biobb_wfmdsetupDocumentation

Release 1.0.0

Bioexcel Project

Jan 07, 2021

Contents

CHAPTER 1

Contents

1.1 Automatic Ligand parameterization tutorial using BioExcel Building Blocks (biobb)

This tutorial aims to illustrate the process of **ligand parameterization** for a **small molecule**, step by step, using the **BioExcel Building Blocks library (biobb)**. The particular example used is the **Sulfasalazine** protein (3-letter code SAS), used to treat rheumatoid arthritis, ulcerative colitis, and Crohn's disease.

OpenBabel and ACPype packages are used to **add hydrogens, energetically minimize the structure**, and **generate parameters** for the **GROMACS** package. With *Generalized Amber Force Field (GAFF) forcefield and AM1-BCC* charges.

1.1.1 Settings

Biobb modules used

- biobb_io: Tools to fetch data to be consumed by the rest of the Biobb building blocks.
- biobb_chemistry: Tools to manipulate chemical data.

Auxiliar libraries used

- nb_conda_kernels: Enables a Jupyter Notebook or JupyterLab application in one conda environment to access kernels for Python, R, and other languages found in other environments.

- nglview: Jupyter/IPython widget to interactively view molecular structures and trajectories in notebooks.
- ipywidgets: Interactive HTML widgets for Jupyter notebooks and the IPython kernel.

Conda Installation and Launch

```
git clone https://github.com/bioexcel/biobb_wf_ligand_parameterization.git
cd biobb_wf_ligand_parameterization
conda env create -f conda_env/environment.yml
conda activate biobb_ligand_parameterization_tutorial
jupyter-nbextension enable --py --user widgetsnbextension
jupyter-nbextension enable --py --user nglview
jupyter-notebook biobb_wf_ligand_parameterization/notebooks/biobb_ligand_
↪parameterization_tutorial.ipynb
```

1.1.2 Tutorial

Click here to view tutorial in Read the Docs

1.1.3 Version

2020.4

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 Automatic Ligand parameterization tutorial using BioExcel Building Blocks (biobb)

This tutorial aims to illustrate the process of **ligand parameterization** for a **small molecule**, step by step, using the **BioExcel Building Blocks library (biobb)**. The particular example used is the **Ibuprofen** small compound (3-letter code IBP, Drugbank code DB01050), a non-steroidal **anti-inflammatory drug** (NSAID) derived from propionic acid and it is considered the first of the propionics.

OpenBabel and ACPype packages are used to **add hydrogens**, **energetically minimize the structure**, and **generate parameters** for the **GROMACS** package. With *Generalized Amber Force Field (GAFF)* forcefield and *AM1-BCC* charges.

Biobb modules used:

- biobb_io: Tools to fetch data to be consumed by the rest of the Biobb building blocks.
- biobb_chemistry: Tools to manipulate chemistry data.

Auxiliar libraries used:

- nb_conda_kernels: Enables a Jupyter Notebook or JupyterLab application in one conda environment to access kernels for Python, R, and other languages found in other environments.
- nglview: Jupyter/IPython widget to interactively view molecular structures and trajectories in notebooks.
- ipywidgets: Interactive HTML widgets for Jupyter notebooks and the IPython kernel.

1.2.1 Conda Installation and Launch

```
git clone https://github.com/bioexcel/biobb_wf_ligand_parameterization.git
cd biobb_wf_ligand_parameterization
conda env create -f conda_env/environment.yml
conda activate biobb_ligand_parameterization_tutorial
jupyter-nbextension enable --py --user widgetsnbextension
jupyter-nbextension enable --py --user nglview
jupyter-notebook biobb_wf_ligand_parameterization/notebooks/biobb_ligand_
↪parameterization_tutorial.ipynb
```

1.2.2 Pipeline steps:

1. *Input Parameters*
2. *Fetching Ligand Structure*
3. *Add Hydrogen Atoms*
4. *Energetically Minimize Hydrogen Atoms*
5. *Generating Ligand Parameters*
6. *Output Files*
7. *Questions & Comments*

1.2.3 Input parameters

Input parameters needed:

- **ligandCode**: 3-letter code of the ligand structure (e.g. IBP)
- **mol_charge**: Molecule net charge (e.g. -1)
- **pH**: Acidity or alkalinity for the small molecule. Hydrogen atoms will be added according to this pH. (e.g. 7.4)

```
import nglview
import ipywidgets
import os

ligandCode = 'IBP'
mol_charge = -1
pH = 7.4
```

1.2.4 Fetching ligand structure

Downloading **ligand structure** in **PDB format** from the IRB PDB MIRROR database. Alternatively, a **PDB file** can be used as starting structure.

Building Blocks used:

- Ligand from **biobb_io.api.ligand**

```
# Ligand: Download ligand structure from MMB PDB mirror REST API (http://mmb.
↪irbbarcelona.org/api/)
# Import module
from biobb_io.api.ligand import Ligand
from biobb_io.api.pdb import Pdb

# Create prop dict and inputs/outputs
input_structure = ligandCode + '.pdb'

prop = {
    'ligand_code' : ligandCode
}

#Create and launch bb
Ligand(output_pdb_path=input_structure,
        properties=prop).launch()
```

Visualizing 3D structure

Visualizing the downloaded/given **ligand PDB structure** using **NGL**:

```
#Show small ligand structure
view = nglview.show_structure_file(input_structure)
view.add_representation(repr_type='ball+stick', selection='all')
view._remote_call('setSize', target='Widget', args=['','300px'])
view.camera='orthographic'
view
```

1.2.5 Add Hydrogen Atoms

Adding **Hydrogen atoms** to the small molecule, according to the given pH.

Building Blocks used:

- BabelAddHydrogens from **biobb_chemistry.babelm.babel_add_hydrogens**

```
# Babel_add_hydrogens: add Hydrogen atoms to a small molecule
# Import module
from biobb_chemistry.babelm.babel_add_hydrogens import BabelAddHydrogens

# Create prop dict and inputs/outputs
output_babel_h = ligandCode + '.H.mol2'

prop = {
    'ph' : pH,
    'input_format' : 'pdb',
    'output_format' : 'mol2'
}

#Create and launch bb
BabelAddHydrogens(input_path=input_structure,
                  output_path=output_babel_h,
                  properties=prop).launch()
```

Visualizing 3D structure

Visualizing the **ligand PDB structure** with the newly added **hydrogen atoms** using **NGL**:

```
#Show small ligand structure
view = nglview.show_structure_file(output_babel_h)
view.add_representation(repr_type='ball+stick', selection='all')
view.camera='orthographic'
view
```

1.2.6 Energetically minimize Hydrogen Atoms

Energetically minimize newly added **Hydrogen atoms**.

Building Blocks used:

- BabelMinimize from **biobb_chemistry.babelm.babel_minimize**

```
# Babel_minimize: Structure energy minimization of a small molecule after being␣
↪modified adding hydrogen atoms
# Import module
from biobb_chemistry.babelm.babel_minimize import BabelMinimize

# Create prop dict and inputs/outputs
output_babel_min = ligandCode + '.H.min.pdb'
prop = {
    'method' : 'sd',
    'criteria' : '1e-10',
    'force_field' : 'GAFF'
}


#Create and launch bb
BabelMinimize(input_path=output_babel_h,
              output_path=output_babel_min,
              properties=prop).launch()
```

Visualizing 3D structure

Visualizing the **ligand PDB structure** with the newly added **hydrogen atoms**, **energetically minimized**, using **NGL**:

```
#Show small ligand structure
view = nglview.show_structure_file(output_babel_min)
view.add_representation(repr_type='ball+stick', selection='all')
view._remote_call('setSize', target='Widget', args=['','300px'])
view.camera='orthographic'
view
```

Visualizing 3D structures

Visualizing all the structures generated so far:

- Original **ligand PDB structure** (left)
- **Ligand PDB structure** with **hydrogen atoms** (middle)
- **Ligand PDB structure** with **hydrogen atoms energetically minimized** (right)

```
#Show different structures generated (for comparison)
view1 = nglview.show_structure_file(input_structure)
view1.add_representation(repr_type='ball+stick')
view1._remote_call('setSize', target='Widget', args=['250px','300px'])
view1.camera='orthographic'
```

(continues on next page)

(continued from previous page)

```
view1
view2 = nglview.show_structure_file(output_babel_h)
view2.add_representation(repr_type='ball+stick')
view2._remote_call('setSize', target='Widget', args=['250px','300px'])
view2.camera='orthographic'
view2
view3 = nglview.show_structure_file(output_babel_min)
view3.add_representation(repr_type='ball+stick')
view3._remote_call('setSize', target='Widget', args=['250px','300px'])
view3.camera='orthographic'
view3
ipywidgets.HBox([view1, view2, view3])
```

1.2.7 Generating ligand parameters

Building GROMACS topology corresponding to the **ligand structure**.

Force field used in this tutorial step is **amberGAFF**: General AMBER Force Field, designed for rational drug design.

Building Blocks used:

- AcpypeParamsGMX from **biobb_chemistry.acpype.acpype_params_gmx**

```
# Acpype_params_gmx: Generation of topologies for GROMACS with ACPype
# Import module
from biobb_chemistry.acpype.acpype_params_gmx import AcpypeParamsGMX

# Create prop dict and inputs/outputs
output_acpype_gro = ligandCode + 'params.gro'
output_acpype_itp = ligandCode + 'params.itp'
output_acpype_top = ligandCode + 'params.top'
output_acpype = ligandCode + 'params'
prop = {
    'basename' : output_acpype,
    'charge' : mol_charge
}

#Create and launch bb
AcpypeParamsGMX(input_path=output_babel_min,
                output_path_gro=output_acpype_gro,
                output_path_itp=output_acpype_itp,
                output_path_top=output_acpype_top,
                properties=prop).launch()
```

Visualizing 3D structure

Visualizing the generated **GROMACS** gro structure corresponding to the parameterized **ligand PDB structure** using **NGL**:

```
#Show small ligand structure
view = nglview.show_structure_file(output_acpype_gro)
view.add_representation(repr_type='ball+stick', selection='all')
view._remote_call('setSize', target='Widget', args=['','300px'])
view.camera='orthographic'
view
```

1.2.8 Output files

Important **Output files** generated:

- IBPparams.gro: **Structure** of the parameterized ligand in gro (GROMACS) format.
- IBPparams.top: **Topology** of the parameterized ligand, including a reference to the IBPparams.itp.
- IBPparams.itp: **Include Topology File (itp)** of the parameterized ligand, including the parameters information: bonds, angles, dihedrals, etc.

1.2.9 Questions & Comments

Questions, issues, suggestions and comments are really welcome!

- GitHub issues:
  - https://github.com/bioexcel/biobb
- BioExcel forum:
  - https://ask.bioexcel.eu/c/BioExcel-Building-Blocks-library

CHAPTER 2

Github repository.