Motif Conformation Finder

Release 1.0

Adriaan Lategan

Sep 14, 2023

CONTENTS

motif_conformations	Calculate the internal coordinates of each instance of a sequence motif in protein structures
pdb_io	Read and write CSV files and parse protein structures
internal_coord	Read and write CSV files and parse protein structures

CHAPTER

ONE

MOTIF_CONFORMATIONS

Calculate the internal coordinates of each instance of a sequence motif in protein structures

Author: Adriaan Lategan

Functions

get_arguments()	Fetch command-line arguments
motif_to_coordinates(motif, query_list, ...)	Read the protein structure files in the specified directory, search for the motif sequence in each structure and output the internal coordinates of the motifs as a csv file

1.1 motif_conformations.get_arguments

motif_conformations.**get_arguments**() → Namespace

Fetch command-line arguments

Return type
argparse.Namespace

1.2 motif_conformations.motif_to_coordinates

motif_conformations.**motif_to_coordinates**(*motif: str*, *query_list: str*, *pdb_directory: str*, *pdb_format: str*, *gzipped: bool*, *output_file: str*) → None

Read the protein structure files in the specified directory, search for the motif sequence in each structure and output the internal coordinates of the motifs as a csv file

Parameters

- **motif** (*str*) – an amino acid sequence to find in the protein structures
- **query_list** (*str*) – a csv file specifying protein file paths and polymer instances
- **pdb_directory** (*str*) – a directory containing protein structure files
- **pdb_format** (*str*) – the format of the protein structure files. Either "pdb" or "cif"
- **gzipped** (*bool*) – true if the protein structure files are compressed with gzip, false if uncompressed

- **output_file** (*str*) – name of the file to which the internal coordinates are written

CHAPTER

TWO

PDB_IO

Read and write CSV files and parse protein structures

Author: Adriaan Lategan

Functions

custom_format(message, category, filename, ...)	Overwrite the format of Warnings

2.1 pdb_io.custom_format

pdb_io.**custom_format**(*message: Warning | str*, *category: type[Warning]*, *filename: str*, *lineno: int*, **args*, ***kwargs*) → str

Overwrite the format of Warnings

Return type
str

Classes

CsvWriter(path_string, fields)	Object for writing values to defined fields in a csv file
PdbFileQuery(protein_id, file_path, chain_ids)	Object representing a specific proteins structure file and the amino protein chain entities that should be parsed from it
PdbQueries(query_list)	List with unique protein structure file query entries
PdbQueryCsv(chain_list[, has_header])	Object for parsing csv files listing protein entity IDs, paths to protein structure files, and protein chain IDs
PdbReader(pdb_directory[, pdb_type, gzipped])	Object for parsing protein structures in pdb or cif format from a specific directory
PolypeptideEntry(pdb_id, model, chain, ...)	Object representing a contiguous polymer of amino acids, with methods to extract amino acid sequences from it

2.2 pdb_io.CsvWriter

class pdb_io.**CsvWriter**(*path_string: str*, *fields: Iterable[str]*)

Bases: `object`

Object for writing values to defined fields in a csv file

Parameters

- **path_string** (*str*) – path of the output file to write
- **fields** (*Iterable[str]*) – column names of csv file

output_handle

text stream for writing output file

Type
IO

fields

column names of csv file

Type
Iterable[str]

__init__(*path_string: str*, *fields: Iterable[str]*)

Object for writing values to defined fields in a csv file

Parameters

- **path_string** (*str*) – path of the output file to write
- **fields** (*Iterable[str]*) – column names of csv file

output_handle

text stream for writing output file

Type
IO

fields

column names of csv file

Type
Iterable[str]

Methods

__init__(path_string, fields)	Object for writing values to defined fields in a csv file
close()	Close the TextIO stream
write_headings()	
write_line(field_values)	add a row to the csv format table and write it to the output file

close()

Close the TextIO stream

write_line(*field_values: Iterable[str]*)

add a row to the csv format table and write it to the output file

Parameters
field_values (*Iterable[str]*) – list containing a string value for each column

2.3 pdb_io.PdbFileQuery

class pdb_io.**PdbFileQuery**(*protein_id: str*, *file_path: str*, *chain_ids: list[str]*)

Bases: `object`

Object representing a specific proteins structure file and the amino protein chain entities that should be parsed from it

protein_id

pdb entity ID of the protein structure

Type
str

file_path

path to the protein structure file

Type
str

chain_ids

list of protein chain entities to parse

Type
list[str]

__init__(*protein_id: str*, *file_path: str*, *chain_ids: list[str]*) → None

Methods

__init__(protein_id, file_path, chain_ids)	
add_chain(chain_id)	
get_polypeptides(pdb_reader, builder)	Identify contiguous chains of amino acids in the protein chain entity
get_structure(pdb_reader)	Parse a protein structure file and return a biopython PDB structure

Attributes

protein_id
file_path
chain_ids

get_polypeptides(*pdb_reader:* PdbReader, *builder: PPBuilder | CaPPBuilder*) → Generator[*PolypeptideEntry*, None, None]

Identify contiguous chains of amino acids in the protein chain entity

Parameters

- **pdb_reader** (PdbReader) – protein structure file parser
- **builder** (*PDB.Polypeptide.PPBuilder or PDB.Polypeptide.CaPPBuilder*) – polypeptide constructor

Yields
PolypeptideEntry – contiguous chains of amino acids from the protein chain entry

get_structure(*pdb_reader:* PdbReader) → Structure | None

Parse a protein structure file and return a biopython PDB structure

Parameters
pdb_reader (*str*) – protein structure file parser

Returns
biopython structure entity

Return type
PDB.Structure.Structure or None

2.4 pdb_io.PdbQueries

class pdb_io.**PdbQueries**(*query_list: list = <factory>*)

Bases: object

List with unique protein structure file query entries

query_list

Type
list[*PdbFileQuery*]

__init__(*query_list: list = <factory>*) → None

Methods

__init__([query_list])	
add_query(protein_id, path_string, chain)	Creates a new PdbFileQuery object, or appends a protein chain entity ID to an existing query

Attributes

query_list

add_query(*protein_id: str*, *path_string: str*, *chain: str*)

Creates a new PdbFileQuery object, or appends a protein chain entity ID to an existing query

Parameters

- **protein_id** (*str*) – pdb entity ID of the protein structure
- **path_string** (*str*) – name of the protein structure file
- **chain** (*str*) – pdb polypeptide instance ID

Return type
None

2.5 pdb_io.PdbQueryCsv

class pdb_io.**PdbQueryCsv**(*chain_list: str*, *has_header: bool = True*)

Bases: `object`

Object for parsing csv files listing protein entity IDs, paths to protein structure files, and protein chain IDs

Parameters

- **chain_list** (*str*) – path to a csv file listing protein structures to parse
- **has_header** (*bool*) – flag to indicate whether the csv file has column headings. Default: True

read

path

path to a csv file listing protein structures to parse

Type
pathlib.Path

has_header

flag to indicate whether the csv file has column headings.

__init__(*chain_list: str*, *has_header: bool = True*) → None

Object for parsing csv files listing protein entity IDs, paths to protein structure files, and protein chain IDs

Parameters

- **chain_list** (*str*) – path to a csv file listing protein structures to parse
- **has_header** (*bool*) – flag to indicate whether the csv file has column headings. Default: True

read

path

path to a csv file listing protein structures to parse

Type
pathlib.Path

has_header

flag to indicate whether the csv file has column headings.

Methods

__init__(chain_list[, has_header])	Object for parsing csv files listing protein entity IDs, paths to protein structure files, and protein chain IDs

Attributes

read	Read a csv file listing PDB_IDs, paths, and chains

property read: *PdbQueries*

Read a csv file listing PDB_IDs, paths, and chains

Return type
PdbQueries

2.6 pdb_io.PdbReader

class pdb_io.**PdbReader**(*pdb_directory: str*, *pdb_type: str = 'cif'*, *gzipped: bool = True*)

Bases: object

Object for parsing protein structures in pdb or cif format from a specific directory

Parameters

- **pdb_directory** (*str*) – a directory containing protein structure files
- **pdb_type** (*str*) – the format of the protein structure files. Default "cif"
- **gzipped** (*bool*) – true if the protein structure files are compressed with gzip, false if uncompressed

directory_queries

pdb_directory

path to the directory containing protein structure files

Type
pathlib.Path

text_handler

method for opening text stream

Type
Callable

parser

Type
PDB.MMCIFParser or PDB.PDBParser

__init__(*pdb_directory: str*, *pdb_type: str = 'cif'*, *gzipped: bool = True*)

Object for parsing protein structures in pdb or cif format from a specific directory

Parameters

- **pdb_directory** (*str*) – a directory containing protein structure files
- **pdb_type** (*str*) – the format of the protein structure files. Default "cif"
- **gzipped** (*bool*) – true if the protein structure files are compressed with gzip, false if uncompressed

directory_queries

pdb_directory

path to the directory containing protein structure files

Type
pathlib.Path

text_handler

method for opening text stream

Type
Callable

parser

Type
PDB.MMCIFParser or PDB.PDBParser

Methods

__init__(pdb_directory[, pdb_type, gzipped])	Object for parsing protein structures in pdb or cif format from a specific directory
read_file(protein_id, file_name)	Parse a protein structure file and return a biopython PDB structure

Attributes

directory_queries	read each file in the directory and get the protein chain IDs and each protein structure file

property directory_queries

read each file in the directory and get the protein chain IDs and each protein structure file

Returns
the path to each protein structure file and the IDs of the polypeptide chains in that protein

Return type
PdbQueries

read_file(*protein_id: str*, *file_name: str*) → Structure | None

Parse a protein structure file and return a biopython PDB structure

Parameters

- **protein_id** (*str*) – pdb entity ID of the protein structure
- **file_name** (*str*) – name of the protein structure file

Returns
Biopython structure entity

Return type
PDB.Structure.Structure or None

2.7 pdb_io.PolypeptideEntry

class pdb_io.**PolypeptideEntry**(*pdb_id: str*, *model: int*, *chain: ~Bio.PDB.Chain.Chain*, *polypeptide: <module 'Bio.PDB.Polypeptide' from '/home/docs/checkouts/readthedocs.org/user_builds/motif-conformation-finder/envs/latest/lib/python3.11/site-packages/Bio/PDB/Polypeptide.py'>*)

Bases: object

Object representing a contiguous polymer of amino acids, with methods to extract amino acid sequences from it

pdb_id

pdb entity ID of the protein structure

Type
str

model

pdb model number

Type
int

chain

protein chain entity object

Type
PDB.Chain.Chain

polypeptide

a contiguous chain of amino acids from the protein chain entry

Type
PDB.Polypeptide

__init__(*pdb_id: str*, *model: int*, *chain: ~Bio.PDB.Chain.Chain*, *polypeptide: <module 'Bio.PDB.Polypeptide' from '/home/docs/checkouts/readthedocs.org/user_builds/motif-conformation-finder/envs/latest/lib/python3.11/site-packages/Bio/PDB/Polypeptide.py'>*) → None

Methods

__init__(pdb_id, model, chain, polypeptide)	
find_motif(motif)	find each instance of the sequence motif in the polypeptides

Attributes

sequence	returns: amino acid sequence of the polypeptide instance :rtype: str
pdb_id	
model	
chain	
polypeptide	

find_motif(*motif: str*) → Generator[list[Bio.PDB.Residue.Residue], None, None]

find each instance of the sequence motif in the polypeptides

Parameters
motif (*str*) –

Yields
list[PDB.Residue.Residue] – residues matching the motif from each polypeptide

property sequence: str

returns: amino acid sequence of the polypeptide instance :rtype: str

CHAPTER

THREE

INTERNAL_COORD

Read and write CSV files and parse protein structures

Author: Adriaan Lategan

Functions

get_bonds(hedra)	Extract groups of 2 atoms from the biopython internal coordinate data
get_coordinate(residue_letter, func, ...)	internal coordinate values for the specified atom chains
get_sidechain_triads(sidechain_atom_groups)	Extract groups of 3 atoms from the biopython internal coordinate sidechain data
get_triads(ic_data)	Extract groups of 3 atoms from the biopython internal coordinate data
split_coordinates(coordinate_type, coordinates)	generate CoordinateData object of each coordinate value

3.1 internal_coord.get_bonds

internal_coord.**get_bonds**(*hedra: list[tuple]*) → list[str]

Extract groups of 2 atoms from the biopython internal coordinate data

Parameters
hedra (*list[tuple]*) – List of groups of atoms that describe internal coordinates

Returns
list of bonded atom pairs

Return type
list[str]

3.2 internal_coord.get_coordinate

internal_coord.**get_coordinate**(*residue_letter: str*, *func: Callable[[tuple[Bio.PDB.internal_coords.AtomKey, Bio.PDB.internal_coords.AtomKey] | tuple[Bio.PDB.internal_coords.AtomKey, Bio.PDB.internal_coords.AtomKey, Bio.PDB.internal_coords.AtomKey] | tuple[Bio.PDB.internal_coords.AtomKey, Bio.PDB.internal_coords.AtomKey, Bio.PDB.internal_coords.AtomKey, Bio.PDB.internal_coords.AtomKey] | str], str | float | None]*, *backbone_hedra: list[str]*, *sidechain_hedra: dict[str, str]*) → dict[str, float]

internal coordinate values for the specified atom chains

Parameters

- **residue_letter** (*str*) – single letter amino acid name
- **func** (*Callable*) – a method of Bio.PDB.internal_coords.IC_Residue to retrieve the desired internal coordinate
- **backbone_hedra** (*list[str]*) – list of atom chains in the residue's backbone
- **sidechain_hedra** – list of atom chains in the residue's sidechain

Returns
dictionary with the coordinate value for each atom chain string

Return type
dict[str, float]

3.3 internal_coord.get_sidechain_triads

internal_coord.**get_sidechain_triads**(*sidechain_atom_groups: dict[str, list[tuple]]*) → dict[str, list[tuple]]

Extract groups of 3 atoms from the biopython internal coordinate sidechain data

Parameters
sidechain_atom_groups (*dict[str, list[tuple]]*) – list of bonded atom groups for the sidechain of each standard amino acid

Returns
list of groups of 3 bonded atoms for the sidechain of each standard amino acid

Return type
dict[str, list[tuple]]

3.4 internal_coord.get_triads

internal_coord.**get_triads**(*ic_data: Iterable*) → list[tuple[str, str, str]]

Extract groups of 3 atoms from the biopython internal coordinate data

Parameters
ic_data (*Iterable*) – Iterable containing groups of bonded atoms that describe internal coordinates

Returns
list of groups of 3 bonded atoms

Return type
list[tuple[str, str, str]]

3.5 internal_coord.split_coordinates

internal_coord.**split_coordinates**(*coordinate_type: str*, *coordinates: dict[str, float]*) → Generator[*CoordinateData*, None, None]

generate CoordinateData object of each coordinate value

Parameters

- **coordinate_type** (*str*) – type of coordinate, for example cartesian, dihedral angle, bond angle or bond length
- **coordinates** (*dict[str, float]*) – dictionary of coordinate identifiers and values

Yields
CoordinateData – Object for protein structure coordinates

Classes

CoordinateData(coordinate_type, ...)	Object for protein structure coordinates .
InternalCoordinates([backbone_atom_chains, ...])	Object for accessing residue internal coordinates
ResidueData(protein, model, chain, position, ...)	Object identifying an amino acid residue and recording its protein structure coordinates

3.6 internal_coord.CoordinateData

class internal_coord.**CoordinateData**(*coordinate_type: str*, *coordinate_id: str*, *coordinate_value: float*)

Bases: object

Object for protein structure coordinates .. attribute:: coordinate_type

type of coordinate, for example cartesian, dihedral angle, bond angle or bond length

type
str

coordinate_id

identifier for a specific coordinate value, such as x, y, z, CA:CB, N:C, psi, phi

Type
str

coordinate_value

floating point value of the coordinate

Type
float

__init__(*coordinate_type: str*, *coordinate_id: str*, *coordinate_value: float*) → None

Methods

__init__(coordinate_type, coordinate_id, ...)

Attributes

coordinate_type
coordinate_id
coordinate_value

3.7 internal_coord.InternalCoordinates

class internal_coord.**InternalCoordinates**(*backbone_atom_chains: tuple[tuple] = (('N', 'CA', 'C', 'O'), ('O', 'C', 'CA', 'CB'), ('CA', 'C', 'O'), ('CB', 'CA', 'C'), ('CA', 'C', 'OXT'), ('N', 'CA', 'C', 'OXT'), ('H', 'N', 'CA'), ('C', 'CA', 'N', 'H'), ('HA', 'CA', 'C'), ('O', 'C', 'CA', 'HA'), ('HA2', 'CA', 'C'), ('O', 'C', 'CA', 'HA2'), ('HA3', 'CA', 'C'), ('O', 'C', 'CA', 'HA3'), ('N', 'CA', 'CB'), ('N', 'CA', 'CB', 'HB'), ('N', 'CA', 'CB', 'HB1'), ('N', 'CA', 'CB', 'HB2'), ('N', 'CA', 'CB', 'HB3'), ('CA', 'CB', 'HB'), ('CA', 'CB', 'HB1'), ('CA', 'CB', 'HB2'), ('CA', 'CB', 'HB3'), ('H1', 'N', 'CA'), ('H2', 'N', 'CA'), ('H3', 'N', 'CA'), ('C', 'CA', 'N', 'H1'), ('C', 'CA', 'N', 'H2'), ('C', 'CA', 'N', 'H3'))*, *sidechain_atom_chains: dict[str, tuple] = {'C': (('CA', 'CB', 'SG'), ('N', 'CA', 'CB', 'SG', 'chi1'), ('CB', 'SG', 'HG'), ('CA', 'CB', 'SG', 'HG')), 'D': (('CA', 'CB', 'CG'), ('N', 'CA', 'CB', 'CG', 'chi1'), ('CB', 'CG', 'OD1'), ('CA', 'CB', 'CG', 'OD1', 'chi2'), ('CB', 'CG', 'OD2'), ('CA', 'CB', 'CG', 'OD2'), ('CG', 'OD2', 'HD2'), ('CB', 'CG', 'OD2', 'HD2')), 'E': (('CA', 'CB', 'CG'), ('N', 'CA', 'CB', 'CG', 'chi1'), ('CB', 'CG', 'CD'), ('CA', 'CB', 'CG', 'CD', 'chi2'), ('CG', 'CD', 'OE1'), ('CB', 'CG', 'CD', 'OE1', 'chi3'), ('CG', 'CD', 'OE2'), ('CB', 'CG', 'CD', 'OE2'), ('CB', 'CG', 'HG2'), ('CB', 'CG', 'HG3'), ('CA', 'CB', 'CG', 'HG2'), ('CA', 'CB', 'CG', 'HG3'), ('CD', 'OE2', 'HE2'), ('CG', 'CD', 'OE2', 'HE2')), 'F': (('CA', 'CB', 'CG'), ('N', 'CA', 'CB', 'CG', 'chi1'), ('CB', 'CG', 'CD1'), ('CA', 'CB', 'CG', 'CD1', 'chi2'), ('CG', 'CD1', 'CE1'), ('CB', 'CG', 'CD1', 'CE1'), ('CD1', 'CE1', 'CZ'), ('CG', 'CD1', 'CE1', 'CZ'), ('CB', 'CG', 'CD2'), ('CA', 'CB', 'CG', 'CD2'), ('CG', 'CD2', 'CE2'), ('CB', 'CG', 'CD2', 'CE2'), ('CG', 'CD1', 'HD1'), ('CB', 'CG', 'CD1', 'HD1'), ('CG', 'CD2', 'HD2'), ('CB', 'CG', 'CD2', 'HD2'), ('CD1', 'CE1', 'HE1'), ('CG', 'CD1', 'CE1', 'HE1'), ('CD2', 'CE2', 'HE2'), ('CG', 'CD2', 'CE2', 'HE2'), ('CE1', 'CZ', 'HZ'), ('CD1', 'CE1', 'CZ', 'HZ')), 'H': (('CA', 'CB', 'CG'), ('N', 'CA', 'CB', 'CG', 'chi1'), ('CB', 'CG', 'ND1'), ('CA', 'CB', 'CG', 'ND1', 'chi2'), ('CG', 'ND1', 'CE1'), ('CB', 'CG', 'ND1', 'CE1'), ('CB', 'CG', 'CD2'), ('CA', 'CB', 'CG', 'CD2'), ('CG', 'CD2', 'NE2'), ('CB', 'CG', 'CD2', 'NE2'), ('CG', 'ND1', 'HD1'), ('CB', 'CG', 'ND1', 'HD1'), ('CG', 'CD2', 'HD2'), ('CB', 'CG', 'CD2', 'HD2'), ('ND1', 'CE1', 'HE1'), ('CG', 'ND1', 'CE1', 'HE1'), ('CD2', 'NE2', 'HE2'), ('CG', 'CD2', 'NE2', 'HE2')), 'I': (('CA', 'CB', 'CG1'), ('N', 'CA', 'CB', 'CG1', 'chi1'), ('CB', 'CG1', 'CD1'), ('CA', 'CB', 'CG1', 'CD1', 'chi2'), ('CA', 'CB', 'CG2'), ('N', 'CA', 'CB', 'CG2'), ('CB', 'CG1', 'HG12'), ('CB', 'CG1', 'HG13'), ('CB', 'CG2', 'HG21'), ('CB', 'CG2', 'HG22'), ('CB', 'CG2', 'HG23'), ('CA', 'CB', 'CG1', 'HG12'), ('CA', 'CB', 'CG1', 'HG13'), ('CA', 'CB', 'CG2', 'HG21'), ('CA', 'CB', 'CG2', 'HG22'), ('CA', 'CB', 'CG2', 'HG23'), ('CG1', 'CD1', 'HD11'), ('CG1', 'CD1', 'HD12'), ('CG1', 'CD1', 'HD13'), ('CB', 'CG1', 'CD1', 'HD11'), ('CB', 'CG1', 'CD1', 'HD12'), ('CB', 'CG1', 'CD1', 'HD13')), 'K': (('CA', 'CB', 'CG'), ('N', 'CA', 'CB', 'CG', 'chi1'), ('CB', 'CG', 'CD'), ('CA', 'CB', 'CG', 'CD', 'chi2'), ('CG', 'CD', 'CE'), ('CB', 'CG', 'CD', 'CE', 'chi3'), ('CD', 'CE', 'NZ'), ('CG', 'CD', 'CE', 'NZ', 'chi4'), ('CB', 'CG', 'HG2'), ('CB', 'CG', 'HG3'), ('CA', 'CB', 'CG', 'HG2'), ('CA', 'CB', 'CG', 'HG3'), ('CG', 'CD', 'HD2'), ('CG', 'CD', 'HD3'), ('CB', 'CG', 'CD', 'HD2'), ('CB', 'CG', 'CD', 'HD3'), ('CD', 'CE', 'HE2'), ('CD', 'CE', 'HE3'), ('CG', 'CD', 'CE', 'HE2'), ('CG', 'CD', 'CE', 'HE3'), ('CE', 'NZ', 'HZ1'), ('CE', 'NZ', 'HZ2'), ('CE', 'NZ', 'HZ3'), ('CD', 'CE', 'NZ', 'HZ1'), ('CD', 'CE', 'NZ', 'HZ2'), ('CD', 'CE', 'NZ', 'HZ3')), 'L': (('CA', 'CB', 'CG'), ('N', 'CA', 'CB', 'CG', 'chi1'), ('CB', 'CG', 'CD1'), ('CA', 'CB', 'CG', 'CD1', 'chi2'), ('CB', 'CG', 'CD2'), ('CA', 'CB', 'CG', 'CD2'), ('CB', 'CG', 'HG'), ('CA', 'CB', 'CG', 'HG'), ('CG', 'CD1', 'HD11'), ('CG', 'CD1', 'HD12'),*

Bases: object

Object for accessing residue internal coordinates

Parameters

- **backbone_atom_chains** (*tuple[tuple]*) – backbone atom chains in a residue
- **sidechain_atom_chains** (*dict[str, tuple]*) – sidechain atom chains in each standard amino acid

backbone_bonds

backbone atom pair identifiers

Type
list[str]

backbone_angle_keys

identifiers for chains of 3 backbone atoms

Type
list[str]

sidechain_bonds

sidechain atom pair identifiers for each standard aminoacid

Type
dict[str, list[str]]

sidechain_angle_keys

identifiers for chains of 3 sidechain atoms for each standard amino acid

Type
dict[str, list[str]]

__init__(*backbone_atom_chains: tuple[tuple] = (('N', 'CA', 'C', 'O'), ('O', 'C', 'CA', 'CB'), ('CA', 'C', 'O'), ('CB', 'CA', 'C'), ('CA', 'C', 'OXT'), ('N', 'CA', 'C', 'OXT'), ('H', 'N', 'CA'), ('C', 'CA', 'N', 'H'), ('HA', 'CA', 'C'), ('O', 'C', 'CA', 'HA'), ('HA2', 'CA', 'C'), ('O', 'C', 'CA', 'HA2'), ('HA3', 'CA', 'C'), ('O', 'C', 'CA', 'HA3'), ('N', 'CA', 'CB'), ('N', 'CA', 'CB', 'HB'), ('N', 'CA', 'CB', 'HB1'), ('N', 'CA', 'CB', 'HB2'), ('N', 'CA', 'CB', 'HB3'), ('CA', 'CB', 'HB'), ('CA', 'CB', 'HB1'), ('CA', 'CB', 'HB2'), ('CA', 'CB', 'HB3'), ('H1', 'N', 'CA'), ('H2', 'N', 'CA'), ('H3', 'N', 'CA'), ('C', 'CA', 'N', 'H1'), ('C', 'CA', 'N', 'H2'), ('C', 'CA', 'N', 'H3'))*, *sidechain_atom_chains: dict[str, tuple] = {'C': (('CA', 'CB', 'SG'), ('N', 'CA', 'CB', 'SG', 'chi1'), ('CB', 'SG', 'HG'), ('CA', 'CB', 'SG', 'HG')), 'D': (('CA', 'CB', 'CG'), ('N', 'CA', 'CB', 'CG', 'chi1'), ('CB', 'CG', 'OD1'), ('CA', 'CB', 'CG', 'OD1', 'chi2'), ('CB', 'CG', 'OD2'), ('CA', 'CB', 'CG', 'OD2'), ('CG', 'OD2', 'HD2'), ('CB', 'CG', 'OD2', 'HD2')), 'E': (('CA', 'CB', 'CG'), ('N', 'CA', 'CB', 'CG', 'chi1'), ('CB', 'CG', 'CD'), ('CA', 'CB', 'CG', 'CD', 'chi2'), ('CG', 'CD', 'OE1'), ('CB', 'CG', 'CD', 'OE1', 'chi3'), ('CG', 'CD', 'OE2'), ('CB', 'CG', 'CD', 'OE2'), ('CB', 'CG', 'HG2'), ('CB', 'CG', 'HG3'), ('CA', 'CB', 'CG', 'HG2'), ('CA', 'CB', 'CG', 'HG3'), ('CD', 'OE2', 'HE2'), ('CG', 'CD', 'OE2', 'HE2')), 'F': (('CA', 'CB', 'CG'), ('N', 'CA', 'CB', 'CG', 'chi1'), ('CB', 'CG', 'CD1'), ('CA', 'CB', 'CG', 'CD1', 'chi2'), ('CG', 'CD1', 'CE1'), ('CB', 'CG', 'CD1', 'CE1'), ('CD1', 'CE1', 'CZ'), ('CG', 'CD1', 'CE1', 'CZ'), ('CB', 'CG', 'CD2'), ('CA', 'CB', 'CG', 'CD2'), ('CG', 'CD2', 'CE2'), ('CB', 'CG', 'CD2', 'CE2'), ('CG', 'CD1', 'HD1'), ('CB', 'CG', 'CD1', 'HD1'), ('CG', 'CD2', 'HD2'), ('CB', 'CG', 'CD2', 'HD2'), ('CD1', 'CE1', 'HE1'), ('CG', 'CD1', 'CE1', 'HE1'), ('CD2', 'CE2', 'HE2'), ('CG', 'CD2', 'CE2', 'HE2'), ('CE1', 'CZ', 'HZ'), ('CD1', 'CE1', 'CZ', 'HZ')), 'H': (('CA', 'CB', 'CG'), ('N', 'CA', 'CB', 'CG', 'chi1'), ('CB', 'CG', 'ND1'), ('CA', 'CB', 'CG', 'ND1', 'chi2'), ('CG', 'ND1', 'CE1'), ('CB', 'CG', 'ND1', 'CE1'), ('CB', 'CG', 'CD2'), ('CA', 'CB', 'CG', 'CD2'), ('CG', 'CD2', 'NE2'), ('CB', 'CG', 'CD2', 'NE2'), ('CG', 'ND1', 'HD1'), ('CB', 'CG', 'ND1', 'HD1'), ('CG', 'CD2', 'HD2'), ('CB', 'CG', 'CD2', 'HD2'), ('ND1', 'CE1', 'HE1'), ('CG', 'ND1', 'CE1', 'HE1'), ('CD2', 'NE2', 'HE2'), ('CG', 'CD2', 'NE2', 'HE2')), 'I': (('CA', 'CB', 'CG1'), ('N', 'CA', 'CB', 'CG1', 'chi1'), ('CB', 'CG1', 'CD1'), ('CA', 'CB', 'CG1', 'CD1', 'chi2'), ('CA', 'CB', 'CG2'), ('N', 'CA', 'CB', 'CG2'), ('CB', 'CG1', 'HG12'), ('CB', 'CG1', 'HG13'), ('CB', 'CG2', 'HG21'), ('CB', 'CG2', 'HG22'), ('CB', 'CG2', 'HG23'), ('CA', 'CB', 'CG1', 'HG12'), ('CA', 'CB', 'CG1', 'HG13'), ('CA', 'CB', 'CG2', 'HG21'), ('CA', 'CB', 'CG2', 'HG22'), ('CA', 'CB', 'CG2', 'HG23'), ('CG1', 'CD1', 'HD11'), ('CG1', 'CD1', 'HD12'), ('CG1', 'CD1', 'HD13'), ('CB', 'CG1', 'CD1', 'HD11'), ('CB', 'CG1', 'CD1', 'HD12'), ('CB', 'CG1', 'CD1', 'HD13')), 'K': (('CA', 'CB', 'CG'), ('N', 'CA', 'CB', 'CG', 'chi1'), ('CB', 'CG', 'CD'), ('CA', 'CB', 'CG', 'CD', 'chi2'), ('CG', 'CD', 'CE'), ('CB', 'CG', 'CD', 'CE', 'chi3'), ('CD', 'CE', 'NZ'), ('CG', 'CD', 'CE', 'NZ', 'chi4'), ('CB', 'CG', 'HG2'), ('CB', 'CG', 'HG3'), ('CA', 'CB', 'CG', 'HG2'), ('CA', 'CB', 'CG', 'HG3'), ('CG', 'CD', 'HD2'), ('CG', 'CD', 'HD3'), ('CB', 'CG', 'CD', 'HD2'), ('CB', 'CG', 'CD', 'HD3'), ('CD', 'CE', 'HE2'), ('CD', 'CE', 'HE3'), ('CG', 'CD', 'CE', 'HE2'), ('CG', 'CD', 'CE', 'HE3'), ('CE', 'NZ', 'HZ1'), ('CE', 'NZ', 'HZ2'), ('CE', 'NZ', 'HZ3'), ('CD', 'CE', 'NZ', 'HZ1'), ('CD', 'CE', 'NZ', 'HZ2'), ('CD', 'CE', 'NZ', 'HZ3')), 'L': (('CA', 'CB', 'CG'), ('N', 'CA', 'CB', 'CG', 'chi1'), ('CB', 'CG', 'CD1'), ('CA', 'CB', 'CG', 'CD1', 'chi2'), ('CB', 'CG', 'CD2'), ('CA', 'CB', 'CG', 'CD2'), ('CB', 'CG', 'HG'), ('CA', 'CB', 'CG', 'HG'), ('CG', 'CD1', 'HD11'), ('CG', 'CD1', 'HD12'), ('CG', 'CD1', 'HD13'), ('CG', 'CD2', 'HD21'), ('CG', 'CD2', 'HD22'), ('CG', 'CD2', 'HD23'), ('CB', 'CG', 'CD1', 'HD11'), ('CB', 'CG', 'CD1', 'HD12'), ('CB', 'CG', 'CD1', 'HD13'), ('CB', 'CG', 'CD2', 'HD21'), ('CB', 'CG', 'CD2', 'HD22'), ('CB', 'CG', 'CD2', 'HD23')), 'M': (('CA', 'CB', 'CG'), ('N', 'CA', 'CB', 'CG', 'chi1'), ('CB', 'CG', 'SD'), ('CA', 'CB', 'CG', 'SD', 'chi2'), ('CG', 'SD', 'CE'), ('CB', 'CG', 'SD', 'CE', 'chi3'), ('CB', 'CG', 'HG2'), ('CB', 'CG', 'HG3'), ('CA', 'CB', 'CG', 'HG2'), ('CA', 'CB', 'CG', 'HG3'), ('SD', 'CE', 'HE1'), ('SD', 'CE', 'HE2'), ('SD', 'CE', 'HE3'), ('CG', 'SD', 'CE', 'HE1'), ('CG', 'SD', 'CE', 'HE2'), ('CG', 'SD', 'CE', 'HE3')), 'N': (('CA', 'CB', 'CG'), ('N', 'CA', 'CB', 'CG', 'chi1'), ('CB', 'CG', 'OD1'), ('CA', 'CB', 'CG', 'OD1', 'chi2'), ('CB', 'CG', 'ND2'), ('CA', 'CB', 'CG', 'ND2'), ('CG', 'ND2', 'HD21'), ('CG', 'ND2', 'HD22'), ('CB', 'CG', 'ND2', 'HD21'), ('CB', 'CG', 'ND2', 'HD22')), 'P': (('CA', 'CB', 'CG'), ('N', 'CA', 'CB', 'CG', 'chi1'), ('CB', 'CG', 'CD'), ('CA', 'CB', 'CG', 'CD', 'chi2'), ('CB', 'CG', 'HG2'), ('CB', 'CG', 'HG3'), ('CA', 'CB', 'CG', 'HG2'), ('CA', 'CB', 'CG', 'HG3'), ('CG', 'CD', 'HD2'), ('CG', 'CD', 'HD3'), ('CB', 'CG', 'CD', 'HD2'), ('CB', 'CG', 'CD', 'HD3')), 'Q': (('CA', 'CB', 'CG'), ('N', 'CA', 'CB', 'CG', 'chi1'), ('CB', 'CG', 'CD'), ('CA', 'CB', 'CG', 'CD', 'chi2'), ('CG', 'CD', 'OE1'), ('CB', 'CG', 'CD', 'OE1', 'chi3'), ('CG', 'CD', 'NE2'), ('CB', 'CG', 'CD', 'NE2'), ('CB', 'CG', 'HG2'), ('CB', 'CG', 'HG3'), ('CA', 'CB', 'CG', 'HG2'), ('CA', 'CB', 'CG', 'HG3'), ('CD', 'NE2', 'HE21'), ('CD', 'NE2', 'HE22'), ('CG', 'CD', 'NE2', 'HE21'), ('CG', 'CD', 'NE2', 'HE22')), 'R': (('CA', 'CB', 'CG'), ('N', 'CA', 'CB', 'CG', 'chi1'), ('CB', 'CG', 'CD'), ('CA', 'CB', 'CG', 'CD', 'chi2'), ('CG', 'CD', 'NE'), ('CB', 'CG', 'CD', 'NE', 'chi3'), ('CD', 'NE', 'CZ'), ('CG', 'CD', 'NE', 'CZ', 'chi4'), ('NE', 'CZ', 'NH1'), ('CD', 'NE', 'CZ', 'NH1', 'chi5'), ('NE', 'CZ', 'NH2'), ('CD', 'NE', 'CZ', 'NH2'), ('CB', 'CG', 'HG2'), ('CB', 'CG', 'HG3'), ('CA', 'CB', 'CG', 'HG2'), ('CA', 'CB', 'CG', 'HG3'), ('CG', 'CD', 'HD2'), ('CG', 'CD', 'HD3'), ('CB', 'CG', 'CD', 'HD2'), ('CB', 'CG', 'CD', 'HD3'), ('CD', 'NE', 'HE'), ('CG', 'CD', 'NE', 'HE'), ('CZ', 'NH1', 'HH11'), ('CZ', 'NH1', 'HH12'),*

Object for accessing residue internal coordinates

Parameters

- **backbone_atom_chains** (*tuple[tuple]*) – backbone atom chains in a residue
- **sidechain_atom_chains** (*dict[str, tuple]*) – sidechain atom chains in each standard amino acid

backbone_bonds

backbone atom pair identifiers

Type
list[str]

backbone_angle_keys

identifiers for chains of 3 backbone atoms

Type
list[str]

sidechain_bonds

sidechain atom pair identifiers for each standard aminoacid

Type
dict[str, list[str]]

sidechain_angle_keys

identifiers for chains of 3 sidechain atoms for each standard amino acid

Type
dict[str, list[str]]

Methods

__init__([backbone_atom_chains, ...])	Object for accessing residue internal coordinates
get_coordinates(chain, residue)	internal coordinates of the residue

get_coordinates(*chain: <module 'Bio.PDB.Chain' from '/home/docs/checkouts/readthedocs.org/user_builds/motif-conformation-finder/envs/latest/lib/python3.11/site-packages/Bio/PDB/Chain.py'>*, *residue: <module 'Bio.PDB.Residue' from '/home/docs/checkouts/readthedocs.org/user_builds/motif-conformation-finder/envs/latest/lib/python3.11/site-packages/Bio/PDB/Residue.py'>*) → *ResidueData*

internal coordinates of the residue

Parameters

- **chain** (*Chain*) – Bio.PDB chain object
- **residue** (*Residue*) – Bio.PDB residue object

Returns
Object identifying an amino acid residue and recording its protein structure coordinates

Return type
ResidueData

3.8 internal_coord.ResidueData

class internal_coord.**ResidueData**(*protein: str*, *model: int*, *chain: str*, *position: int*, *residue_name: str*, *coordinates: Generator[CoordinateData, None, None]*)

Bases: object

Object identifying an amino acid residue and recording its protein structure coordinates

protein

pdb entity ID of the protein structure

Type
str

model

pdb model number

Type
int

chain

pdb polypeptide instance ID

Type
str

position

residue position index in the protein chain

Type
int

residue_name

amino acid type as its 3-letter name, e.g. ALA or PRO

Type
str

coordinates

protein structure coordinates describing the residue as atomic positions or internal angles

Type
Generator[*CoordinateData*, None, None]

__init__(*protein: str*, *model: int*, *chain: str*, *position: int*, *residue_name: str*, *coordinates: Generator[CoordinateData, None, None]*) → None

Methods

__init__(protein, model, chain, position, ...)

Attributes

protein
model
chain
position
residue_name
coordinates

CHAPTER

FOUR

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

INDEX

Symbols

A

B

C

D

F

G

H

I

M

O

P

Q

R

S

T

W