biobb$_{m}odel Documentation$

Release 3.7.0

Bioexcel Project

Nov 22, 2021

Contents

CHAPTER 1

Contents

1.1 biobb_model

1.1.1 Introduction

Biobb_model is the Biobb module collection to check and model 3d structures, create mutations or reconstruct missing atoms. Biobb (BioExcel building blocks) packages are Python building blocks that create new layer of compatibility and interoperability over popular bioinformatics tools. The latest documentation of this package can be found in our readthedocs site: latest API documentation.

1.1.2 Version

v3.7.0 2021.3

1.1.3 Installation

Using PIP:

> **Important:** PIP only installs the package. All the dependencies must be installed separately. To perform a complete installation, please use ANACONDA, DOCKER or SINGULARITY.

- Installation:

```
pip install "biobb_model>=3.7.0"
```

- Usage: Python API documentation

Using ANACONDA:

- Installation:

```
conda install -c bioconda "biobb_model>=3.7.0"
```

- Usage: With conda installation BioBBs can be used with the Python API documentation and the Command Line documentation

Using DOCKER:

- Installation:

```
docker pull quay.io/biocontainers/biobb_model:3.7.0--pyhdfd78af_0
```

- Usage:

```
docker run quay.io/biocontainers/biobb_model:3.7.0--pyhdfd78af_0 <command>
```

Using SINGULARITY:

MacOS users: it's strongly recommended to avoid Singularity and use **Docker** as containerization system.

- Installation:

```
singularity pull --name biobb_model.sif shub://bioexcel/biobb_model
```

- Usage:

```
singularity exec biobb_model.sif <command>
```

The command list and specification can be found at the Command Line documentation.

1.1.4 Copyright & Licensing

This software has been developed in the MMB group at the BSC & IRB for the European BioExcel, funded by the European Commission (EU H2020 823830, EU H2020 675728).

1.2 biobb_model

1.2.1 configuration package

Submodules

model.mutate module

Module containing the Mutate class and the command line interface.

class model.mutate.**Mutate**(*input_pdb_path: str*, *output_pdb_path: str*, *properties: dict = None*, ***kwargs*)

Bases: biobb_common.generic.biobb_object.BiobbObject

biobb_model Mutate

Class to mutate one amino acid by another in a 3d structure.

Mutate side chain with minimal atom replacement. if the use_modeller property is added the Modeller suite will be used to optimize the side chains.

Parameters

- **input_pdb_path** (*str*) – Input PDB file path. File type: input. Sample file. Accepted formats: pdb (edam:format_1476).
- **output_pdb_path** (*str*) – Output PDB file path. File type: output. Sample file. Accepted formats: pdb (edam:format_1476).
- **properties** (*dict - Python dictionary object containing the tool parameters, not input/output files*) –
  - **mutation_list** (*str*) - ("A:Val2Ala") Mutation list in the format "Chain:WT_AA_ThreeLeterCode Resnum MUT_AA_ThreeLeterCode" (no spaces between the elements) separated by commas. If no chain is provided as chain code all the chains in the pdb file will be mutated. ie: "A:ALA15CYS"
  - **use_modeller** (*bool*) - (False) Use Modeller suite to optimize the side chains.
  - **remove_tmp** (*bool*) - (True) [WF property] Remove temporal files.
  - **restart** (*bool*) - (False) [WF property] Do not execute if output files exist.

Examples

This is a use example of how to use the building block from Python:

```
from biobb_model.model.mutate import mutate
prop = { 'mutation_list': 'A:Val2Ala',
         'use_modeller': True }
mutate(input_pdb_path='/path/to/myStructure.pdb',
       output_pdb_path='/path/to/newStructure.pdb',
       properties=prop)
```

Info:

- **wrapped_software:**

- name: In house
- license: Apache-2.0

- **ontology:**
  - name: EDAM
  - schema: http://edamontology.org/EDAM.owl

launch() → int
Execute the *Mutate* object.

model.mutate.**main**()

model.mutate.**mutate**(*input_pdb_path: str*, *output_pdb_path: str*, *properties: dict = None*, ***kwargs*) → int
Create *Mutate* class and execute the *launch()* method.

model.fix_side_chain module

Module containing the FixSideChain class and the command line interface.

class model.fix_side_chain.**FixSideChain**(*input_pdb_path: str*, *output_pdb_path: str*, *properties: dict = None*, ***kwargs*)
Bases: biobb_common.generic.biobb_object.BiobbObject

biobb_model FixSideChain
Class to model the missing atoms in amino acid side chains of a PDB.
Model the missing atoms in amino acid side chains of a PDB using biobb_structure_checking if the use_modeller property is added the Modeller suite will also be used to rebuild the missing atoms.

Parameters

- **input_pdb_path** (*str*) – Input PDB file path. File type: input. Sample file. Accepted formats: pdb (edam:format_1476).
- **output_pdb_path** (*str*) – Output PDB file path. File type: output. Sample file. Accepted formats: pdb (edam:format_1476).
- **properties** (*dict - Python dictionary object containing the tool parameters, not input/output files*) –
  - **use_modeller** (*bool*) - (False) Use Modeller suite to rebuild the missing side chain atoms.
  - **remove_tmp** (*bool*) - (True) [WF property] Remove temporal files.
  - **restart** (*bool*) - (False) [WF property] Do not execute if output files exist.

Examples

This is a use example of how to use the building block from Python:

```
from biobb_model.model.fix_side_chain import fix_side_chain
prop = { 'use_modeller': True }
fix_side_chain(input_pdb_path='/path/to/myStructure.pdb',
               output_pdb_path='/path/to/newStructure.pdb',
               properties=prop)
```

Info:

- **wrapped_software:**
  - name: In house
  - license: Apache-2.0
- **ontology:**
  - name: EDAM
  - schema: http://edamontology.org/EDAM.owl

launch() → int
Execute the *FixSideChain* object.

model.fix_side_chain.**fix_side_chain**(*input_pdb_path: str*, *output_pdb_path: str*, *properties: dict = None*, ***kwargs*) → int
Create *FixSideChain* class and execute the *launch()* method.

model.fix_side_chain.**main**()

model.fix_backbone module

Module containing the FixBackbone class and the command line interface.

class model.fix_backbone.**FixBackbone**(*input_pdb_path: str*, *input_fasta_canonical_sequence_path: str*, *output_pdb_path: str*, *properties: dict = None*, ***kwargs*)
Bases: biobb_common.generic.biobb_object.BiobbObject

biobb_model FixBackbone
Class to model the missing atoms in the backbone of a PDB structure.
Model the missing atoms in the backbone of a PDB structure using biobb_structure_checking and the Modeller suite.

Parameters

- **input_pdb_path** (*str*) – Input PDB file path. File type: input. Sample file. Accepted formats: pdb (edam:format_1476).
- **input_fasta_canonical_sequence_path** (*str*) – Input FASTA file path. File type: input. Sample file. Accepted formats: fasta (edam:format_1476).
- **output_pdb_path** (*str*) – Output PDB file path. File type: output. Sample file. Accepted formats: pdb (edam:format_1476).
- **properties** (*dict - Python dictionary object containing the tool parameters, not input/output files*) –
  - **add_caps** (*bool*) - (False) Add caps to terminal residues.
  - **remove_tmp** (*bool*) - (True) [WF property] Remove temporal files.
  - **restart** (*bool*) - (False) [WF property] Do not execute if output files exist.

Examples

This is a use example of how to use the building block from Python:

```
from biobb_model.model.fix_backbone import fix_backbone
prop = { 'restart': False }
fix_backbone(input_pdb_path='/path/to/myStructure.pdb',
             input_fasta_canonical_sequence_path='/path/to/myCanonicalSequence.
↪fasta',
              output_pdb_path='/path/to/newStructure.pdb',
              properties=prop)
```

Info:

- **wrapped_software:**
  - name: In house
  - license: Apache-2.0
- **ontology:**
  - name: EDAM
  - schema: http://edamontology.org/EDAM.owl

launch() → int
Execute the *FixBackbone* object.

model.fix_backbone.**fix_backbone**(*input_pdb_path: str*, *input_fasta_canonical_sequence_path: str*, *output_pdb_path: str*, *properties: dict = None*, ***kwargs*) → int
Create *FixBackbone* class and execute the *launch()* method.

model.fix_backbone.**main**()

model.fix_amides module

Module containing the FixAmides class and the command line interface.

class model.fix_amides.**FixAmides**(*input_pdb_path: str*, *output_pdb_path: str*, *properties: dict = None*, ***kwargs*)
Bases: biobb_common.generic.biobb_object.BiobbObject

biobb_model FixAmides
Fix amide groups from residues.
Flip the clashing amide groups to avoid clashes.

Parameters

- **input_pdb_path** (*str*) – Input PDB file path. File type: input. Sample file. Accepted formats: pdb (edam:format_1476).
- **output_pdb_path** (*str*) – Output PDB file path. File type: output. Sample file. Accepted formats: pdb (edam:format_1476).
- **properties** (*dict - Python dictionary object containing the tool parameters, not input/output files*) –

- **remove_tmp** (*bool*) - (True) [WF property] Remove temporal files.
- **restart** (*bool*) - (False) [WF property] Do not execute if output files exist.

Examples

This is a use example of how to use the building block from Python:

```
from biobb_model.model.fix_amides import fix_amides
prop = { 'restart': False }
fix_amides(input_pdb_path='/path/to/myStructure.pdb',
           output_pdb_path='/path/to/newStructure.pdb',
           properties=prop)
```

Info:

- **wrapped_software:**
  - name: In house
  - license: Apache-2.0
- **ontology:**
  - name: EDAM
  - schema: http://edamontology.org/EDAM.owl

launch() → int
Execute the *FixAmides* object.

model.fix_amides.**fix_amides**(*input_pdb_path: str*, *output_pdb_path: str*, *properties: dict = None*, ***kwargs*) → int
Create *FixAmides* class and execute the *launch()* method.

model.fix_amides.**main**()

model.fix_chirality module

Module containing the FixChirality class and the command line interface.

class model.fix_chirality.**FixChirality**(*input_pdb_path: str*, *output_pdb_path: str*, *properties: dict = None*, ***kwargs*)
Bases: biobb_common.generic.biobb_object.BiobbObject

biobb_model FixChirality
Fix chirality errors of residues.
Fix stereochemical errors in residue side-chains changing It's chirality.

Parameters

- **input_pdb_path** (*str*) – Input PDB file path. File type: input. Sample file. Accepted formats: pdb (edam:format_1476).
- **output_pdb_path** (*str*) – Output PDB file path. File type: output. Sample file. Accepted formats: pdb (edam:format_1476).

- **properties** (*dict - Python dictionary object containing the tool parameters, not input/output files*) –
  - **remove_tmp** (*bool*) - (True) [WF property] Remove temporal files.
  - **restart** (*bool*) - (False) [WF property] Do not execute if output files exist.

Examples

This is a use example of how to use the building block from Python:

```
from biobb_model.model.fix_chirality import fix_chirality
prop = { 'restart': False }
fix_chirality(input_pdb_path='/path/to/myStructure.pdb',
              output_pdb_path='/path/to/newStructure.pdb',
              properties=prop)
```

Info:

- **wrapped_software:**
  - name: In house
  - license: Apache-2.0
- **ontology:**
  - name: EDAM
  - schema: http://edamontology.org/EDAM.owl

launch() → int
Execute the FixChirality object.

model.fix_chirality.**fix_chirality**(*input_pdb_path: str*, *output_pdb_path: str*, *properties: dict = None*, ***kwargs*) → int
Create FixChirality class and execute the launch() method.

model.fix_chirality.**main**()

model.checking_log module

Module containing the CheckingLog class and the command line interface.

class model.checking_log.**CheckingLog**(*input_pdb_path: str*, *output_log_path: str*, *properties: dict = None*, ***kwargs*)
Bases: biobb_common.generic.biobb_object.BiobbObject

biobb_model CheckingLog
Class to check the errors of a PDB structure.
Check the errors of a PDB structure and create a report log file.

Parameters

- **input_pdb_path** (*str*) – Input PDB file path. File type: input. Sample file. Accepted formats: pdb (edam:format_1476).

- **output_log_path** (*str*) – Output report log file path. File type: output. Sample file. Accepted formats: log (edam:format_2330).
- **properties** (*dict - Python dictionary object containing the tool parameters, not input/output files*) –
  - **remove_tmp** (*bool*) - (True) [WF property] Remove temporal files.
  - **restart** (*bool*) - (False) [WF property] Do not execute if output files exist.

Examples

This is a use example of how to use the building block from Python:

```
from biobb_model.model.checking_log import checking_log
prop = { 'restart': False }
checking_log(input_pdb_path='/path/to/myStructure.pdb',
             output_log_path='/path/to/myReport.log',
             properties=prop)
```

Info:

- **wrapped_software:**
  - name: In house
  - license: Apache-2.0
- **ontology:**
  - name: EDAM
  - schema: http://edamontology.org/EDAM.owl

launch() → int
Execute the *CheckingLog* object.

model.checking_log.**checking_log**(*input_pdb_path: str*, *output_log_path: str*, *properties: dict = None*, ***kwargs*) → int
Create *CheckingLog* class and execute the *launch()* method.

model.checking_log.**main**()

1.3 BioBB MODEL Command Line Help

Generic usage:

```
biobb_command [-h] --config CONFIG --input_file(s) <input_file(s)> --output_file
↪<output_file>
```

1.3.1 Fix_amides

Fix amide groups from residues.

Get help

Command:

```
fix_amides -h
```

```
/bin/sh: fix_amides: command not found
```

I / O Arguments

Syntax: input_argument (datatype) : Definition

Config input / output arguments for this building block:

- **input_pdb_path** (*string*): Input PDB file path. File type: input. Sample file. Accepted formats: PDB
- **output_pdb_path** (*string*): Output PDB file path. File type: output. Sample file. Accepted formats: PDB

Config

Syntax: input_parameter (datatype) - (default_value) Definition

Config parameters for this building block:

- **remove_tmp** (*boolean*): (True) Remove temporal files..
- **restart** (*boolean*): (False) Do not execute if output files exist..

YAML

Common config file

```
properties:
  restart: false
```

Command line

```
fix_amides --config config_fix_amides.yml --input_pdb_path 5s2z.pdb --output_pdb_path␣
↪output_amide_pdb_path.pdb
```

JSON

Common config file

```
{
  "properties": {
    "restart": false
  }
}
```

Command line

```
fix_amides --config config_fix_amides.json --input_pdb_path 5s2z.pdb --output_pdb_
↪path output_amide_pdb_path.pdb
```

1.3.2 Fix_chirality

Fix chirality errors of residues.

Get help

Command:

```
fix_chirality -h
```

```
/bin/sh: fix_chirality: command not found
```

I / O Arguments

Syntax: input_argument (datatype) : Definition

Config input / output arguments for this building block:

- **input_pdb_path** (*string*): Input PDB file path. File type: input. Sample file. Accepted formats: PDB
- **output_pdb_path** (*string*): Output PDB file path. File type: output. Sample file. Accepted formats: PDB

Config

Syntax: input_parameter (datatype) - (default_value) Definition

Config parameters for this building block:

- **remove_tmp** (*boolean*): (True) Remove temporal files..
- **restart** (*boolean*): (False) Do not execute if output files exist..

YAML

Common config file

```
properties:
  restart: false
```

Command line

```
fix_chirality --config config_fix_chirality.yml --input_pdb_path 5s2z.pdb --output_
↪pdb_path output_amide_pdb_path.pdb
```

JSON

Common config file

```
{
  "properties": {
    "restart": false
  }
}
```

Command line

```
fix_chirality --config config_fix_chirality.json --input_pdb_path 5s2z.pdb --output_
↪pdb_path output_amide_pdb_path.pdb
```

1.3.3 Checking_log

Class to check the errors of a PDB structure.

Get help

Command:

```
checking_log -h
```

```
/bin/sh: checking_log: command not found
```

I / O Arguments

Syntax: input_argument (datatype) : Definition

Config input / output arguments for this building block:

- **input_pdb_path** (*string*): Input PDB file path. File type: input. Sample file. Accepted formats: PDB
- **output_log_path** (*string*): Output report log file path. File type: output. Sample file. Accepted formats: LOG

Config

Syntax: input_parameter (datatype) - (default_value) Definition

Config parameters for this building block:

- **remove_tmp** (*boolean*): (True) Remove temporal files..
- **restart** (*boolean*): (False) Do not execute if output files exist..

YAML

Common config file

```
properties:
  restart: false
```

Command line

```
checking_log --config config_checking_log.yml --input_pdb_path 2ki5.pdb --output_log_
↪path checking.log
```

JSON

Common config file

```
{
  "properties": {
    "restart": false
  }
}
```

Command line

```
checking_log --config config_checking_log.json --input_pdb_path 2ki5.pdb --output_log_
↪path checking.log
```

1.3.4 Fix_backbone

Class to model the missing atoms in the backbone of a PDB structure.

Get help

Command:

```
fix_backbone -h
```

```
/bin/sh: fix_backbone: command not found
```

I / O Arguments

Syntax: input_argument (datatype) : Definition

Config input / output arguments for this building block:

- **input_pdb_path** (*string*): Input PDB file path. File type: input. Sample file. Accepted formats: PDB

- **input_fasta_canonical_sequence_path** (*string*): Input FASTA file path. File type: input. Sample file. Accepted formats: FASTA
- **output_pdb_path** (*string*): Output PDB file path. File type: output. Sample file. Accepted formats: PDB

Config

Syntax: input_parameter (datatype) - (default_value) Definition

Config parameters for this building block:

- **add_caps** (*boolean*): (False) Add caps to terminal residues..
- **remove_tmp** (*boolean*): (True) Remove temporal files..
- **restart** (*boolean*): (False) Do not execute if output files exist..

YAML

Common config file

```
properties:
  add_caps: true
  restart: false
```

Command line

```
fix_backbone --config config_fix_backbone.yml --input_pdb_path 2ki5.pdb --input_fasta_
↪canonical_sequence_path 2ki5.fasta --output_pdb_path output_pdb_path.pdb
```

JSON

Common config file

```
{
  "properties": {
    "restart": false,
    "add_caps": true
  }
}
```

Command line

```
fix_backbone --config config_fix_backbone.json --input_pdb_path 2ki5.pdb --input_
↪fasta_canonical_sequence_path 2ki5.fasta --output_pdb_path output_pdb_path.pdb
```

1.3.5 Mutate

Class to mutate one amino acid by another in a 3d structure.

Get help

Command:

```
mutate -h
```

```
/bin/sh: mutate: command not found
```

I / O Arguments

Syntax: input_argument (datatype) : Definition

Config input / output arguments for this building block:

- **input_pdb_path** (*string*): Input PDB file path. File type: input. Sample file. Accepted formats: PDB
- **output_pdb_path** (*string*): Output PDB file path. File type: output. Sample file. Accepted formats: PDB

Config

Syntax: input_parameter (datatype) - (default_value) Definition

Config parameters for this building block:

- **mutation_list** (*string*): (A:Val2Ala) Mutation list in the format "Chain:WT_AA_ThreeLeterCode Resnum MUT_AA_ThreeLeterCode" (no spaces between the elements) separated by commas. If no chain is provided as chain code all the chains in the pdb file will be mutated. ie: "A:ALA15CYS".
- **use_modeller** (*boolean*): (False) Use Modeller suite to optimize the side chains..
- **remove_tmp** (*boolean*): (True) Remove temporal files..
- **restart** (*boolean*): (False) Do not execute if output files exist..

YAML

Common config file

```
properties:
  mutation_list: Leu49Ile, B:arg51Lys
```

Command line

```
mutate --config config_mutate.yml --input_pdb_path 2ki5.pdb --output_pdb_path output_
↪mutated_pdb_path.pdb
```

JSON

Common config file

```
{
  "properties": {
    "mutation_list": "Leu49Ile, B:arg51Lys"
  }
}
```

Command line

```
mutate --config config_mutate.json --input_pdb_path 2ki5.pdb --output_pdb_path output_
↪mutated_pdb_path.pdb
```

1.3.6 Fix_side_chain

Class to model the missing atoms in amino acid side chains of a PDB.

Get help

Command:

```
fix_side_chain -h
```

```
/bin/sh: fix_side_chain: command not found
```

I / O Arguments

Syntax: input_argument (datatype) : Definition

Config input / output arguments for this building block:

- **input_pdb_path** (*string*): Input PDB file path. File type: input. Sample file. Accepted formats: PDB
- **output_pdb_path** (*string*): Output PDB file path. File type: output. Sample file. Accepted formats: PDB

Config

Syntax: input_parameter (datatype) - (default_value) Definition

Config parameters for this building block:

- **use_modeller** (*boolean*): (False) Use Modeller suite to rebuild the missing side chain atoms..
- **remove_tmp** (*boolean*): (True) Remove temporal files..
- **restart** (*boolean*): (False) Do not execute if output files exist..

YAML

Common config file

```
properties:
  restart: false
```

Command line

```
fix_side_chain --config config_fix_side_chain.yml --input_pdb_path 2ki5.pdb --output_
↪pdb_path output_pdb_path.pdb
```

JSON

Common config file

```
{
  "properties": {
    "restart": false
  }
}
```

Command line

```
fix_side_chain --config config_fix_side_chain.json --input_pdb_path 2ki5.pdb --output_
↪pdb_path output_pdb_path.pdb
```

1.4 Biobb Model changelog

1.4.1 What's new in version 3.7.0?

In version 3.7.0 the dependency biobb_common has been updated to 3.7.0 version.

New features

- Update to biobb_common 3.7.0 (general)
- Update to biobb_structure_checking 3.8.5 (general)

1.4.2 What's new in version 3.0.1?

In version 3.0.0 Python has been updated to version 3.7 and Biopython to version 1.76. Big changes in the documentation style and content. Finally a new conda installation recipe has been introduced.

New features

- Update to Python 3.7 (general)
- Update to Biopython 1.76 (general)

- New conda installer (installation)
- Adding type hinting for easier usage (API)
- Deprecating os.path in favour of pathlib.path (modules)
- New command line documentation (documentation)

Bug fixes

- Replace container Quay.io badge (documentation)
- Remove unused system and step arguments from command line causing execution errors (cli) #9
- Remove system argument from commandline (cli)

Other changes

- New documentation styles (documentation) #8

CHAPTER 2

Indices and tables

- genindex
- modindex
- search

CHAPTER 3

Github repository.

Python Module Index

m

Index