
package	Value: 'labblouin'

### 18.3 Class pfamFile

#### 18.3.1 Methods

\_\_\_init\_\_\_(self, filepath)

 $\mathbf{parse}(self)$ 

 $\mathbf{cluster}(self)$ 

# $19 \quad {\bf Module\ labblouin.plotGM}$

this will plot a gm of a 2D shape

## 19.1 Variables

Name	Description
package	Value: 'labblouin'

## 20 Module labblouin.psub

Run as many of a input set of jobs/commands as there are cores at any given time (i.e. emulate a queue such as qsub present on GRID-powered hardware).

### 20.1 Functions

call(in etc)		
Can( ilisti)		

### 20.2 Variables

Name	Description
package	Value: 'labblouin'

### 21 Module labblouin.sabmark

#### 21.1 Variables

Name	Description
package	Value: 'labblouin'

### 21.2 Class sabmarkFile

#### 21.2.1 Methods

$$\underline{\hspace{1cm}}$$
 init $\underline{\hspace{1cm}}$  (self, path,  $l$ ,  $fp$ )

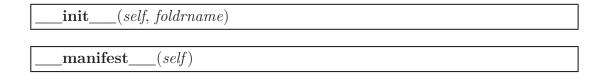
### 21.3 Class sabmarkSummary

#### 21.3.1 Methods

init(self, path)	
$\_$ read $\_$ ( $self$ )	

### 21.4 Class sabmarkFolder

#### 21.4.1 Methods



### 21.5 Class sabmarkDatabase

#### 21.5.1 Methods

$\_\_init\_\_\_(self, dbpath)$	
$\mathbf{traverse}(self)$	

groups(self, fkey)

# 22 Module labblouin.scop

### 22.1 Variables

Name	Description	
SCOP_VERSION	Value: '1.75'	
SCOP_PARSE_FILE	Value:	
	'http://scop.mrc-lmb.cam.ac.uk/scop/parse	/dir.des.scop.tx.
SCOP_PDBDB_FILE	Value:	
	'http://scop.mrc-lmb.cam.ac.uk/scop/parse	/dir.cla.scop.tx.
LOCAL_PARSE_FILE	Value: 'directory.scop'	
LOCAL_PDBDB_FILE	Value: 'pdbdb.scop'	
LOCAL_FLAT_FILE	Value: 'scop.pickl'	
package	Value: 'labblouin'	

## 22.2 Class scopHierarchy

#### 22.2.1 Methods

init(self, cacheloc)
$\boxed{ \_\_downloadparse}\_\_(\mathit{self})$
$\begin{tabular}{lllllllllllllllllllllllllllllllllll$
parsefile(self, fi)
pdbdbfile( <i>self</i> , <i>fi</i> )
readpickle(self, fi)
$\boxed{\textbf{populateHierarchy}(\textit{self})}$
getDissimilar(self, suprfquery, num=100)
$\boxed{\mathbf{query}(\textit{self}, \textit{dicti}, \textit{q})}$

etSimilar(self, suprfquery, num=100)
--------------------------------------

getFamilies(self, suprfquery)

### 22.2.2 Class Variables

Name	Description
classes	Value: {}
folds	Value: {}
superfamilies	Value: {}
families	Value: {}
domains	Value: {}
species	Value: {}
entries	Value: {}

## 22.3 Class scopItem

### 22.3.1 Methods

init(self, sunid, sccs, shortname, desc)
--

addPDB(self, pdb)

getPDBs(self)

Class sym model Module labblouin.sym

### 23 Module labblouin.svm

#### 23.1 Variables

Name	Description
package	Value: 'labblouin'

## 23.2 Class svm\_model

#### 23.2.1 Methods

 $\underline{\phantom{a}}$  init $\underline{\phantom{a}}$  (self, data=[], labels=[], kernel=2, c=10)

optimize(self, low=1, up=50, steps=5, by='proportions')

Optimize c-value on the basis of by (default:proportions of correct classifications).

predict(self, data)

| load(self, fin) |

save(self, fin)

### 23.3 Class cross\_validate

#### 23.3.1 Methods

\_\_init\_\_\_(self, model, test=[])

perform(self)

Perform the cross-validation.

Variables Module labblouin.timer

## 24 Module labblouin.timer

## 24.1 Functions

		$\overline{}$
		- 1
Locat Time()		- 1
get i iiie( )		- 1
• • • • • • • • • • • • • • • • • • •		- 1

estimateRemainingTime(starttime, numAt, totalNum)

## 24.2 Variables

Name	Description
package	Value: 'labblouin'

# $\mathbf{Index}$

labblouin (package), 2	labblouin.homology.writeFirstFromFasta
labblouin.expresso (module), 30 labblouin.expresso.expressoParser (class), 30	(function), 31 labblouin.homology.writeModellerPIR (func- tion), 31
labblouin.FASTAnet (module), 3–5	labblouin.homology.writeSOAPSSAlignment
labblouin.FASTAnet.FASTAsequence (class),	(function), 31
_ , , , , , , , , , , , , , , , , , , ,	abblouin.homstrad (module), 34–36
labblouin.FASTAnet.FASTAstructure (class),	labblouin.homstrad.hasNoFolders (func-
3-5	tion), 34
labblouin.homology (module), 31–33	labblouin.homstrad.homstradDatabase (class),
labblouin.homology.checkForFileCollision	35–36
(function), 31	labblouin.homstrad.homstradFolder (class),
labblouin.homology.clean Fasta File $(func-$	34–35
tion), 31	abblouin.IO (module), 6–7
labblouin.homology.cleanModellerFolder	labblouin.IO.copyFile (function), 6
(function), 32	labblouin.IO.getFileName (function), 6
labblouin.homology.compareStructures (function), 32	labblouin.IO.getFilesInFolder (function), 6
labblouin.homology.completePDB (function), 31	labblouin.IO.getFolderName (function), 6
labblouin.homology.extractRandomFasta	labblouin.IO.makeFolder (function), 6
(function), 31	labblouin.IO.moveFile (function), 6
labblouin.homology.fasta (class), 33	labblouin.IO.removeFile (function), 6
labblouin.homology.getFirstResidueNumber la	abblouin.loadingBar (module), 37
(function), 31	labblouin.loadingBar.loadingBar (class),
labblouin.homology.getModellerPDB (func-	37
	abblouin.logfile (module), 38–39
labblouin.homology.getRandomPDBFragment (function), 31	labblouin.logfile.calculateTime (function), 38
labblouin.homology.manualModeller (class),	labblouin.logfile.logfile (class), 38–39
33	labblouin.logfile.parseTime (function), 38
$labblouin. homology. radius Of Gyration \ (func-$	labblouin.logfile.XMLfile (class), 38
tion), 32	abblouin.mdsa (module), 40–41
labblouin.homology.rmsd (function), 32	labblouin.mdsa.mdsaAlignment (class),
labblouin.homology.rrmsd (function), 32	40-41
labblouin.homology.startPymol (function), 32	labblouin.mdsa.mdsaDatabase (class), 41 abblouin.memEfficient (module), 42–45
labblouin.homology.system (function), 31	labblouin. mem Efficient. mem Efficient Dictionary
labblouin.homology.tmscore (function),	(class), 42-45
	abblouin.Ordination (module), 8–9
labblouin.homology.writeAlignmentFromFasta (function), 31	a labblouin.Ordination.ORD (class), 8–9 abblouin.passToqsub (module), 46
	- , , , , ,

INDEX

labblouin.passToqsub.returnScript (function), 46	labblouin.pfam.pfamFile (class), 49 labblouin.pfam.printListToFile (function),
labblouin.pdbCompare (module), 47	49
labblouin.pdbCompare.getChecksum (func-	labblouin.pfam.processSequenceSearch (func-
tion), 47	tion), 48
labblouin.pdbCompare.outputResults (func-	labblouin.pfam.readPfamFile (function),
tion), 47	48
labblouin.pdbCompare.pdbCrosscheck (func-	labblouin.pfam.removeListDuplicates (func-
tion), 47	tion), 49
labblouin.PDBnet (module), 11–26	labblouin.pfam.writeSequencesToFasta (func-
labblouin.PDBnet.PDBatom (class), 11–	tion), 48
12	labblouin.pfam.writeSequencesToFile (func-
labblouin.PDBnet.PDBchain (class), 14-	tion), 48
17	labblouin.pfam.writeSequenceToFasta (func-
labblouin.PDBnet.PDBfile (class), 24–	tion), 48
26	labblouin.plotGM (module), 50
labblouin.PDBnet.PDBmodel (class), 17-	labblouin.psub (module), 51
19	labblouin.psub.call (function), 51
labblouin.PDBnet.PDBresidue (class),	labblouin.RegExpress (module), 27
13–14	labblouin.RegExpress.check_if_is_align
labblouin.PDBnet.PDBstructure (class),	(function), 27
19–24	labblouin.RegExpress.read_fasta (func-
labblouin. PDB net. PDB terminator $(class)$ ,	tion), 27
12–13	labblouin.RegExpress.reg_express (func-
labblouin.PDBS2VMDstate (module), 10	tion), 27
labblouin. PDBS2VMD state. GetFDs $(func-$	labblouin.RegExpress.reg_express_col (func-
tion), 10	tion), 27
labblouin.pfam (module), 48–49	labblouin.RegExpress.strip_gaps (func-
labblouin.pfam.decompressGzipFile $(func-$	tion), 27
	labblouin.sabmark (module), 52–53
labblouin.pfam.doSequenceSearch (func- tion), 48	labblouin.sabmark.sabmarkDatabase (class), 52–53
labblouin.pfam.downloadFamilySequences	labblouin.sabmark.sabmarkFile (class),
(function), 48	52
labblouin.pfam.extractPDBChain (func-	labblouin.sabmark.sabmarkFolder (class),
tion), 48	52
labblouin.pfam.extractPDBSequences (func-	labblouin.sabmark.sabmarkSummary (class),
tion), 48	52
labblouin.pfam.grabNCBIAccessionMetadatal	labblouin.scop (module), 54–55
(function), 49	labblouin.scop.scopHierarchy (class), 54–
labblouin.pfam.grabPDBFile (function),	55
49	labblouin.scop.scopItem (class), 55
labblouin.pfam.list2txttable (function),	labblouin.SeqMask (module), 28–29
49	$labblouin. Seq Mask. dict 2 fasta \ (function),$

INDEX

28 labblouin.SeqMask.fasta2dict (function), labblouin.SeqMask.InferSeqs\_landmarks (function), 28 labblouin. Seq Mask. In fer Single Lett ter Code(function), 28 labblouin.SeqMask.MaskAln (function), labblouin.SeqMask.Mem2List2tuple (function), 28 labblouin.SeqMask.modseq2Fasta (function), 28 labblouin.SeqMask.str2list (function), 28 labblouin.svm (module), 56 labblouin.svm.cross\_validate (class), 56 labblouin.svm\_model (class), 56 labblouin.timer (module), 57 labblouin.timer.estimateRemainingTime (function), 57 labblouin.timer.getTime (function), 57