biobb$_{m}dDocumentation$

Release 3.7.1

Bioexcel Project

Nov 18, 2021

Contents

CHAPTER 1

Contents

1.1 biobb_md

1.1.1 Introduction

Biobb_md is the Biobb module collection to perform molecular dynamics simulations. 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

v3.7.1 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_md>=3.7.1"
```

- Usage: Python API documentation

Using ANACONDA:

- Installation:

```
conda install -c bioconda "biobb_md>=3.7.1"
```

- 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_md:3.7.1--pyhdfd78af_0
```

- Usage:

```
docker run quay.io/biocontainers/biobb_md:3.7.1--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_md.sif shub://bioexcel/biobb_md
```

- Usage:

```
singularity exec biobb_md.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_md

1.2.1 gromacs package

Submodules

gromacs.editconf module

Module containing the Editconf class and the command line interface.

class gromacs.editconf.**Editconf**(*input_gro_path: str*, *output_gro_path: str*, *properties: dict = None*, ***kwargs*)

Bases: biobb_common.generic.biobb_object.BiobbObject

biobb_md Editconf

Wrapper class for the GROMACS editconf module.

The GROMACS solvate module generates a box around the selected structure.

Parameters

- **input_gro_path** (*str*) – Path to the input GRO file. File type: input. Sample file. Accepted formats: gro (edam:format_2033).
- **output_gro_path** (*str*) – Path to the output GRO file. File type: output. Sample file. Accepted formats: gro (edam:format_2033).
- **properties** (*dict - Python dictionary object containing the tool parameters, not input/output files*) –
  - **distance_to_molecule** (*float*) - (1.0) [0~100|0.1] Distance of the box from the outermost atom in nm. ie 1.0nm = 10 Angstroms.
  - **box_type** (*str*) - ("cubic") Geometrical shape of the solvent box. Values: cubic (rectangular box with all sides equal), triclinic (triclinic box), dodecahedron (rhombic dodecahedron), octahedron (truncated octahedron).
  - **center_molecule** (*bool*) - (True) Center molecule in the box.
  - **gmx_lib** (*str*) - (None) Path set GROMACS GMXLIB environment variable.
  - **gmx_path** (*str*) - ("gmx") Path to the GROMACS executable binary.
  - **remove_tmp** (*bool*) - (True) [WF property] Remove temporal files.
  - **restart** (*bool*) - (False) [WF property] Do not execute if output files exist.
  - **container_path** (*str*) - (None) Path to the binary executable of your container.
  - **container_image** (*str*) - ("gromacs/gromacs:latest") Container Image identifier.
  - **container_volume_path** (*str*) - ("/data") Path to an internal directory in the container.
  - **container_working_dir** (*str*) - (None) Path to the internal CWD in the container.
  - **container_user_id** (*str*) - (None) User number id to be mapped inside the container.
  - **container_shell_path** (*str*) - ("/bin/bash") Path to the binary executable of the container shell.

Examples

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

```
from biobb_md.gromacs.editconf import editconf
prop = { 'distance_to_molecule': 1.0,
         'box_type': 'cubic'}
editconf(input_gro_path='/path/to/structure.gro',
         output_gro_path='/path/to/newStructure.gro',
         properties=prop)
```

Info:

- **wrapped_software:**
  - name: GROMACS Solvate
  - version: >5.1
  - license: LGPL 2.1
- **ontology:**
  - name: EDAM
  - schema: http://edamontology.org/EDAM.owl

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

gromacs.editconf.**editconf**(*input_gro_path: str*, *output_gro_path: str*, *properties: dict = None*, ***kwargs*) → int
Create *Editconf* class and execute the *launch()* method.

gromacs.editconf.**main**()

gromacs.genion module

Module containing the Genion class and the command line interface.

class gromacs.genion.**Genion**(*input_tpr_path: str*, *output_gro_path: str*, *input_top_zip_path: str*, *output_top_zip_path: str*, *input_ndx_path: str = None*, *properties: dict = None*, ***kwargs*)
Bases: biobb_common.generic.biobb_object.BiobbObject

biobb_md Genion
Wrapper class for the GROMACS genion module.
The GROMACS genion module randomly replaces solvent molecules with monoatomic ions. The group of solvent molecules should be continuous and all molecules should have the same number of atoms.

Parameters

- **input_tpr_path** (*str*) – Path to the input portable run input TPR file. File type: input. Sample file. Accepted formats: tpr (edam:format_2333).
- **output_gro_path** (*str*) – Path to the input structure GRO file. File type: output. Sample file. Accepted formats: gro (edam:format_2033).

- **input_top_zip_path** (*str*) – Path the input TOP topology in zip format. File type: input. Sample file. Accepted formats: zip (edam:format_3987).
- **output_top_zip_path** (*str*) – Path the output topology TOP and ITP files zipball. File type: output. Sample file. Accepted formats: zip (edam:format_3987).
- **input_ndx_path** (*str*) *(Optional)* – Path to the input index NDX file. File type: input. Accepted formats: ndx (edam:format_2330).
- **properties** (*dict - Python dictionary object containing the tool parameters, not input/output files*) –
  - **replaced_group** (*str*) - ("SOL") Group of molecules that will be replaced by the solvent.
  - **neutral** (*bool*) - (False) Neutralize the charge of the system.
  - **concentration** (*float*) - (0.05) [0~10|0.01] Concentration of the ions in (mol/liter).
  - **seed** (*int*) - (1993) Seed for random number generator.
  - **gmx_lib** (*str*) - (None) Path set GROMACS GMXLIB environment variable.
  - **gmx_path** (*str*) - ("gmx") Path to the GROMACS executable binary.
  - **remove_tmp** (*bool*) - (True) [WF property] Remove temporal files.
  - **restart** (*bool*) - (False) [WF property] Do not execute if output files exist.
  - **container_path** (*str*) - (None) Path to the binary executable of your container.
  - **container_image** (*str*) - ("gromacs/gromacs:latest") Container Image identifier.
  - **container_volume_path** (*str*) - ("/data") Path to an internal directory in the container.
  - **container_working_dir** (*str*) - (None) Path to the internal CWD in the container.
  - **container_user_id** (*str*) - (None) User number id to be mapped inside the container.
  - **container_shell_path** (*str*) - ("/bin/bash") Path to the binary executable of the container shell.

Examples

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

```
from biobb_md.gromacs.genion import genion
prop = { 'concentration': 0.05,
         'replaced_group': 'SOL'}
genion(input_tpr_path='/path/to/myPortableBinaryRunInputFile.tpr',
       output_gro_path='/path/to/newStructure.gro',
       input_top_zip_path='/path/to/myTopology.zip',
       output_top_zip_path='/path/to/newTopology.zip',
       properties=prop)
```

Info:

- **wrapped_software:**
  - name: GROMACS Genion
  - version: >5.1
  - license: LGPL 2.1
- **ontology:**

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

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

gromacs.genion.**genion**(*input_tpr_path: str*, *output_gro_path: str*, *input_top_zip_path: str*, *output_top_zip_path: str*, *properties: dict = None*, ***kwargs*) → int
Create *Genion* class and execute the *launch()* method.

gromacs.genion.**main**()
Command line execution of this building block. Please check the command line documentation.

gromacs.genrestr module

Module containing the Genrestr class and the command line interface.

class gromacs.genrestr.**Genrestr**(*input_structure_path: str*, *output_itp_path: str*, *input_ndx_path: str = None*, *properties: dict = None*, ***kwargs*)
Bases: biobb_common.generic.biobb_object.BiobbObject

biobb_md Genrestr
Wrapper of the GROMACS genrestr module.
The GROMACS genrestr module, produces an #include file for a topology containing a list of atom numbers and three force constants for the x-, y-, and z-direction based on the contents of the -f file. A single isotropic force constant may be given on the command line instead of three components.

Parameters

- **input_structure_path** (*str*) – Path to the input structure PDB, GRO or TPR format. File type: input. Sample file. Accepted formats: pdb (edam:format_1476), gro (edam:format_2033), tpr (edam:format_2333).
- **output_itp_path** (*str*) – Path the output ITP topology file with restrains. File type: output. Sample file. Accepted formats: itp (edam:format_3883).
- **input_ndx_path** (*str) (Optional*) – Path to the input GROMACS index file, NDX format. File type: input. Sample file. Accepted formats: ndx (edam:format_2330).
- **properties** (*dict - Python dictionary object containing the tool parameters, not input/output files*) –
  - **restrained_group** (*str*) - ("system") Index group that will be restrained.
  - **force_constants** (*str*) - ("500 500 500") Array of three floats defining the force constants
  - **gmx_lib** (*str*) - (None) Path set GROMACS GMXLIB environment variable.
  - **gmx_path** (*str*) - ("gmx") Path to the GROMACS executable binary.
  - **remove_tmp** (*bool*) - (True) [WF property] Remove temporal files.
  - **restart** (*bool*) - (False) [WF property] Do not execute if output files exist.
  - **container_path** (*str*) - (None) Path to the binary executable of your container.
  - **container_image** (*str*) - ("gromacs/gromacs:latest") Container Image identifier.
  - **container_volume_path** (*str*) - ("/data") Path to an internal directory in the container.

- **container_working_dir** (*str*) - (None) Path to the internal CWD in the container.
- **container_user_id** (*str*) - (None) User number id to be mapped inside the container.
- **container_shell_path** (*str*) - ("/bin/bash") Path to the binary executable of the container shell.

Examples

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

```
from biobb_md.gromacs.genrestr import genrestr
prop = { 'restrained_group': 'system',
         'force_constants': '500 500 500' }
genrestr(input_structure_path='/path/to/myStructure.gro',
         output_itp_path='/path/to/newTopologyAddOn.itp',
         properties=prop)
```

Info:

- **wrapped_software:**
  - name: GROMACS Genrestr
  - version: >5.1
  - license: LGPL 2.1
- **ontology:**
  - name: EDAM
  - schema: http://edamontology.org/EDAM.owl

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

gromacs.genrestr.**genrestr**(*input_structure_path: str*, *output_itp_path: str*, *input_ndx_path: str = None*, *properties: dict = None*, ***kwargs*) → int
Create *Genrestr* class and execute the *launch()* method.

gromacs.genrestr.**main**()
Command line execution of this building block. Please check the command line documentation.

gromacs.grompp module

Module containing the Grompp class and the command line interface.

class gromacs.grompp.**Grompp**(*input_gro_path: str*, *input_top_zip_path: str*, *output_tpr_path: str*, *input_cpt_path: str = None*, *input_ndx_path: str = None*, *input_mdp_path: str = None*, *properties: dict = None*, ***kwargs*)
Bases: biobb_common.generic.biobb_object.BiobbObject

biobb_md Grompp
Wrapper of the GROMACS grompp module.

The GROMACS preprocessor module needs to be fed with the input system and the dynamics parameters to create a portable binary run input file TPR. The simulation parameters can be specified by two methods: 1.The predefined mdp settings defined at simulation_type property or 2.A custom mdp file defined at the input_mdp_path argument. These two methods are mutually exclusive. In both cases can be further modified by adding parameters to the mdp section in the yaml configuration file. The simulation parameter names and default values can be consulted in the official MDP specification.

Parameters

- **input_gro_path** (*str*) – Path to the input GROMACS structure GRO file. File type: input. Sample file. Accepted formats: gro (edam:format_2033).
- **input_top_zip_path** (*str*) – Path to the input GROMACS topology TOP and ITP files in zip format. File type: input. Sample file. Accepted formats: zip (edam:format_3987).
- **output_tpr_path** (*str*) – Path to the output portable binary run file TPR. File type: output. Sample file. Accepted formats: tpr (edam:format_2333).
- **input_cpt_path** (*str) (Optional*) – Path to the input GROMACS checkpoint file CPT. File type: input. Accepted formats: cpt (edam:format_2333).
- **input_ndx_path** (*str) (Optional*) – Path to the input GROMACS index files NDX. File type: input. Accepted formats: ndx (edam:format_2330).
- **input_mdp_path** (*str) (Optional*) – Path to the input GROMACS MDP file. File type: input. Accepted formats: mdp (edam:format_2330).
- **properties** (*dict - Python dictionary object containing the tool parameters, not input/output files*) –
  - **mdp** (*dict*) - ({}) MDP options specification.
  - **simulation_type** (*str*) - (None) Default options for the mdp file. Each one creates a different mdp file. Values: minimization (Energy minimization using steepest descent algorithm is used), nvt (substance N Volume V and Temperature T are conserved), npt (substance N pressure P and Temperature T are conserved), free (No design constraints applied; Free MD), ions (Synonym of minimization), index (Creates an empty mdp file).
  - **maxwarn** (*int*) - (0) [0~1000|1] Maximum number of allowed warnings. If simulation_type is index default is 10.
  - **gmx_lib** (*str*) - (None) Path set GROMACS GMXLIB environment variable.
  - **gmx_path** (*str*) - ("gmx") Path to the GROMACS executable binary.
  - **remove_tmp** (*bool*) - (True) [WF property] Remove temporal files.
  - **restart** (*bool*) - (False) [WF property] Do not execute if output files exist.
  - **container_path** (*str*) - (None) Path to the binary executable of your container.
  - **container_image** (*str*) - ("gromacs/gromacs:latest") Container Image identifier.
  - **container_volume_path** (*str*) - ("/data") Path to an internal directory in the container.
  - **container_working_dir** (*str*) - (None) Path to the internal CWD in the container.
  - **container_user_id** (*str*) - (None) User number id to be mapped inside the container.
  - **container_shell_path** (*str*) - ("/bin/bash") Path to the binary executable of the container shell.

Examples

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

```
from biobb_md.gromacs.grompp import grompp

prop = { 'simulation_type': 'minimization',
         'mdp':
            { 'emtol':'500',
              'nsteps':'5000'}}
grompp(input_gro_path='/path/to/myStructure.gro',
       input_top_zip_path='/path/to/myTopology.zip',
       output_tpr_path='/path/to/newCompiledBin.tpr',
       properties=prop)
```

Info:

- **wrapped_software:**
  - name: GROMACS Grompp
  - version: >5.1
  - license: LGPL 2.1
- **ontology:**
  - name: EDAM
  - schema: http://edamontology.org/EDAM.owl

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

gromacs.grompp.**grompp**(*input_gro_path: str*, *input_top_zip_path: str*, *output_tpr_path: str*, *input_cpt_path: str = None*, *input_ndx_path: str = None*, *input_mdp_path: str = None*, *properties: dict = None*, ***kwargs*) → int
Create *Grompp* class and execute the *launch()* method.

gromacs.grompp.**main**()
Command line execution of this building block. Please check the command line documentation.

gromacs.grompp_mdrun module

Module containing the GromppMDrun class and the command line interface.

class gromacs.grompp_mdrun.**GromppMdrun**(*input_gro_path: str*, *input_top_zip_path: str*, *output_trr_path: str*, *output_gro_path: str*, *output_edr_path: str*, *output_log_path: str*, *input_cpt_path: str = None*, *input_ndx_path: str = None*, *input_mdp_path: str = None*, *output_xtc_path: str = None*, *output_cpt_path: str = None*, *output_dhdl_path: str = None*, *properties: dict = None*, ***kwargs*)
Bases: biobb_common.generic.biobb_object.BiobbObject

biobb_md GromppMdrun
Wrapper of the GROMACS grompp module and the GROMACS mdrun module.

Grompp The GROMACS preprocessor module needs to be fed with the input system and the dynamics parameters to create a portable binary run input file TPR. The simulation parameters can be specified by two methods: 1.The predefined mdp settings defined at simulation_type property or 2.A custom mdp file defined at the input_mdp_path argument. These two methods are mutually exclusive. In both cases can be further modified by adding parameters to the mdp section in the yaml configuration file. The simulation parameter names and default values can be consulted in the official MDP specification. MDRun is the main computational chemistry engine within GROMACS. It performs Molecular Dynamics simulations, but it can also perform Stochastic Dynamics, Energy Minimization, test particle insertion or (re)calculation of energies.

Parameters

- **input_gro_path** (*str*) – Path to the input GROMACS structure GRO file. File type: input. Sample file. Accepted formats: gro (edam:format_2033).
- **input_top_zip_path** (*str*) – Path to the input GROMACS topology TOP and ITP files in zip format. File type: input. Sample file. Accepted formats: zip (edam:format_3987).
- **output_trr_path** (*str*) – Path to the GROMACS uncompressed raw trajectory file TRR. File type: output. Sample file. Accepted formats: trr (edam:format_3910).
- **output_gro_path** (*str*) – Path to the output GROMACS structure GRO file. File type: output. Sample file. Accepted formats: gro (edam:format_2033).
- **output_edr_path** (*str*) – Path to the output GROMACS portable energy file EDR. File type: output. Sample file. Accepted formats: edr (edam:format_2330).
- **output_log_path** (*str*) – Path to the output GROMACS trajectory log file LOG. File type: output. Sample file. Accepted formats: log (edam:format_2330).
- **input_cpt_path** (*str) (Optional*) – Path to the input GROMACS checkpoint file CPT. File type: input. Accepted formats: cpt (edam:format_2333).
- **input_ndx_path** (*str) (Optional*) – Path to the input GROMACS index files NDX. File type: input. Accepted formats: ndx (edam:format_2330).
- **input_mdp_path** (*str) (Optional*) – Path to the input GROMACS MDP file. File type: input. Accepted formats: mdp (edam:format_2330).
- **output_xtc_path** (*str) (Optional*) – Path to the GROMACS compressed trajectory file XTC. File type: output. Accepted formats: xtc (edam:format_3875).
- **output_cpt_path** (*str) (Optional*) – Path to the output GROMACS checkpoint file CPT. File type: output. Accepted formats: cpt (edam:format_2333).
- **output_dhdl_path** (*str) (Optional*) – Path to the output dhdl.xvg file only used when free energy calculation is turned on. File type: output. Accepted formats: xvg (edam:format_2033).
- **properties** (*dict - Python dictionary object containing the tool parameters, not input/output files*) –
  - **mdp** (*dict*) - ({}) MDP options specification.
  - **simulation_type** (*str*) - ("minimization") Default options for the mdp file. Each creates a different mdp file. Values: minimization (Energy minimization using steepest descent algorithm is used), nvt (substance N Volume V and Temperature T are conserved), npt (substance N pressure P and Temperature T are conserved), free (No design constraints applied; Free MD), ions (Synonym of minimization), index (Creates an empty mdp file).
  - **maxwarn** (*int*) - (10) [0~1000|1] Maximum number of allowed warnings.

- **mpi_bin** (*str*) - (None) Path to the MPI runner. Usually "mpirun" or "srun".
- **mpi_np** (*str*) - (None) Number of MPI processes. Usually an integer bigger than 1.
- **mpi_hostlist** (*str*) - (None) Path to the MPI hostlist file.
- **checkpoint_time** (*int*) - (15) [0~1000|1] Checkpoint writing interval in minutes. Only enabled if an output_cpt_path is provided.
- **num_threads** (*int*) - (0) [0-1000|1] Let GROMACS guess. The number of threads that are going to be used.
- **num_threads_mpi** (*int*) - (0) [0-1000|1] Let GROMACS guess. The number of GROMACS MPI threads that are going to be used.
- **num_threads_omp** (*int*) - (0) [0-1000|1] Let GROMACS guess. The number of GROMACS OPENMP threads that are going to be used.
- **num_threads_omp_pme** (*int*) - (0) [0-1000|1] Let GROMACS guess. The number of GROMACS OPENMP_PME threads that are going to be used.
- **use_gpu** (*bool*) - (False) Use settings appropriate for GPU. Adds: -nb gpu -pme gpu
- **gpu_id** (*str*) - (None) List of unique GPU device IDs available to use.
- **gpu_tasks** (*str*) - (None) List of GPU device IDs, mapping each PP task on each node to a device.
- **gmx_lib** (*str*) - (None) Path set GROMACS GMXLIB environment variable.
- **gmx_path** (*str*) - ("gmx") Path to the GROMACS executable binary.
- **remove_tmp** (*bool*) - (True) [WF property] Remove temporal files.
- **restart** (*bool*) - (False) [WF property] Do not execute if output files exist.
- **container_path** (*str*) - (None) Path to the binary executable of your container.
- **container_image** (*str*) - ("gromacs/gromacs:latest") Container Image identifier.
- **container_volume_path** (*str*) - ("/data") Path to an internal directory in the container.
- **container_working_dir** (*str*) - (None) Path to the internal CWD in the container.
- **container_user_id** (*str*) - (None) User number id to be mapped inside the container.
- **container_shell_path** (*str*) - ("/bin/bash") Path to the binary executable of the container shell.

Examples

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

```
from biobb_md.gromacs.grompp_mdrun import grompp_mdrun
prop = { 'num_threads': 0,
         'gmx_path': 'gmx',
          'mdp':
            { 'simulation_type': 'minimization',
              'emtol':'500',
              'nsteps':'5000'}
        }
grompp_mdrun(input_gro_path='/path/to/myStructure.gro',
             input_top_zip_path='/path/to/myTopology.zip',
```

(continues on next page)

(continued from previous page)

```
            output_trr_path='/path/to/newTrajectory.trr',
            output_gro_path='/path/to/newStructure.gro',
            output_edr_path='/path/to/newEnergy.edr',
            output_log_path='/path/to/newSimulationLog.log',
            properties=prop)
```

Info:

- **wrapped_software:**
  - name: GROMACS Grompp & MDRun
  - version: >5.1
  - license: LGPL 2.1
- **ontology:**
  - name: EDAM
  - schema: http://edamontology.org/EDAM.owl

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

gromacs.grompp_mdrun.**grompp_mdrun**(*input_gro_path: str*, *input_top_zip_path: str*, *output_trr_path: str*, *output_gro_path: str*, *output_edr_path: str*, *output_log_path: str*, *input_cpt_path: str = None*, *input_ndx_path: str = None*, *input_mdp_path: str = None*, *output_xtc_path: str = None*, *output_cpt_path: str = None*, *output_dhdl_path: str = None*, *properties: dict = None*, ***kwargs*) → int

gromacs.grompp_mdrun.**main**()
Command line execution of this building block. Please check the command line documentation.

gromacs.make_ndx module

Module containing the MakeNdx class and the command line interface.

class gromacs.make_ndx.**MakeNdx**(*input_structure_path: str*, *output_ndx_path: str*, *input_ndx_path: str = None*, *properties: dict = None*, ***kwargs*)
Bases: biobb_common.generic.biobb_object.BiobbObject

biobb_md MakeNdx
Wrapper of the GROMACS make_ndx module.
The GROMACS make_ndx module, generates an index file using the atoms of the selection.

Parameters

- **input_structure_path** (*str*) – Path to the input GRO/PDB/TPR file. File type: input. Sample file. Accepted formats: gro (edam:format_2033), pdb (edam:format_1476), tpr (edam:format_2333).
- **output_ndx_path** (*str*) – Path to the output index NDX file. File type: output. Sample file. Accepted formats: ndx (edam:format_2330).

- **input_ndx_path** (*str*) *(Optional)* – Path to the input index NDX file. File type: input. Accepted formats: ndx (edam:format_2330).
- **properties** (*dict - Python dictionary object containing the tool parameters, not input/output files*) –
  - **selection** (*str*) - ("a CA C N O") Heavy atoms. Atom selection string.
  - **gmx_lib** (*str*) - (None) Path set GROMACS GMXLIB environment variable.
  - **gmx_path** (*str*) - ("gmx") Path to the GROMACS executable binary.
  - **remove_tmp** (*bool*) - (True) [WF property] Remove temporal files.
  - **restart** (*bool*) - (False) [WF property] Do not execute if output files exist.
  - **container_path** (*str*) - (None) Path to the binary executable of your container.
  - **container_image** (*str*) - ("gromacs/gromacs:latest") Container Image identifier.
  - **container_volume_path** (*str*) - ("/data") Path to an internal directory in the container.
  - **container_working_dir** (*str*) - (None) Path to the internal CWD in the container.
  - **container_user_id** (*str*) - (None) User number id to be mapped inside the container.
  - **container_shell_path** (*str*) - ("/bin/bash") Path to the binary executable of the container shell.

Examples

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

```
from biobb_md.gromacs.make_ndx import make_ndx
prop = { 'selection': 'a CA C N O' }
make_ndx(input_structure_path='/path/to/myStructure.gro',
         output_ndx_path='/path/to/newIndex.ndx',
         properties=prop)
```

Info:

- **wrapped_software:**
  - name: GROMACS MakeNdx
  - version: >5.1
  - license: LGPL 2.1
- **ontology:**
  - name: EDAM
  - schema: http://edamontology.org/EDAM.owl

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

gromacs.make_ndx.**main**()
Command line execution of this building block. Please check the command line documentation.

gromacs.make_ndx.**make_ndx**(*input_structure_path: str*, *output_ndx_path: str*, *input_ndx_path: str = None*, *properties: dict = None*, ***kwargs*) → int
Create *MakeNdx* class and execute the *launch()* method.

gromacs.gmxselect module

Module containing the Select class and the command line interface.

class gromacs.gmxselect.**Gmxselect**(*input_structure_path: str*, *output_ndx_path: str*, *input_ndx_path: str = None*, *properties: dict = None*, ***kwargs*)

Bases: biobb_common.generic.biobb_object.BiobbObject

biobb_md Gmxselect

Wrapper of the GROMACS select module.

The GROMACS select module writes out basic data about dynamic selections. It can be used for some simple analyses, or the output can be combined with output from other programs and/or external analysis programs to calculate more complex things.

Parameters

- **input_structure_path** (*str*) – Path to the input GRO/PDB/TPR file. File type: input. Sample file. Accepted formats: pdb (edam:format_1476), gro (edam:format_2033), tpr (edam:format_2333).
- **output_ndx_path** (*str*) – Path to the output index NDX file. File type: output. Sample file. Accepted formats: ndx (edam:format_2330).
- **input_ndx_path** (*str) (Optional*) – Path to the input index NDX file. File type: input. Accepted formats: ndx (edam:format_2330).
- **properties** (*dict - Python dictionary object containing the tool parameters, not input/output files*) –
  - **selection** (*str*) - ("a CA C N O") Heavy atoms. Atom selection string.
  - **append** (*bool*) - (False) Append the content of the input_ndx_path to the output_ndx_path.
  - **gmx_path** (*str*) - ("gmx") Path to the GROMACS executable binary.
  - **gmx_lib** (*str*) - (None) Path set GROMACS GMXLIB environment variable.
  - **remove_tmp** (*bool*) - (True) [WF property] Remove temporal files.
  - **restart** (*bool*) - (False) [WF property] Do not execute if output files exist.
  - **container_path** (*str*) - (None) Path to the binary executable of your container.
  - **container_image** (*str*) - ("gromacs/gromacs:latest") Container Image identifier.
  - **container_volume_path** (*str*) - ("/data") Path to an internal directory in the container.
  - **container_working_dir** (*str*) - (None) Path to the internal CWD in the container.
  - **container_user_id** (*str*) - (None) User number id to be mapped inside the container.
  - **container_shell_path** (*str*) - ("/bin/bash") Path to the binary executable of the container shell.

Examples

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

```
from biobb_md.gromacs.gmxselect import gmxselect
prop = { 'selection': '"Mynewgroup" group "Protein-H" and not same residue as
↪within 0.4 of resname ARG' }
gmxselect(input_structure_path='/path/to/myStructure.gro',
          output_ndx_path='/path/to/newIndex.ndx',
          properties=prop)
```

Info:

- **wrapped_software:**
  - name: GROMACS Gmxselect
  - version: >5.1
  - license: LGPL 2.1
- **ontology:**
  - name: EDAM
  - schema: http://edamontology.org/EDAM.owl

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

gromacs.gmxselect.**gmxselect**(*input_structure_path: str*, *output_ndx_path: str*, *input_ndx_path: str = None*, *properties: dict = None*, ***kwargs*) → int
Create *Gmxselect* class and execute the *launch()* method.

gromacs.gmxselect.**main**()
Command line execution of this building block. Please check the command line documentation.

gromacs.mdrun module

Module containing the MDrun class and the command line interface.

class gromacs.mdrun.**Mdrun**(*input_tpr_path: str*, *output_trr_path: str*, *output_gro_path: str*, *output_edr_path: str*, *output_log_path: str*, *input_cpt_path: str = None*, *output_xtc_path: str = None*, *output_cpt_path: str = None*, *output_dhdl_path: str = None*, *properties: dict = None*, ***kwargs*)
Bases: biobb_common.generic.biobb_object.BiobbObject

biobb_md Mdrun
Wrapper of the GROMACS mdrun module.
MDRun is the main computational chemistry engine within GROMACS. It performs Molecular Dynamics simulations, but it can also perform Stochastic Dynamics, Energy Minimization, test particle insertion or (re)calculation of energies.

Parameters

- **input_tpr_path** (*str*) – Path to the portable binary run input file TPR. File type: input. Sample file. Accepted formats: tpr (edam:format_2333).
- **output_trr_path** (*str*) – Path to the GROMACS uncompressed raw trajectory file TRR. File type: output. Sample file. Accepted formats: trr (edam:format_3910).

- **output_gro_path** (*str*) – Path to the output GROMACS structure GRO file. File type: output. Sample file. Accepted formats: gro (edam:format_2033).
- **output_edr_path** (*str*) – Path to the output GROMACS portable energy file EDR. File type: output. Sample file. Accepted formats: edr (edam:format_2330).
- **output_log_path** (*str*) – Path to the output GROMACS trajectory log file LOG. File type: output. Sample file. Accepted formats: log (edam:format_2330).
- **input_cpt_path** (*str) (Optional*) – Path to the input GROMACS checkpoint file CPT. File type: input. Accepted formats: cpt (edam:format_2333).
- **output_xtc_path** (*str) (Optional*) – Path to the GROMACS compressed trajectory file XTC. File type: output. Accepted formats: xtc (edam:format_3875).
- **output_cpt_path** (*str) (Optional*) – Path to the output GROMACS checkpoint file CPT. File type: output. Accepted formats: cpt (edam:format_2333).
- **output_dhdl_path** (*str) (Optional*) – Path to the output dhdl.xvg file only used when free energy calculation is turned on. File type: output. Accepted formats: xvg (edam:format_2033).
- **properties** (*dict - Python dictionary object containing the tool parameters, not input/output files*) –
  - **mpi_bin** (*str*) - (None) Path to the MPI runner. Usually "mpirun" or "srun".
  - **mpi_np** (*int*) - (0) [0~1000|1] Number of MPI processes. Usually an integer bigger than 1.
  - **mpi_flags** (*str*) - (None) Path to the MPI hostlist file.
  - **checkpoint_time** (*int*) - (15) [0~1000|1] Checkpoint writing interval in minutes. Only enabled if an output_cpt_path is provided.
  - **num_threads** (*int*) - (0) [0~1000|1] Let GROMACS guess. The number of threads that are going to be used.
  - **num_threads_mpi** (*int*) - (0) [0~1000|1] Let GROMACS guess. The number of GROMACS MPI threads that are going to be used.
  - **num_threads_omp** (*int*) - (0) [0~1000|1] Let GROMACS guess. The number of GROMACS OPENMP threads that are going to be used.
  - **num_threads_omp_pme** (*int*) - (0) [0~1000|1] Let GROMACS guess. The number of GROMACS OPENMP_PME threads that are going to be used.
  - **use_gpu** (*bool*) - (False) Use settings appropriate for GPU. Adds: -nb gpu -pme gpu
  - **gpu_id** (*str*) - (None) List of unique GPU device IDs available to use.
  - **gpu_tasks** (*str*) - (None) List of GPU device IDs, mapping each PP task on each node to a device.
  - **gmx_lib** (*str*) - (None) Path set GROMACS GMXLIB environment variable.
  - **gmx_path** (*str*) - ("gmx") Path to the GROMACS executable binary.
  - **remove_tmp** (*bool*) - (True) [WF property] Remove temporal files.
  - **restart** (*bool*) - (False) [WF property] Do not execute if output files exist.
  - **container_path** (*str*) - (None) Path to the binary executable of your container.
  - **container_image** (*str*) - ("gromacs/gromacs:latest") Container Image identifier.

- **container_volume_path** (*str*) - ("/data") Path to an internal directory in the container.
- **container_working_dir** (*str*) - (None) Path to the internal CWD in the container.
- **container_user_id** (*str*) - (None) User number id to be mapped inside the container.
- **container_shell_path** (*str*) - ("/bin/bash") Path to the binary executable of the container shell.

Examples

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

```
from biobb_md.gromacs.mdrun import mdrun
prop = { 'num_threads': 0,
         'gmx_path': 'gmx' }
mdrun(input_tpr_path='/path/to/myPortableBinaryRunInputFile.tpr',
      output_trr_path='/path/to/newTrajectory.trr',
      output_gro_path='/path/to/newStructure.gro',
      output_edr_path='/path/to/newEnergy.edr',
      output_log_path='/path/to/newSimulationLog.log',
      properties=prop)
```

Info:

- **wrapped_software:**
  - name: GROMACS Mdrun
  - version: >5.1
  - license: LGPL 2.1
  - multinode: mpi
- **ontology:**
  - name: EDAM
  - schema: http://edamontology.org/EDAM.owl

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

gromacs.mdrun.**main**()
Command line execution of this building block. Please check the command line documentation.

gromacs.mdrun.**mdrun**(*input_tpr_path: str*, *output_trr_path: str*, *output_gro_path: str*, *output_edr_path: str*, *output_log_path: str*, *input_cpt_path: str = None*, *output_xtc_path: str = None*, *output_cpt_path: str = None*, *output_dhdl_path: str = None*, *properties: dict = None*, ***kwargs*) → int
Create *Mdrun* class and execute the *launch()* method.

gromacs.pdb2gmx module

Module containing the Pdb2gmx class and the command line interface.

class gromacs.pdb2gmx.**Pdb2gmx**(*input_pdb_path: str*, *output_gro_path: str*, *output_top_zip_path: str*, *properties: dict = None*, ***kwargs*)
Bases: biobb_common.generic.biobb_object.BiobbObject

biobb_md Pdb2gmx

Wrapper class for the GROMACS pdb2gmx module.

The GROMACS pdb2gmx module, reads a .pdb (or .gro) file, reads some database files, adds hydrogens to the molecules and generates coordinates in GROMACS (GROMOS), or optionally .pdb, format and a topology in GROMACS format. These files can subsequently be processed to generate a run input file.

Parameters

- **input_pdb_path** (*str*) – Path to the input PDB file. File type: input. Sample file. Accepted formats: pdb (edam:format_1476).
- **output_gro_path** (*str*) – Path to the output GRO file. File type: output. Sample file. Accepted formats: gro (edam:format_2033).
- **output_top_zip_path** (*str*) – Path the output TOP topology in zip format. File type: output. Sample file. Accepted formats: zip (edam:format_3987).
- **properties** (*dict - Python dictionary object containing the tool parameters, not input/output files*) –
  - **water_type** (*str*) - ("spce") Water molecule type. Values: spc, spce, tip3p, tip4p, tip5p, tips3p.
  - **force_field** (*str*) - ("amber99sb-ildn") Force field to be used during the conversion. Values: gromos45a3, charmm27, gromos53a6, amber96, amber99, gromos43a2, gromos54a7, gromos43a1, amberGS, gromos53a5, amber99sb, amber03, amber99sb-ildn, oplsaa, amber94.
  - **ignh** (*bool*) - (False) Should pdb2gmx ignore the hidrogens in the original structure.
  - **his** (*str*) - (None) Histidine protonation array.
  - **merge** (*bool*) - (False) Merge all chains into a single molecule.
  - **gmx_lib** (*str*) - (None) Path set GROMACS GMXLIB environment variable.
  - **gmx_path** (*str*) - ("gmx") Path to the GROMACS executable binary.
  - **remove_tmp** (*bool*) - (True) [WF property] Remove temporal files.
  - **restart** (*bool*) - (False) [WF property] Do not execute if output files exist.
  - **container_path** (*str*) - (None) Path to the binary executable of your container.
  - **container_image** (*str*) - ("gromacs/gromacs:latest") Container Image identifier.
  - **container_volume_path** (*str*) - ("/data") Path to an internal directory in the container.
  - **container_working_dir** (*str*) - (None) Path to the internal CWD in the container.
  - **container_user_id** (*str*) - (None) User number id to be mapped inside the container.
  - **container_shell_path** (*str*) - ("/bin/bash") Path to the binary executable of the container shell.

Examples

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

```
from biobb_md.gromacs.pdb2gmx import pdb2gmx
prop = { 'his': '0 0 1 1 0 0 0' }
pdb2gmx(input_pdb_path='/path/to/myStructure.pdb',
        output_gro_path='/path/to/newStructure.gro',
        output_top_zip_path='/path/to/newTopology.zip',
        properties=prop)
```

Info:

- **wrapped_software:**
  - name: GROMACS Pdb2gmx
  - version: >5.1
  - license: LGPL 2.1
- **ontology:**
  - name: EDAM
  - schema: http://edamontology.org/EDAM.owl

launch() → int
Execute the *Pdb2gmx* object.

gromacs.pdb2gmx.**main**()
Command line execution of this building block. Please check the command line documentation.

gromacs.pdb2gmx.**pdb2gmx**(*input_pdb_path: str*, *output_gro_path: str*, *output_top_zip_path: str*, *properties: dict = None*, ***kwargs*) → int
Create *Pdb2gmx* class and execute the *launch()* method.

gromacs.solvate module

Module containing the Editconf class and the command line interface.

class gromacs.solvate.**Solvate**(*input_solute_gro_path: str*, *output_gro_path: str*, *input_top_zip_path: str*, *output_top_zip_path: str*, *input_solvent_gro_path: str = None*, *properties: dict = None*, ***kwargs*)
Bases: biobb_common.generic.biobb_object.BiobbObject

biobb_md Solvate
Wrapper of the GROMACS solvate module.
The GROMACS solvate module, generates a box of solvent around the selected structure.

Parameters

- **input_solute_gro_path** (*str*) – Path to the input GRO file. File type: input. Sample file. Accepted formats: gro (edam:format_2033).
- **output_gro_path** (*str*) – Path to the output GRO file. File type: output. Sample file. Accepted formats: gro (edam:format_2033).
- **input_top_zip_path** (*str*) – Path the input TOP topology in zip format. File type: input. Sample file. Accepted formats: zip (edam:format_3987).

- **output_top_zip_path** (*str*) – Path the output topology in zip format. File type: output. Sample file. Accepted formats: zip (edam:format_3987).
- **input_solvent_gro_path** (*str*) *(Optional)* – (spc216.gro) Path to the GRO file containing the structure of the solvent. File type: input. Accepted formats: gro (edam:format_2033).
- **properties** (*dict - Python dictionary object containing the tool parameters, not input/output files*) –
  - **shell** (*float*) - (0.0) [0~100|0.1] Thickness in nanometers of optional water layer around solute.
  - **gmx_lib** (*str*) - (None) Path set GROMACS GMXLIB environment variable.
  - **gmx_path** (*str*) - ("gmx") Path to the GROMACS executable binary.
  - **remove_tmp** (*bool*) - (True) [WF property] Remove temporal files.
  - **restart** (*bool*) - (False) [WF property] Do not execute if output files exist.
  - **container_path** (*str*) - (None) Path to the binary executable of your container.
  - **container_image** (*str*) - ("gromacs/gromacs:latest") Container Image identifier.
  - **container_volume_path** (*str*) - ("/data") Path to an internal directory in the container.
  - **container_working_dir** (*str*) - (None) Path to the internal CWD in the container.
  - **container_user_id** (*str*) - (None) User number id to be mapped inside the container.
  - **container_shell_path** (*str*) - ("/bin/bash") Path to the binary executable of the container shell.

Examples

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

```
from biobb_md.gromacs.solvate import Solvate
prop = { 'shell': 1.0 }
solvate(input_solute_gro_path='/path/to/myStructure.gro',
        output_gro_path='/path/to/newStructure.gro',
        input_top_zip_path='/path/to/myTopology.zip',
        output_top_zip_path='/path/to/newTopology.zip',
        properties=prop)
```

Info:

- **wrapped_software:**
  - name: GROMACS Solvate
  - version: >5.1
  - license: LGPL 2.1
- **ontology:**
  - name: EDAM
  - schema: http://edamontology.org/EDAM.owl

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

gromacs.solvate.**main**()
Command line execution of this building block. Please check the command line documentation.

gromacs.solvate.**solvate**(*input_solute_gro_path: str*, *output_gro_path: str*, *input_top_zip_path: str*, *output_top_zip_path: str*, *input_solvent_gro_path: str = None*, *properties: dict = None*, ***kwargs*) → int
Create *Solvate* class and execute the *launch()* method.

1.2.2 gromacs_extra package

Submodules

gromacs_extra.ndx2resttop module

Module containing the Ndx2resttop class and the command line interface.

class gromacs_extra.ndx2resttop.**Ndx2resttop**(*input_ndx_path: str*, *input_top_zip_path: str*, *output_top_zip_path: str*, *properties: dict = None*, ***kwargs*)
Bases: biobb_common.generic.biobb_object.BiobbObject

biobb_md Ndx2resttop
Generate a restrained topology from an index NDX file.
This module automatizes the process of restrained topology generation starting from an index NDX file.

Parameters

- **input_ndx_path** (*str*) – Path to the input NDX index file. File type: input. Sample file. Accepted formats: ndx (edam:format_2330).
- **input_top_zip_path** (*str*) – Path the input TOP topology in zip format. File type: input. Sample file. Accepted formats: zip (edam:format_3987).
- **output_top_zip_path** (*str*) – Path the output TOP topology in zip format. File type: output. Sample file. Accepted formats: zip (edam:format_3987).
- **properties** (*dict - Python dictionary object containing the tool parameters, not input/output files*) –
  - **force_constants** (*str*) - ("500 500 500") Array of three floats defining the force constants.
  - **ref_rest_chain_triplet_list** (*str*) - (None) Triplet list composed by (reference group, restrain group, chain) list.

Examples

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

```
from biobb_md.gromacs_extra.ndx2resttop import ndx2resttop
prop = { 'ref_rest_chain_triplet_list': '( Chain_A, Chain_A_noMut, A ), ( Chain_B,
↪ Chain_B_noMut, B ), ( Chain_C, Chain_C_noMut, C ), ( Chain_D, Chain_D_noMut, D␣
↪)' }
ndx2resttop(input_ndx_path='/path/to/myIndex.ndx',
            input_top_zip_path='/path/to/myTopology.zip',
```

(continues on next page)

(continued from previous page)

```
            output_top_zip_path='/path/to/newTopology.zip',
            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 *Ndx2resttop* object.

gromacs_extra.ndx2resttop.**main**()
Command line execution of this building block. Please check the command line documentation.

gromacs_extra.ndx2resttop.**ndx2resttop**(*input_ndx_path: str*, *input_top_zip_path: str*, *output_top_zip_path: str*, *properties: dict = None*, ***kwargs*) → int
Create *Ndx2resttop* class and execute the *launch()* method.

gromacs_extra.append_ligand module

Module containing the AppendLigand class and the command line interface.

class gromacs_extra.append_ligand.**AppendLigand**(*input_top_zip_path: str*, *input_itp_path: str*, *output_top_zip_path: str*, *input_posres_itp_path: str = None*, *properties: dict = None*, ***kwargs*)
Bases: biobb_common.generic.biobb_object.BiobbObject

biobb_md AppendLigand
This class takes a ligand ITP file and inserts it in a topology.
This module automatizes the process of inserting a ligand ITP file in a GROMACS topology.

Parameters

- **input_top_zip_path** (*str*) – Path the input topology TOP and ITP files zipball. File type: input. Sample file. Accepted formats: zip (edam:format_3987).
- **input_itp_path** (*str*) – Path to the ligand ITP file to be inserted in the topology. File type: input. Sample file. Accepted formats: itp (edam:format_3883).
- **output_top_zip_path** (*str*) – Path/Name the output topology TOP and ITP files zipball. File type: output. Sample file. Accepted formats: zip (edam:format_3987).
- **input_posres_itp_path** (*str) (Optional*) – Path to the position restriction ITP file. File type: input. Accepted formats: itp (edam:format_3883).
- **properties** (*dic*) –

- **posres_name** (*str*) - ("POSRES_LIGAND") String to be included in the ifdef clause.
- **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_md.gromacs_extra.append_ligand import append_ligand
prop = { 'posres_name': 'POSRES_LIGAND' }
append_ligand(input_top_zip_path='/path/to/myTopology.zip',
              input_itp_path='/path/to/myTopologyAddOn.itp',
              output_top_zip_path='/path/to/newTopology.zip',
              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 *AppendLigand* object.

gromacs_extra.append_ligand.**append_ligand**(*input_top_zip_path: str*, *input_itp_path: str*, *output_top_zip_path: str*, *input_posres_itp_path: str = None*, *properties: dict = None*, ***kwargs*) → int
Create *AppendLigand* class and execute the *launch()* method.

gromacs_extra.append_ligand.**main**()
Command line execution of this building block. Please check the command line documentation.

1.3 BioBB MD Command Line Help

Generic usage:

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

1.3.1 Mdrun

Wrapper of the GROMACS mdrun module.

Get help

Command:

```
mdrun -h
```

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

I / O Arguments

Syntax: input_argument (datatype) : Definition

Config input / output arguments for this building block:

- **input_tpr_path** (*string*): Path to the portable binary run input file TPR. File type: input. Sample file. Accepted formats: TPR
- **output_trr_path** (*string*): Path to the GROMACS uncompressed raw trajectory file TRR. File type: output. Sample file. Accepted formats: TRR
- **output_gro_path** (*string*): Path to the output GROMACS structure GRO file. File type: output. Sample file. Accepted formats: GRO
- **output_edr_path** (*string*): Path to the output GROMACS portable energy file EDR. File type: output. Sample file. Accepted formats: EDR
- **output_log_path** (*string*): Path to the output GROMACS trajectory log file LOG. File type: output. Sample file. Accepted formats: LOG
- **input_cpt_path** (*string*): Path to the input GROMACS checkpoint file CPT. File type: input. Sample file. Accepted formats: CPT
- **output_xtc_path** (*string*): Path to the GROMACS compressed trajectory file XTC. File type: output. Sample file. Accepted formats: XTC
- **output_cpt_path** (*string*): Path to the output GROMACS checkpoint file CPT. File type: output. Sample file. Accepted formats: CPT
- **output_dhdl_path** (*string*): Path to the output dhdl.xvg file only used when free energy calculation is turned on. File type: output. Sample file. Accepted formats: XVG

Config

Syntax: input_parameter (datatype) - (default_value) Definition

Config parameters for this building block:

- **mpi_bin** (*string*): (None) Path to the MPI runner. Usually “mpirun” or “srun”..
- **mpi_np** (*integer*): (0) Number of MPI processes. Usually an integer bigger than 1..
- **mpi_flags** (*string*): (None) Path to the MPI hostlist file..
- **checkpoint_time** (*integer*): (15) Checkpoint writing interval in minutes. Only enabled if an output_cpt_path is provided..
- **num_threads** (*integer*): (0) Let GROMACS guess. The number of threads that are going to be used..
- **num_threads_mpi** (*integer*): (0) Let GROMACS guess. The number of GROMACS MPI threads that are going to be used..

- **num_threads_omp** (*integer*): (0) Let GROMACS guess. The number of GROMACS OPENMP threads that are going to be used..
- **num_threads_omp_pme** (*integer*): (0) Let GROMACS guess. The number of GROMACS OPENMP_PME threads that are going to be used..
- **use_gpu** (*boolean*): (False) Use settings appropriate for GPU. Adds: -nb gpu -pme gpu.
- **gpu_id** (*string*): (None) List of unique GPU device IDs available to use..
- **gpu_tasks** (*string*): (None) List of GPU device IDs, mapping each PP task on each node to a device..
- **gmx_lib** (*string*): (None) Path set GROMACS GMXLIB environment variable..
- **gmx_path** (*string*): (gmx) Path to the GROMACS executable binary..
- **remove_tmp** (*boolean*): (True) Remove temporal files..
- **restart** (*boolean*): (False) Do not execute if output files exist..
- **container_path** (*string*): (None) Path to the binary executable of your container..
- **container_image** (*string*): (gromacs/gromacs:latest) Container Image identifier..
- **container_volume_path** (*string*): (/data) Path to an internal directory in the container..
- **container_working_dir** (*string*): (None) Path to the internal CWD in the container..
- **container_user_id** (*string*): (None) User number id to be mapped inside the container..
- **container_shell_path** (*string*): (/bin/bash) Path to the binary executable of the container shell..

YAML

Common config file

```
properties:
  gmx_path: gmx
  num_threads: 0
```

Docker config file

```
properties:
  container_image: longr/gromacs-docker:latest
  container_path: docker
  container_volume_path: /inout
```

Singularity config file

```
properties:
  container_image: gromacs.simg
  container_path: singularity
  container_volume_path: /inout
```

Command line

```
mdrun --config config_mdrun.yml --input_tpr_path mdrun.tpr --output_trr_path ref_
↪mdrun.trr --output_gro_path ref_mdrun.gro --output_edr_path ref_mdrun.edr --output_
↪log_path ref_mdrun.log --input_cpt_path input.cpt --output_xtc_path output.xtc --
↪output_cpt_path output.cpt --output_dhdl_path output.xvg
```

JSON

Common config file

```
{
  "properties": {
    "num_threads": 0,
    "gmx_path": "gmx"
  }
}
```

Docker config file

```
{
  "properties": {
    "container_path": "docker",
    "container_image": "longr/gromacs-docker:latest",
    "container_volume_path": "/inout"
  }
}
```

Singularity config file

```
{
  "properties": {
    "container_path": "singularity",
    "container_image": "gromacs.simg",
    "container_volume_path": "/inout"
  }
}
```

Command line

```
mdrun --config config_mdrun.json --input_tpr_path mdrun.tpr --output_trr_path ref_
↪mdrun.trr --output_gro_path ref_mdrun.gro --output_edr_path ref_mdrun.edr --output_
↪log_path ref_mdrun.log --input_cpt_path input.cpt --output_xtc_path output.xtc --
↪output_cpt_path output.cpt --output_dhdl_path output.xvg
```

1.3.2 Make_ndx

Wrapper of the GROMACS make_ndx module.

Get help

Command:

```
make_ndx -h
```

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

I / O Arguments

Syntax: input_argument (datatype) : Definition

Config input / output arguments for this building block:

- **input_structure_path** (*string*): Path to the input GRO/PDB/TPR file. File type: input. Sample file. Accepted formats: GRO, PDB, TPR
- **output_ndx_path** (*string*): Path to the output index NDX file. File type: output. Sample file. Accepted formats: NDX
- **input_ndx_path** (*string*): Path to the input index NDX file. File type: input. Sample file. Accepted formats: NDX

Config

Syntax: input_parameter (datatype) - (default_value) Definition

Config parameters for this building block:

- **selection** (*string*): (a CA C N O) Heavy atoms. Atom selection string..
- **gmx_lib** (*string*): (None) Path set GROMACS GMXLIB environment variable..
- **gmx_path** (*string*): (gmx) Path to the GROMACS executable binary..
- **remove_tmp** (*boolean*): (True) Remove temporal files..
- **restart** (*boolean*): (False) Do not execute if output files exist..
- **container_path** (*string*): (None) Path to the binary executable of your container..
- **container_image** (*string*): (gromacs/gromacs:latest) Container Image identifier..
- **container_volume_path** (*string*): (/data) Path to an internal directory in the container..
- **container_working_dir** (*string*): (None) Path to the internal CWD in the container..
- **container_user_id** (*string*): (None) User number id to be mapped inside the container..
- **container_shell_path** (*string*): (/bin/bash) Path to the binary executable of the container shell..

YAML

Common config file

```
properties:
  selection: a CA C N O
```

Docker config file

```
properties:
  container_image: longr/gromacs-docker:latest
  container_path: docker
  container_volume_path: /tmp
```

Singularity config file

```
properties:
  container_image: gromacs.simg
  container_path: singularity
  container_volume_path: /tmp
```

Command line

```
make_ndx --config config_make_ndx.yml --input_structure_path make_ndx.tpr --output_
↪ndx_path ref_make_ndx.ndx --input_ndx_path input.ndx
```

JSON

Common config file

```
{
  "properties": {
    "selection": "a CA C N O"
  }
}
```

Docker config file

```
{
  "properties": {
    "container_path": "docker",
    "container_image": "longr/gromacs-docker:latest",
    "container_volume_path": "/tmp"
  }
}
```

Singularity config file

```
{
  "properties": {
    "container_path": "singularity",
    "container_image": "gromacs.simg",
    "container_volume_path": "/tmp"
```

(continues on next page)

(continued from previous page)

```
    }
}
```

Command line

```
make_ndx --config config_make_ndx.json --input_structure_path make_ndx.tpr --output_
↪ndx_path ref_make_ndx.ndx --input_ndx_path input.ndx
```

1.3.3 Editconf

Wrapper class for the GROMACS editconf module.

Get help

Command:

```
editconf -h
```

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

I / O Arguments

Syntax: input_argument (datatype) : Definition

Config input / output arguments for this building block:

- **input_gro_path** (*string*): Path to the input GRO file. File type: input. Sample file. Accepted formats: GRO
- **output_gro_path** (*string*): Path to the output GRO file. File type: output. Sample file. Accepted formats: GRO

Config

Syntax: input_parameter (datatype) - (default_value) Definition

Config parameters for this building block:

- **distance_to_molecule** (*number*): (1.0) Distance of the box from the outermost atom in nm. ie 1.0nm = 10 Angstroms..
- **box_type** (*string*): (cubic) Geometrical shape of the solvent box. .
- **center_molecule** (*boolean*): (True) Center molecule in the box..
- **gmx_lib** (*string*): (None) Path set GROMACS GMXLIB environment variable..
- **gmx_path** (*string*): (gmx) Path to the GROMACS executable binary..
- **remove_tmp** (*boolean*): (True) Remove temporal files..
- **restart** (*boolean*): (False) Do not execute if output files exist..
- **container_path** (*string*): (None) Path to the binary executable of your container..
- **container_image** (*string*): (gromacs/gromacs:latest) Container Image identifier..

- **container_volume_path** (*string*): (/data) Path to an internal directory in the container..
- **container_working_dir** (*string*): (None) Path to the internal CWD in the container..
- **container_user_id** (*string*): (None) User number id to be mapped inside the container..
- **container_shell_path** (*string*): (/bin/bash) Path to the binary executable of the container shell..

YAML

Common config file

```
properties:
  box_type: cubic
  distance_to_molecule: 1.0
```

Docker config file

```
properties:
  container_image: longr/gromacs-docker:latest
  container_path: docker
  container_volume_path: /tmp
```

Singularity config file

```
properties:
  container_image: gromacs.simg
  container_path: singularity
  container_volume_path: /tmp
```

Command line

```
editconf --config config_editconf.yml --input_gro_path editconf.gro --output_gro_path␣
↪ref_editconf.gro
```

JSON

Common config file

```
{
  "properties": {
    "distance_to_molecule": 1.0,
    "box_type": "cubic"
  }
}
```

Docker config file

```
{
  "properties": {
    "container_path": "docker",
    "container_image": "longr/gromacs-docker:latest",
    "container_volume_path": "/tmp"
  }
}
```

Singularity config file

```
{
  "properties": {
    "container_path": "singularity",
    "container_image": "gromacs.simg",
    "container_volume_path": "/tmp"
  }
}
```

Command line

```
editconf --config config_editconf.json --input_gro_path editconf.gro --output_gro_
↪path ref_editconf.gro
```

1.3.4 Ndx2resttop

Generate a restrained topology from an index NDX file.

Get help

Command:

```
ndx2resttop -h
```

```
/bin/sh: ndx2resttop: command not found
```

I / O Arguments

Syntax: input_argument (datatype) : Definition

Config input / output arguments for this building block:

- **input_ndx_path** (*string*): Path to the input NDX index file. File type: input. Sample file. Accepted formats: NDX
- **input_top_zip_path** (*string*): Path the input TOP topology in zip format. File type: input. Sample file. Accepted formats: ZIP

- **output_top_zip_path** (*string*): Path the output TOP topology in zip format. File type: output. Sample file. Accepted formats: ZIP

Config

Syntax: input_parameter (datatype) - (default_value) Definition

Config parameters for this building block:

- **force_constants** (*string*): (500 500 500) Array of three floats defining the force constants..
- **ref_rest_chain_triplet_list** (*string*): (None) Triplet list composed by (reference group, restrain group, chain) list..

YAML

Common config file

```
properties:
  ref_rest_chain_triplet_list: ( Chain_A, Chain_A_noMut, A ), ( Chain_B, Chain_B_
↪noMut,
    B ), ( Chain_C, Chain_C_noMut, C ), ( Chain_D, Chain_D_noMut, D )
```

Command line

```
ndx2resttop --config config_ndx2resttop.yml --input_ndx_path ndx2resttop.ndx --input_
↪top_zip_path ndx2resttop.zip --output_top_zip_path ref_ndx2resttop.zip
```

JSON

Common config file

```
{
  "properties": {
    "ref_rest_chain_triplet_list": "( Chain_A, Chain_A_noMut, A ), ( Chain_B, Chain_B_
↪noMut, B ), ( Chain_C, Chain_C_noMut, C ), ( Chain_D, Chain_D_noMut, D )"
  }
}
```

Command line

```
ndx2resttop --config config_ndx2resttop.json --input_ndx_path ndx2resttop.ndx --input_
↪top_zip_path ndx2resttop.zip --output_top_zip_path ref_ndx2resttop.zip
```

1.3.5 Grompp

Wrapper of the GROMACS grompp module.

Get help

Command:

```
grompp -h
```

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

I / O Arguments

Syntax: input_argument (datatype) : Definition

Config input / output arguments for this building block:

- **input_gro_path** (*string*): Path to the input GROMACS structure GRO file. File type: input. Sample file. Accepted formats: GRO
- **input_top_zip_path** (*string*): Path to the input GROMACS topology TOP and ITP files in zip format. File type: input. Sample file. Accepted formats: ZIP
- **output_tpr_path** (*string*): Path to the output portable binary run file TPR. File type: output. Sample file. Accepted formats: TPR
- **input_cpt_path** (*string*): Path to the input GROMACS checkpoint file CPT. File type: input. Sample file. Accepted formats: CPT
- **input_ndx_path** (*string*): Path to the input GROMACS index files NDX. File type: input. Sample file. Accepted formats: NDX
- **input_mdp_path** (*string*): Path to the input GROMACS MDP file. File type: input. Sample file. Accepted formats: MDP

Config

Syntax: input_parameter (datatype) - (default_value) Definition

Config parameters for this building block:

- **mdp** (*object*): ({}) MDP options specification..
- **simulation_type** (*string*): (None) Default options for the mdp file. Each one creates a different mdp file. .
- **maxwarn** (*integer*): (0) Maximum number of allowed warnings. If simulation_type is index default is 10..
- **gmx_lib** (*string*): (None) Path set GROMACS GMXLIB environment variable..
- **gmx_path** (*string*): (gmx) Path to the GROMACS executable binary..
- **remove_tmp** (*boolean*): (True) Remove temporal files..
- **restart** (*boolean*): (False) Do not execute if output files exist..
- **container_path** (*string*): (None) Path to the binary executable of your container..
- **container_image** (*string*): (gromacs/gromacs:latest) Container Image identifier..
- **container_volume_path** (*string*): (/data) Path to an internal directory in the container..
- **container_working_dir** (*string*): (None) Path to the internal CWD in the container..
- **container_user_id** (*string*): (None) User number id to be mapped inside the container..
- **container_shell_path** (*string*): (/bin/bash) Path to the binary executable of the container shell..

YAML

Common config file

```
properties:
  maxwarn: 1
  mdp:
    ld-seed: '1'
```

Docker config file

```
properties:
  container_image: gromacs/gromacs:latest
  container_path: docker
  container_volume_path: /inout
  container_working_dir: /inout
  maxwarn: 1
  mdp:
    ld-seed: '1'
```

Singularity config file

```
properties:
  container_image: gromacs.simg
  container_path: singularity
  container_volume_path: /inout
  container_working_dir: /inout
  maxwarn: 1
  mdp:
    ld-seed: '1'
```

Command line

```
grompp --config config_grompp.yml --input_gro_path grompp.gro --input_top_zip_path␣
↪grompp.zip --output_tpr_path ref_grompp.tpr --input_cpt_path input.cpt --input_ndx_
↪path input.ndx --input_mdp_path input.mdp
```

JSON

Common config file

```
{
  "properties": {
    "maxwarn": 1,
    "mdp": {
      "ld-seed": "1"
    }
  }
}
```

Docker config file

```
{
  "properties": {
    "maxwarn": 1,
    "mdp": {
      "ld-seed": "1"
    },
    "container_path": "docker",
    "container_image": "gromacs/gromacs:latest",
    "container_volume_path": "/inout",
    "container_working_dir": "/inout"
  }
}
```

Singularity config file

```
{
  "properties": {
    "maxwarn": 1,
    "mdp": {
      "ld-seed": "1"
    },
    "container_path": "singularity",
    "container_image": "gromacs.simg",
    "container_volume_path": "/inout",
    "container_working_dir": "/inout"
  }
}
```

Command line

```
grompp --config config_grompp.json --input_gro_path grompp.gro --input_top_zip_path␣
↪grompp.zip --output_tpr_path ref_grompp.tpr --input_cpt_path input.cpt --input_ndx_
↪path input.ndx --input_mdp_path input.mdp
```

1.3.6 Gmxselect

Wrapper of the GROMACS select module.

Get help

Command:

```
gmxselect -h
```

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

I / O Arguments

Syntax: input_argument (datatype) : Definition

Config input / output arguments for this building block:

- **input_structure_path** (*string*): Path to the input GRO/PDB/TPR file. File type: input. Sample file. Accepted formats: PDB, GRO, TPR
- **output_ndx_path** (*string*): Path to the output index NDX file. File type: output. Sample file. Accepted formats: NDX
- **input_ndx_path** (*string*): Path to the input index NDX file. File type: input. Sample file. Accepted formats: NDX

Config

Syntax: input_parameter (datatype) - (default_value) Definition

Config parameters for this building block:

- **selection** (*string*): (a CA C N O) Heavy atoms. Atom selection string..
- **append** (*boolean*): (False) Append the content of the input_ndx_path to the output_ndx_path..
- **gmx_path** (*string*): (gmx) Path to the GROMACS executable binary..
- **gmx_lib** (*string*): (None) Path set GROMACS GMXLIB environment variable..
- **remove_tmp** (*boolean*): (True) Remove temporal files..
- **restart** (*boolean*): (False) Do not execute if output files exist..
- **container_path** (*string*): (None) Path to the binary executable of your container..
- **container_image** (*string*): (gromacs/gromacs:latest) Container Image identifier..
- **container_volume_path** (*string*): (/data) Path to an internal directory in the container..
- **container_working_dir** (*string*): (None) Path to the internal CWD in the container..
- **container_user_id** (*string*): (None) User number id to be mapped inside the container..
- **container_shell_path** (*string*): (/bin/bash) Path to the binary executable of the container shell..

YAML

Common config file

```
properties:
  selection: '"Mynewgroup" group "Protein-H" and not same residue as within 0.4 of
    resname ARG'
```

Docker config file

```
properties:
  container_image: gromacs/gromacs:latest
  container_path: docker
  container_volume_path: /inout
  selection: \"Mynewgroup\" group \"Protein-H\" and not same residue as within 0.4
    of resname ARG
```

Singularity config file

```
properties:
  container_image: gromacs.simg
  container_path: singularity
  container_volume_path: /inout
  selection: '"Mynewgroup" group "Protein-H" and not same residue as within 0.4 of
    resname ARG'
```

Command line

```
gmxselect --config config_gmxselect.yml --input_structure_path make_ndx.tpr --output_
↪ndx_path ref_select.ndx --input_ndx_path input.ndx
```

JSON

Common config file

```
{
  "properties": {
    "selection": "\"Mynewgroup\" group \"Protein-H\" and not same residue as within 0.
↪4 of resname ARG"
  }
}
```

Docker config file

```
{
  "properties": {
    "selection": "\\\"Mynewgroup\\\" group \\\"Protein-H\\\" and not same residue as␣
↪within 0.4 of resname ARG",
    "container_path": "docker",
    "container_image": "gromacs/gromacs:latest",
    "container_volume_path": "/inout"
  }
}
```

Singularity config file

```
{
  "properties": {
    "selection": "\"Mynewgroup\" group \"Protein-H\" and not same residue as within 0.
↪4 of resname ARG",
    "container_path": "singularity",
    "container_image": "gromacs.simg",
    "container_volume_path": "/inout"
  }
}
```

Command line

```
gmxselect --config config_gmxselect.json --input_structure_path make_ndx.tpr --output_
↪ndx_path ref_select.ndx --input_ndx_path input.ndx
```

1.3.7 Pdb2gmx

Wrapper class for the GROMACS pdb2gmx module.

Get help

Command:

```
pdb2gmx -h
```

```
/bin/sh: pdb2gmx: command not found
```

I / O Arguments

Syntax: input_argument (datatype) : Definition

Config input / output arguments for this building block:

- **input_pdb_path** (*string*): Path to the input PDB file. File type: input. Sample file. Accepted formats: PDB
- **output_gro_path** (*string*): Path to the output GRO file. File type: output. Sample file. Accepted formats: GRO
- **output_top_zip_path** (*string*): Path the output TOP topology in zip format. File type: output. Sample file. Accepted formats: ZIP

Config

Syntax: input_parameter (datatype) - (default_value) Definition

Config parameters for this building block:

- **water_type** (*string*): (spce) Water molecule type. .
- **force_field** (*string*): (amber99sb-ildn) Force field to be used during the conversion. .
- **ignh** (*boolean*): (False) Should pdb2gmx ignore the hidrogens in the original structure..

- **his** (*string*): (None) Histidine protonation array..
- **merge** (*boolean*): (False) Merge all chains into a single molecule..
- **gmx_lib** (*string*): (None) Path set GROMACS GMXLIB environment variable..
- **gmx_path** (*string*): (gmx) Path to the GROMACS executable binary..
- **remove_tmp** (*boolean*): (True) Remove temporal files..
- **restart** (*boolean*): (False) Do not execute if output files exist..
- **container_path** (*string*): (None) Path to the binary executable of your container..
- **container_image** (*string*): (gromacs/gromacs:latest) Container Image identifier..
- **container_volume_path** (*string*): (/data) Path to an internal directory in the container..
- **container_working_dir** (*string*): (None) Path to the internal CWD in the container..
- **container_user_id** (*string*): (None) User number id to be mapped inside the container..
- **container_shell_path** (*string*): (/bin/bash) Path to the binary executable of the container shell..

YAML

Common config file

```
properties:
  his: 0 0 1 1 0 0 0
```

Docker config file

```
properties:
  container_image: longr/gromacs-docker:latest
  container_path: docker
  container_volume_path: /inout
  his: 0 0 1 1 0 0 0
```

Singularity config file

```
properties:
  container_image: gromacs.simg
  container_path: singularity
  container_volume_path: /inout
  his: 0 0 1 1 0 0 0
```

Command line

```
pdb2gmx --config config_pdb2gmx.yml --input_pdb_path egfr.pdb --output_gro_path ref_
↪pdb2gmx.gro --output_top_zip_path ref_pdb2gmx.zip
```

JSON

Common config file

```
{
  "properties": {
    "his": "0 0 1 1 0 0 0"
  }
}
```

Docker config file

```
{
  "properties": {
    "his": "0 0 1 1 0 0 0",
    "container_path": "docker",
    "container_image": "longr/gromacs-docker:latest",
    "container_volume_path": "/inout"
  }
}
```

Singularity config file

```
{
  "properties": {
    "his": "0 0 1 1 0 0 0",
    "container_path": "singularity",
    "container_image": "gromacs.simg",
    "container_volume_path": "/inout"
  }
}
```

Command line

```
pdb2gmx --config config_pdb2gmx.json --input_pdb_path egfr.pdb --output_gro_path ref_
↪pdb2gmx.gro --output_top_zip_path ref_pdb2gmx.zip
```

1.3.8 Append_ligand

This class takes a ligand ITP file and inserts it in a topology.

Get help

Command:

```
append_ligand -h
```

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

I / O Arguments

Syntax: input_argument (datatype) : Definition

Config input / output arguments for this building block:

- **input_top_zip_path** (*string*): Path the input topology TOP and ITP files zipball. File type: input. Sample file. Accepted formats: ZIP
- **input_itp_path** (*string*): Path to the ligand ITP file to be inserted in the topology. File type: input. Sample file. Accepted formats: ITP
- **output_top_zip_path** (*string*): Path/Name the output topology TOP and ITP files zipball. File type: output. Sample file. Accepted formats: ZIP
- **input_posres_itp_path** (*string*): Path to the position restriction ITP file. File type: input. Sample file. Accepted formats: ITP

Config

Syntax: input_parameter (datatype) - (default_value) Definition

Config parameters for this building block:

- **posres_name** (*string*): (POSRES_LIGAND) String to be included in the ifdef clause..
- **remove_tmp** (*boolean*): (True) Remove temporal files..
- **restart** (*boolean*): (False) Do not execute if output files exist..

YAML

Common config file

```
properties:
  posres_name: POSRES_LIGAND
```

Command line

```
append_ligand --config config_append_ligand.yml --input_top_zip_path ndx2resttop.zip -
↪-input_itp_path pep_ligand.itp --output_top_zip_path ref_appendligand.zip --input_
↪posres_itp_path input.itp
```

JSON

Common config file

```
{
  "properties": {
    "posres_name": "POSRES_LIGAND"
  }
}
```

Command line

```
append_ligand --config config_append_ligand.json --input_top_zip_path ndx2resttop.zip␣
↪--input_itp_path pep_ligand.itp --output_top_zip_path ref_appendligand.zip --input_
↪posres_itp_path input.itp
```

1.3.9 Solvate

Wrapper of the GROMACS solvate module.

Get help

Command:

```
solvate -h
```

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

I / O Arguments

Syntax: input_argument (datatype) : Definition

Config input / output arguments for this building block:

- **input_solute_gro_path** (*string*): Path to the input GRO file. File type: input. Sample file. Accepted formats: GRO
- **output_gro_path** (*string*): Path to the output GRO file. File type: output. Sample file. Accepted formats: GRO
- **input_top_zip_path** (*string*): Path the input TOP topology in zip format. File type: input. Sample file. Accepted formats: ZIP
- **output_top_zip_path** (*string*): Path the output topology in zip format. File type: output. Sample file. Accepted formats: ZIP
- **input_solvent_gro_path** (*string*): (spc216.gro) Path to the GRO file containing the structure of the solvent. File type: input. Sample file. Accepted formats: GRO

Config

Syntax: input_parameter (datatype) - (default_value) Definition

Config parameters for this building block:

- **shell** (*number*): (0.0) Thickness in nanometers of optional water layer around solute..
- **gmx_lib** (*string*): (None) Path set GROMACS GMXLIB environment variable..

- **gmx_path** (*string*): (gmx) Path to the GROMACS executable binary..
- **remove_tmp** (*boolean*): (True) Remove temporal files..
- **restart** (*boolean*): (False) Do not execute if output files exist..
- **container_path** (*string*): (None) Path to the binary executable of your container..
- **container_image** (*string*): (gromacs/gromacs:latest) Container Image identifier..
- **container_volume_path** (*string*): (/data) Path to an internal directory in the container..
- **container_working_dir** (*string*): (None) Path to the internal CWD in the container..
- **container_user_id** (*string*): (None) User number id to be mapped inside the container..
- **container_shell_path** (*string*): (/bin/bash) Path to the binary executable of the container shell..

YAML

Common config file

```
properties:
  gmx_path: gmx
  restart: 'False'
```

Docker config file

```
properties:
  container_image: longr/gromacs-docker:latest
  container_path: docker
  container_volume_path: /inout
```

Singularity config file

```
properties:
  container_image: gromacs.simg
  container_path: singularity
  container_volume_path: /inout
```

Command line

```
solvate --config config_solvate.yml --input_solute_gro_path solvate.gro --output_gro_
↪path ref_solvate.gro --input_top_zip_path solvate.zip --output_top_zip_path ref_
↪solvate.zip --input_solvent_gro_path input.gro
```

JSON

Common config file

```
{
  "properties": {
    "gmx_path": "gmx",
    "restart": "False"
  }
}
```

Docker config file

```
{
  "properties": {
    "container_path": "docker",
    "container_image": "longr/gromacs-docker:latest",
    "container_volume_path": "/inout"
  }
}
```

Singularity config file

```
{
  "properties": {
    "container_path": "singularity",
    "container_image": "gromacs.simg",
    "container_volume_path": "/inout"
  }
}
```

Command line

```
solvate --config config_solvate.json --input_solute_gro_path solvate.gro --output_gro_
↪path ref_solvate.gro --input_top_zip_path solvate.zip --output_top_zip_path ref_
↪solvate.zip --input_solvent_gro_path input.gro
```

1.3.10 Genion

Wrapper class for the GROMACS genion module.

Get help

Command:

```
genion -h
```

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

I / O Arguments

Syntax: input_argument (datatype) : Definition

Config input / output arguments for this building block:

- **input_tpr_path** (*string*): Path to the input portable run input TPR file. File type: input. Sample file. Accepted formats: TPR
- **output_gro_path** (*string*): Path to the input structure GRO file. File type: output. Sample file. Accepted formats: GRO
- **input_top_zip_path** (*string*): Path the input TOP topology in zip format. File type: input. Sample file. Accepted formats: ZIP
- **output_top_zip_path** (*string*): Path the output topology TOP and ITP files zipball. File type: output. Sample file. Accepted formats: ZIP
- **input_ndx_path** (*string*): Path to the input index NDX file. File type: input. Sample file. Accepted formats: NDX

Config

Syntax: input_parameter (datatype) - (default_value) Definition

Config parameters for this building block:

- **replaced_group** (*string*): (SOL) Group of molecules that will be replaced by the solvent..
- **neutral** (*boolean*): (False) Neutralize the charge of the system..
- **concentration** (*number*): (0.05) Concentration of the ions in (mol/liter)..
- **seed** (*integer*): (1993) Seed for random number generator..
- **gmx_lib** (*string*): (None) Path set GROMACS GMXLIB environment variable..
- **gmx_path** (*string*): (gmx) Path to the GROMACS executable binary..
- **remove_tmp** (*boolean*): (True) Remove temporal files..
- **restart** (*boolean*): (False) Do not execute if output files exist..
- **container_path** (*string*): (None) Path to the binary executable of your container..
- **container_image** (*string*): (gromacs/gromacs:latest) Container Image identifier..
- **container_volume_path** (*string*): (/data) Path to an internal directory in the container..
- **container_working_dir** (*string*): (None) Path to the internal CWD in the container..
- **container_user_id** (*string*): (None) User number id to be mapped inside the container..
- **container_shell_path** (*string*): (/bin/bash) Path to the binary executable of the container shell..

YAML

Common config file

```
properties:
  concentration: 0.05
  replaced_group: SOL
```

Docker config file

```
properties:
  container_image: longr/gromacs-docker:latest
  container_path: docker
  container_volume_path: /data
  container_working_dir: /data
```

Singularity config file

```
properties:
  container_image: gromacs.simg
  container_path: singularity
  container_volume_path: /data
  container_working_dir: /data
```

Command line

```
genion --config config_genion.yml --input_tpr_path genion.tpr --output_gro_path ref_
↪genion.gro --input_top_zip_path genion.zip --output_top_zip_path ref_genion.zip --
↪input_ndx_path input.ndx
```

JSON

Common config file

```
{
  "properties": {
    "concentration": 0.05,
    "replaced_group": "SOL"
  }
}
```

Docker config file

```
{
  "properties": {
    "container_path": "docker",
    "container_image": "longr/gromacs-docker:latest",
    "container_volume_path": "/data",
    "container_working_dir": "/data"
  }
}
```

Singularity config file

```
{
  "properties": {
    "container_path": "singularity",
    "container_image": "gromacs.simg",
    "container_volume_path": "/data",
    "container_working_dir": "/data"
  }
}
```

Command line

```
genion --config config_genion.json --input_tpr_path genion.tpr --output_gro_path ref_
↪genion.gro --input_top_zip_path genion.zip --output_top_zip_path ref_genion.zip --
↪input_ndx_path input.ndx
```

1.3.11 Grompp_mdrun

Wrapper of the GROMACS grompp module and the GROMACS mdrun module.

Get help

Command:

```
grompp_mdrun -h
```

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

I / O Arguments

Syntax: input_argument (datatype) : Definition

Config input / output arguments for this building block:

- **input_gro_path** (*string*): Path to the input GROMACS structure GRO file. File type: input. Sample file. Accepted formats: GRO
- **input_top_zip_path** (*string*): Path to the input GROMACS topology TOP and ITP files in zip format. File type: input. Sample file. Accepted formats: ZIP
- **output_trr_path** (*string*): Path to the GROMACS uncompressed raw trajectory file TRR. File type: output. Sample file. Accepted formats: TRR
- **output_gro_path** (*string*): Path to the output GROMACS structure GRO file. File type: output. Sample file. Accepted formats: GRO
- **output_edr_path** (*string*): Path to the output GROMACS portable energy file EDR. File type: output. Sample file. Accepted formats: EDR
- **output_log_path** (*string*): Path to the output GROMACS trajectory log file LOG. File type: output. Sample file. Accepted formats: LOG

- **input_cpt_path** (*string*): Path to the input GROMACS checkpoint file CPT. File type: input. Sample file. Accepted formats: CPT
- **input_ndx_path** (*string*): Path to the input GROMACS index files NDX. File type: input. Sample file. Accepted formats: NDX
- **input_mdp_path** (*string*): Path to the input GROMACS MDP file. File type: input. Sample file. Accepted formats: MDP
- **output_xtc_path** (*string*): Path to the GROMACS compressed trajectory file XTC. File type: output. Sample file. Accepted formats: XTC
- **output_cpt_path** (*string*): Path to the output GROMACS checkpoint file CPT. File type: output. Sample file. Accepted formats: CPT
- **output_dhdl_path** (*string*): Path to the output dhdl.xvg file only used when free energy calculation is turned on. File type: output. Sample file. Accepted formats: XVG

Config

Syntax: input_parameter (datatype) - (default_value) Definition

Config parameters for this building block:

- **mdp** (*object*): ({}) MDP options specification..
- **simulation_type** (*string*): (minimization) Default options for the mdp file. Each creates a different mdp file. .
- **maxwarn** (*integer*): (10) Maximum number of allowed warnings..
- **mpi_bin** (*string*): (None) Path to the MPI runner. Usually “mpirun” or “srun”..
- **mpi_np** (*string*): (None) Number of MPI processes. Usually an integer bigger than 1..
- **mpi_hostlist** (*string*): (None) Path to the MPI hostlist file..
- **checkpoint_time** (*integer*): (15) Checkpoint writing interval in minutes. Only enabled if an output_cpt_path is provided..
- **num_threads** (*integer*): (0) Let GROMACS guess. The number of threads that are going to be used..
- **num_threads_mpi** (*integer*): (0) Let GROMACS guess. The number of GROMACS MPI threads that are going to be used..
- **num_threads_omp** (*integer*): (0) Let GROMACS guess. The number of GROMACS OPENMP threads that are going to be used..
- **num_threads_omp_pme** (*integer*): (0) Let GROMACS guess. The number of GROMACS OPENMP_PME threads that are going to be used..
- **use_gpu** (*boolean*): (False) Use settings appropriate for GPU. Adds: -nb gpu -pme gpu.
- **gpu_id** (*string*): (None) List of unique GPU device IDs available to use..
- **gpu_tasks** (*string*): (None) List of GPU device IDs, mapping each PP task on each node to a device..
- **gmx_lib** (*string*): (None) Path set GROMACS GMXLIB environment variable..
- **gmx_path** (*string*): (gmx) Path to the GROMACS executable binary..
- **remove_tmp** (*boolean*): (True) Remove temporal files..
- **restart** (*boolean*): (False) Do not execute if output files exist..
- **container_path** (*string*): (None) Path to the binary executable of your container..

- **container_image** (*string*): (gromacs/gromacs:latest) Container Image identifier..
- **container_volume_path** (*string*): (/data) Path to an internal directory in the container..
- **container_working_dir** (*string*): (None) Path to the internal CWD in the container..
- **container_user_id** (*string*): (None) User number id to be mapped inside the container..
- **container_shell_path** (*string*): (/bin/bash) Path to the binary executable of the container shell..

YAML

Common config file

```
properties:
  gmx_path: gmx
  maxwarn: 1
  mdp:
    dt: 0.0001
    ld-seed: '1'
  num_threads: 0
  simulation_type: free
```

Docker config file

```
properties:
  container_image: longr/gromacs-docker:latest
  container_path: docker
  container_volume_path: /inout
  gmx_path: gmx
  maxwarn: 1
  mdp:
    dt: 0.0001
    ld-seed: '1'
  num_threads: 0
  simulation_type: free
```

Singularity config file

```
properties:
  container_image: gromacs.simg
  container_path: singularity
  container_volume_path: /inout
  gmx_path: gmx
  mdp:
    dt: 0.0001
    ld-seed: '1'
  num_threads: 0
  simulation_type: free
```

Command line

```
grompp_mdrun --config config_grompp_mdrun.yml --input_gro_path grompp.gro --input_top_
↪zip_path grompp.zip --output_trr_path ref_mdrun.trr --output_gro_path ref_mdrun.gro␣
↪--output_edr_path ref_mdrun.edr --output_log_path ref_gmx_mdrun.log --input_cpt_
↪path input.cpt --input_ndx_path input.ndx --input_mdp_path input.mdp --output_xtc_
↪path output.xtc --output_cpt_path output.cpt --output_dhdl_path output.xvg
```

JSON

Common config file

```
{
  "properties": {
    "simulation_type": "free",
    "maxwarn": 1,
    "mdp": {
      "ld-seed": "1",
      "dt": 0.0001
    },
    "num_threads": 0,
    "gmx_path": "gmx"
  }
}
```

Docker config file

```
{
  "properties": {
    "simulation_type": "free",
    "maxwarn": 1,
    "mdp": {
      "ld-seed": "1",
      "dt": 0.0001
    },
    "num_threads": 0,
    "gmx_path": "gmx",
    "container_path": "docker",
    "container_image": "longr/gromacs-docker:latest",
    "container_volume_path": "/inout"
  }
}
```

Singularity config file

```
{
  "properties": {
    "simulation_type": "free",
    "mdp": {
      "ld-seed": "1",
```

(continues on next page)

(continued from previous page)

```
      "dt": 0.0001
    },
    "num_threads": 0,
    "gmx_path": "gmx",
    "container_path": "singularity",
    "container_image": "gromacs.simg",
    "container_volume_path": "/inout"
  }
}
```

Command line

```
grompp_mdrun --config config_grompp_mdrun.json --input_gro_path grompp.gro --input_
↪top_zip_path grompp.zip --output_trr_path ref_mdrun.trr --output_gro_path ref_mdrun.
↪gro --output_edr_path ref_mdrun.edr --output_log_path ref_gmx_mdrun.log --input_cpt_
↪path input.cpt --input_ndx_path input.ndx --input_mdp_path input.mdp --output_xtc_
↪path output.xtc --output_cpt_path output.cpt --output_dhdl_path output.xvg
```

1.3.12 Genrestr

Wrapper of the GROMACS genrestr module.

Get help

Command:

```
genrestr -h
```

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

I / O Arguments

Syntax: input_argument (datatype) : Definition

Config input / output arguments for this building block:

- **input_structure_path** (*string*): Path to the input structure PDB, GRO or TPR format. File type: input. Sample file. Accepted formats: PDB, GRO, TPR
- **output_itp_path** (*string*): Path the output ITP topology file with restrains. File type: output. Sample file. Accepted formats: ITP
- **input_ndx_path** (*string*): Path to the input GROMACS index file, NDX format. File type: input. Sample file. Accepted formats: NDX

Config

Syntax: input_parameter (datatype) - (default_value) Definition

Config parameters for this building block:

- **restrained_group** (*string*): (system) Index group that will be restrained..
- **force_constants** (*string*): (500 500 500) Array of three floats defining the force constants.
- **gmx_lib** (*string*): (None) Path set GROMACS GMXLIB environment variable..
- **gmx_path** (*string*): (gmx) Path to the GROMACS executable binary..
- **remove_tmp** (*boolean*): (True) Remove temporal files..
- **restart** (*boolean*): (False) Do not execute if output files exist..
- **container_path** (*string*): (None) Path to the binary executable of your container..
- **container_image** (*string*): (gromacs/gromacs:latest) Container Image identifier..
- **container_volume_path** (*string*): (/data) Path to an internal directory in the container..
- **container_working_dir** (*string*): (None) Path to the internal CWD in the container..
- **container_user_id** (*string*): (None) User number id to be mapped inside the container..
- **container_shell_path** (*string*): (/bin/bash) Path to the binary executable of the container shell..

YAML

Common config file

```
properties:
  force_constants: 500 500 500
  restrained_group: system
```

Docker config file

```
properties:
  container_image: longr/gromacs-docker:latest
  container_path: docker
  container_volume_path: /data
  container_working_dir: /data
```

Singularity config file

```
properties:
  container_image: gromacs.simg
  container_path: singularity
  container_volume_path: /data
  container_working_dir: /data
```

Command line

```
genrestr --config config_genrestr.yml --input_structure_path genrestr.gro --output_
↪itp_path ref_genrestr.itp --input_ndx_path genrestr.ndx
```

JSON

Common config file

```
{
  "properties": {
    "restrained_group": "system",
    "force_constants": "500 500 500"
  }
}
```

Docker config file

```
{
  "properties": {
    "container_path": "docker",
    "container_image": "longr/gromacs-docker:latest",
    "container_volume_path": "/data",
    "container_working_dir": "/data"
  }
}
```

Singularity config file

```
{
  "properties": {
    "container_path": "singularity",
    "container_image": "gromacs.simg",
    "container_volume_path": "/data",
    "container_working_dir": "/data"
  }
}
```

Command line

```
genrestr --config config_genrestr.json --input_structure_path genrestr.gro --output_
↪itp_path ref_genrestr.itp --input_ndx_path genrestr.ndx
```

1.4 Biobb MD changelog

1.4.1 What's new in version 3.7.1?

Bug fixes

- Problem with Gmx version check (mdrun) #48

1.4.2 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)

1.4.3 What's new in version 3.6.0?

In version 3.6.0 Python has been updated to version 3.7 and Biopython to version 1.79. 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 1.79 (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

Other changes

- New documentation styles (documentation) #8

CHAPTER 2

Indices and tables

- genindex
- modindex
- search

CHAPTER 3

Github repository.

Python Module Index

g

Index

S