
DP-GEN

DeepModeling

Apr 22, 2024

OVERVIEW

1	Overview	1
1.1	About DP-GEN	1
1.2	Download and install	1
1.3	Use DP-GEN	2
1.4	Case Studies	2
1.5	License	2
2	Command line interface	3
2.1	Sub-commands	3
3	Code Structure	9
4	Run	11
4.1	Overview of the Run process	11
4.2	Example-of-param.json	13
4.3	Example of machine.json	16
4.4	dpgen run param parameters	18
4.5	dpgen run machine parameters	40
5	Init	77
5.1	Init_bulk	77
5.2	dpgen init_bulk parameters	78
5.3	dpgen init_bulk machine parameters	81
5.4	Init_surf	93
5.5	dpgen init_surf parameters	96
5.6	dpgen init_surf machine parameters	99
5.7	init_reaction	111
5.8	dpgen init_reaction parameters	112
5.9	dpgen init_reaction machine parameters	114
6	Simplify	153
6.1	Simplify	153
6.2	dpgen simplify parameters	155
6.3	dpgen simplify machine parameters	168
7	Auto test	207
7.1	Autotest Overview: Autotest for Deep Generator	207
7.2	Make run and post	209
7.3	Relaxation	210
7.4	Property	220
7.5	Refine	237

7.6	Reproduce	239
8	User Guide	243
8.1	Discussions	243
8.2	Issue	243
8.3	Tutorials	243
8.4	Example for parameters	243
8.5	Pull requests - How to contribute	244
8.6	Troubleshooting	244
8.7	Common Errors	244
9	Contributing Guide	247
9.1	Contributing Guide	247
10	DP-GEN API	251
10.1	dpngen package	251
11	Authors	305
	Bibliography	309
	Python Module Index	311
	Index	313

OVERVIEW

1.1 About DP-GEN

DP-GEN (Deep Generator) is a software written in Python, delicately designed to generate a deep learning based model of interatomic potential energy and force field. DP-GEN is dependent on [DeepMD-kit](#). With highly scalable interface with common softwares for molecular simulation, DP-GEN is capable to automatically prepare scripts and maintain job queues on HPC machines (High Performance Cluster) and analyze results.

If you use this software in any publication, please cite:

Yuzhi Zhang, Haidi Wang, Weijie Chen, Jinzhe Zeng, Linfeng Zhang, Han Wang, and Weinan E, DP-GEN: A concurrent learning platform for the generation of reliable deep learning based potential energy models, *Computer Physics Communications*, 2020, 107206.

1.1.1 Highlighted features

- **Accurate and efficient:** DP-GEN is capable to sample more than tens of million structures and select only a few for first principles calculation. DP-GEN will finally obtain a uniformly accurate model.
- **User-friendly and automatic:** Users may install and run DP-GEN easily. Once succusefully running, DP-GEN can dispatch and handle all jobs on HPCs, and thus there's no need for any personal effort.
- **Highly scalable:** With modularized code structures, users and developers can easily extend DP-GEN for their most relevant needs. DP-GEN currently supports for HPC systems (Slurm, PBS, LSF and cloud machines), Deep Potential interface with DeePMD-kit, MD interface with [LAMMPS](#), [Gromacs](#) and *ab-initio* calculation interface with VASP, PWSCF, CP2K, SIESTA and Gaussian, Abacus, PWMAT, etc . We're sincerely welcome and embraced to users' contributions, with more possibilities and cases to use DP-GEN.

1.2 Download and install

DP-GEN only supports Python 3.9 and above. You can use one of the following methods to install DP-GEN:

- Install via pip: `pip install dpgen`
- Install via conda: ``conda install -c conda-forge dpgen``
- Install from source code: `git clone https://github.com/deepmodeling/dpgen && pip install ./dpgen`

To test if the installation is successful, you may execute

```
dpgen -h
```

1.3 Use DP-GEN

A quick-start on using DPGEN can be found [here](#). You can follow the [Handson tutorial](#), it is friendly to new users.

1.4 Case Studies

- [Practical-Guidelines-for-DP](#)

Before starting a new Deep Potential (DP) project, we suggest people (especially those who are newbies) read the following context first to get some insights into what tools we can use, what kinds of risks and difficulties we may meet, and how we can advance a new DP project smoothly.

- [Convergence-Test](#)

to ensure the data quality, the reliability of the final model, as well as the feasibility of the project, a convergence test should be done first.

- [Gas-phase](#)

In this tutorial, we will take the simulation of methane combustion as an example and introduce the procedure of DP-based MD simulation.

- [Mg-Y_alloy](#)

We will briefly analyze the candidate configurational space of a metallic system by taking Mg-based Mg-Y binary alloy as an example. The task is divided into steps during the DP-GEN process.

- [Transfer-learning](#)

This tutorial will introduce how to implement potential energy surface (PES) transfer-learning by using the DP-GEN software. In DP-GEN (version > 0.8.0), the “simplify” module is designed for this purpose.

1.5 License

The project dpgen is licensed under [GNU LGPLv3.0](#)

COMMAND LINE INTERFACE

dpngen is a convenient script that uses DeepGenerator to prepare initial data, drive DeepMDkit and analyze results. This script works based on several sub-commands with their own options. To see the options for the sub-commands, type “dpngen sub-command -h”.

```
usage: dpngen [-h]
              {init_surf,init_bulk,auto_gen_param,init_reaction,run,run/report,collect,
              ↪simplify,autotest,db,gui}
              ...
```

2.1 Sub-commands

2.1.1 init_surf

Generating initial data for surface systems.

```
dpngen init_surf [-h] PARAM [MACHINE]
```

Positional Arguments

PARAM	parameter file, json/yaml format
MACHINE	machine file, json/yaml format

2.1.2 init_bulk

Generating initial data for bulk systems.

```
dpngen init_bulk [-h] PARAM [MACHINE]
```

Positional Arguments

PARAM	parameter file, json/yaml format
MACHINE	machine file, json/yaml format

2.1.3 auto_gen_param

auto gen param.json

```
dpngen auto_gen_param [-h] PARAM
```

Positional Arguments

PARAM	parameter file, json/yaml format
--------------	----------------------------------

2.1.4 init_reaction

Generating initial data for reactive systems.

```
dpngen init_reaction [-h] PARAM [MACHINE]
```

Positional Arguments

PARAM	parameter file, json/yaml format
MACHINE	machine file, json/yaml format

2.1.5 run

Main process of Deep Potential Generator.

```
dpngen run [-h] [-d] PARAM MACHINE
```

Positional Arguments

PARAM	parameter file, json/yaml format
MACHINE	machine file, json/yaml format

Named Arguments

-d, --debug log debug info
Default: False

2.1.6 run/report

Report the systems and the thermodynamic conditions of the labeled frames.

```
dpgen run/report [-h] [-s] [-i] [-t] [-p PARAM] [-v] JOB_DIR
```

Positional Arguments

JOB_DIR the directory of the DP-GEN job,

Named Arguments

-s, --stat-sys count the labeled frames for each system
Default: False

-i, --stat-iter print the iteration candidate,failed,accurate count and fp calculation,success and fail count
Default: False

-t, --stat-time print the iteration time, warning!! assume model_devi parallel cores == 1
Default: False

-p, --param the json file provides DP-GEN paramters, should be located in JOB_DIR
Default: "param.json"

-v, --verbose being loud
Default: False

2.1.7 collect

Collect data.

```
dpgen collect [-h] [-p PARAMETER] [-v] [-m] [-s] JOB_DIR OUTPUT
```

Positional Arguments

JOB_DIR the directory of the DP-GEN job
OUTPUT the output directory of data

Named Arguments

-p, --parameter	the json file provides DP-GEN paramters, should be located in JOB_DIR Default: "param.json"
-v, --verbose	print number of data in each system Default: False
-m, --merge	merge the systems with the same chemical formula Default: False
-s, --shuffle	shuffle the data systems Default: False

2.1.8 simplify

Simplify data.

```
dpngen simplify [-h] [-d] PARAM MACHINE
```

Positional Arguments

PARAM	parameter file, json/yaml format
MACHINE	machine file, json/yaml format

Named Arguments

-d, --debug	log debug info Default: False
--------------------	----------------------------------

2.1.9 autotest

Auto-test for Deep Potential.

```
dpngen autotest [-h] [-d] TASK PARAM [MACHINE]
```

Positional Arguments

TASK	task can be make, run or post
PARAM	parameter file, json/yaml format
MACHINE	machine file, json/yaml format

Named Arguments

-d, --debug	log debug info
	Default: False

2.1.10 db

Collecting data from DP-GEN.

```
dpgen db [-h] PARAM
```

Positional Arguments

PARAM	parameter file, json format
--------------	-----------------------------

2.1.11 gui

Serve DP-GUI.

```
dpgen gui [-h] [-p PORT] [--bind_all]
```

Named Arguments

-p, --port	The port to serve DP-GUI on. Default: 6042
--bind_all	Serve on all public interfaces. This will expose your DP-GUI instance to the network on both IPv4 and IPv6 (where available). Default: False

CODE STRUCTURE

Let's look at the home page of DP-GEN. <https://github.com/deepmodeling/dpgen>

```
build
CITATION.cff
conda
dist
doc
dpgen
dpgen.egg-info
examples
LICENSE
README.md
requirements.txt
setup.py
tests
```


- `tests` : unittest tools for developers.
- `examples`: templates for PARAM and MACHINE files for different software, versions and tasks. For details of the parameters in PARAM, you can refer to `TASK parameters` chapters in this document. If you are confused about how to set up a JSON file, you can also use `dpgui`

Most of the code related to DP-GEN functions is in the `dpgen` directory. Open the `dpgen` directory, and we can see

```
arginfo.py
auto_test
collect
data
database
_date.py
dispatcher
generator
__init__.py
main.py
__pycache__
remote
simplify
tools
util.py
_version.py
```

- `auto_test` corresponds to `dpgen autotest`, for undertaking materials property analysis.

- `collect` corresponds to `dpgen collect`.
- `data` corresponds to `dpgen init_bulk`, `dpgen init_surf` and `dpgen init_reaction`, for preparing initial data of bulk and surf systems.
- `database` is the source code for collecting data generated by DP-GEN and interface with database.
- `simplify` corresponds to `dpgen simplify`.
- `remote` and `dispatcher` : source code for automatically submitting scripts, maintaining job queues and collecting results. **Notice this part has been integrated into `dpdispatcher`** generator is the core part of DP-GEN. It's for main process of deep generator. Let's open this folder.



```
arginfo.py
ch4
__init__.py
lib
run.py
```

`run.py` is the core of DP-GEN, corresponding to `dpgen run`. We can find `make_train`, `run_train`, ... `post_fp`, and other steps related functions here.

4.1 Overview of the Run process

The run process contains a series of successive iterations, undertaken in order such as heating the system to certain temperatures. Each iteration is composed of three steps: exploration, labeling, and training. Accordingly, there are three sub-folders: 00.train, 01.model_devi, and 02.fp in each iteration.

00.train: DP-GEN will train several (default 4) models based on initial and generated data. The only difference between these models is the random seed for neural network initialization.

01.model_devi : represent for model-deviation. DP-GEN will use models obtained from 00.train to run Molecular Dynamics(default LAMMPS). Larger deviation for structure properties (default is the force of atoms) means less accuracy of the models. Using this criterion, a few structures will be selected and put into the next stage 02.fp for more accurate calculation based on First Principles.

02.fp : Selected structures will be calculated by first-principles methods(default VASP). DP-GEN will obtain some new data and put them together with initial data and data generated in previous iterations. After that, new training will be set up and DP-GEN will enter the next iteration!

In the run process of the DP-GEN, we need to specify the basic information about the system, the initial data, and details of the training, exploration, and labeling tasks. In addition, we need to specify the software, machine environment, and computing resource and enable the process of job generation, submission, query, and collection automatically. We can perform the run process as we expect by specifying the keywords in param.json and machine.json, and they will be introduced in detail in the following sections.

Here, we give a general description of the run process. We can execute the run process of DP-GEN easily by:

```
dpgen run param.json machine.json
```

The following files or folders will be created and upgraded by codes

- iter.00000x contains the main results that DP-GEN generates in the first iteration.
- record.dpgen records the current stage of the run process.
- dpgen.log includes time and iteration information.

When the first iteration is completed, the folder structure of iter.000000 is like this:

```
$ ls iter.000000
00.train 01.model_devi 02.fp
```

In folder iter.000000/ 00.train:

- Folder 00x contains the input and output files of the DeePMD-kit, in which a model is trained.

- graph.00x.pb is the model DeePMD-kit generates. The only difference between these models is the random seed for neural network initialization.

In folder iter.000000/ 01.model_devi

- Folder confs contains the initial configurations for LAMMPS MD converted from POSCAR you set in *sys_configs* of param.json.
- Folder task.000.00000x contains the input and output files of the LAMMPS. In folder task.000.00000x, file model_devi.out records the model deviation of concerned labels, energy and force in MD. It serves as the criterion for selecting which structures and doing first-principle calculations.

In folder iter.000000/ 02.fp

- candidate.shuffle.000.out records which structures will be selected from last step 01.model_devi. There are always far more candidates than the maximum you expect to calculate at one time. In this condition, DP-GEN will randomly choose up to *fp_task_max* structures and form the folder task.*.
- rest_accurate.shuffle.000.out records the other structures where our model is accurate (*max_devi_f* is less than *model_devi_f_trust_lo*, no need to calculate any more),
- rest_failed.shuffled.000.out records the other structures where our model is too inaccurate (larger than *model_devi_f_trust_hi*, there may be some error).
- data.000: After first-principle calculations, DP-GEN will collect these data and change them into the format DeePMD-kit needs. In the next iteration's 00.train, these data will be trained together as well as the initial data.

DP-GEN identifies the stage of the run process by a record file, record.dpgen, which will be created and upgraded by codes. Each line contains two numbers: the first is the index of iteration, and the second, ranging from 0 to 9, records which stage in each iteration is currently running.

Index of iterations	Stage in each iteration	Process
0	0	make_train
0	1	run_train
0	2	post_train
0	3	make_model_devi
0	4	run_model_devi
0	5	post_model_devi
0	6	make_fp
0	7	run_fp
0	8	post_fp

0,1,2 correspond to make_train, run_train, post_train. DP-GEN will write scripts in make_train, run the task by specific machine in run_train and collect result in post_train. The records for model_devi and fp stage follow similar rules.

If the process of DP-GEN stops for some reasons, DP-GEN will automatically recover the main process by record.dpgen. You may also change it manually for your purpose, such as removing the last iterations and recovering from one checkpoint. When re-running dpgen, the process will start from the stage that the last line record.

4.2 Example-of-param.json

We have provided different examples of param.json in dpgen/examples/run/. In this section, we give a description of the param.json, taking dpgen/examples/run/dp2.x-lammps-vasp/param_CH4_deepmd-kit-2.0.1.json as an example. This is a param.json for a gas-phase methane molecule. Here, DeePMD-kit (v2.x), LAMMPS and VASP codes are used for training, exploration and labeling respectively.

4.2.1 basics

The basics related keys in param.json are given as follows

```
"type_map": [
  "H",
  "C"
],
"mass_map": [
  1,
  12
],
```

The basics related keys specify the basic information about the system. *type_map* gives the atom types, i.e. “H” and “C”. *mass_map* gives the standard atom weights, i.e. “1” and “12”.

4.2.2 data

The data related keys in param.json are given as follows

```
"init_data_prefix": "...../init/",
"init_data_sys": [
  "CH4.POSCAR.01x01x01/02.md/sys-0004-0001/deepmd"
],

"sys_configs_prefix": "...../init/",
"sys_configs": [
  [
    "CH4.POSCAR.01x01x01/01.scale_pert/sys-0004-0001/scale*/00000*/POSCAR"
  ],
  [
    "CH4.POSCAR.01x01x01/01.scale_pert/sys-0004-0001/scale*/00001*/POSCAR"
  ]
],
```

The data related keys specify the init data for training initial DP models and structures used for model_devi calculations. *init_data_prefix* and *init_data_sys* specify the location of the init data. *sys_configs_prefix* and *sys_configs* specify the location of the structures.

Here, the init data is provided at “..... /init/CH4.POSCAR.01x01x01/02.md/sys-0004-0001/deepmd”. These structures are divided into two groups and provided at “...../init/CH4.POSCAR.01x01x01/01.scale_pert/sys-0004-0001/scale*/00000*/POSCAR” and “...../init/CH4.POSCAR.01x01x01/01.scale_pert/sys-0004-0001/scale*/00001*/POSCAR”.

4.2.3 training

The training related keys in param.json are given as follows

```
"numb_models": 4,
"default_training_param": {
  },
```

The training related keys specify the details of training tasks. `numb_models` specifies the number of models to be trained. “default_training_param” specifies the training parameters for deepmd-kit.

Here, 4 DP models will be trained in `00.train`. A detailed explanation of training parameters can be found in DeePMD-kit’s documentation (<https://docs.deepmodeling.com/projects/deepmd/en/master/>).

4.2.4 exploration

The exploration related keys in param.json are given as follows

```
"model_devi_dt": 0.002,
"model_devi_skip": 0,
"model_devi_f_trust_lo": 0.05,
"model_devi_f_trust_hi": 0.15,
"model_devi_clean_traj": true,
"model_devi_jobs": [
  {
    "sys_idx": [
      0
    ],
    "temps": [
      100
    ],
    "press": [
      1.0
    ],
    "trj_freq": 10,
    "nsteps": 300,
    "ensemble": "nvt",
    "_idx": "00"
  },
  {
    "sys_idx": [
      1
    ],
    "temps": [
      100
    ],
    "press": [
      1.0
    ],
    "trj_freq": 10,
    "nsteps": 3000,
    "ensemble": "nvt",
    "_idx": "01"
```

(continues on next page)

(continued from previous page)

```

    }
  ],

```

The exploration related keys specify the details of exploration tasks. `model_devi_dt` specifies timestep for MD simulation. `model_devi_skip` specifies the number of structures skipped for saving in each MD. `model_devi_f_trust_lo` and `model_devi_f_trust_hi` specify the lower and upper bound of model_devi of forces for the selection. `model_devi_clean_traj` specifies whether to clean traj folders in MD. If type of model_devi_clean_traj is boolean type then it denote whether to clean traj folders in MD since they are too large. In `model_devi_jobs`, `sys_idx` specifies the group of structures used for model_devi calculations, `temps` specifies the temperature (K) in MD, `press` specifies the pressure (Bar) in MD, `trj_freq` specifies the frequency of trajectory saved in MD, `nsteps` specifies the running steps of MD, `ensemble` specifies the ensemble used in MD, and “_idx” specifies the index of iteration.

Here, MD simulations are performed at the temperature of 100 K and the pressure of 1.0 Bar with an integrator time of 2 fs under the nvt ensemble. Two iterations are set in `model_devi_jobs`. MD simulations are run for 300 and 3000 time steps with the first and second groups of structures in `sys_configs` in 00 and 01 iterations. We choose to save all structures generated in MD simulations and have set `trj_freq` as 10, so 30 and 300 structures are saved in 00 and 01 iterations. If the “max_devi_f” of saved structure falls between 0.05 and 0.15, DP-GEN will treat the structure as a candidate. We choose to clean traj folders in MD since they are too large. If you want to save the most recent n iterations of traj folders, you can set `model_devi_clean_traj` to be an integer.

4.2.5 labeling

The labeling related keys in param.json are given as follows

```

"fp_style": "vasp",
"shuffle_poscar": false,
"fp_task_max": 20,
"fp_task_min": 1,
"fp_pp_path": "...../methane/",
"fp_pp_files": [
  "POTCAR"
],
"fp_incar": "...../INCAR_methane"

```

The labeling related keys specify the details of labeling tasks. `fp_style` specifies software for First Principles. `fp_task_max` and `fp_task_min` specify the minimum and maximum of structures to be calculated in 02.fp of each iteration. `fp_pp_path` and `fp_pp_files` specify the location of the psuedo-potential file to be used for 02.fp. `run_jdata[fp_style=vasp]/fp_incar` specifies input file for VASP. INCAR must specify KSPACING and KGAMMA.

Here, a minimum of 1 and a maximum of 20 structures will be labeled using the VASP code with the INCAR provided at “...../INCAR_methane” and POTCAR provided at “...../methane/POTCAR” in each iteration. Note that the order of elements in POTCAR should correspond to the order in `type_map`.

All the keys of the DP-GEN are explained in detail in the section Parameters.

4.3 Example of machine.json

4.3.1 DPDispatcher Update Note

DPDispatcher has updated and the api of machine.json is changed. DP-GEN will use the new DPDispatcher if the value of key `api_version` in machine.json is equal to or large than 1.0. And for now, DPDispatcher is maintained on a separate repo (<https://github.com/deepmodeling/dpdispatcher>). Please check the documents (<https://deepmd.readthedocs.io/projects/dpdispatcher/en/latest/>) for more information about the new DPDispatcher.

DP-GEN will use the old DPDispatcher if the key `api_version` is not specified in machine.json or the `api_version` is smaller than 1.0. This gurantees that the old machine.json still works.

4.3.2 New DPDispatcher

Each iteration in the run process of DP-GEN is composed of three steps: exploration, labeling, and training. Accordingly, machine.json is composed of three parts: train, model_devi, and fp. Each part is a list of dicts. Each dict can be considered as an independent environment for calculation.

In this section, we will show you how to perform train task at a local workstation, model_devi task at a local Slurm cluster, and fp task at a remote PBS cluster using the new DPDispatcher. For each task, three types of keys are needed:

- Command: provides the command used to execute each step.
- Machine: specifies the machine environment (local workstation, local or remote cluster, or cloud server).
- Resources: specify the number of groups, nodes, CPU, and GPU; enable the virtual environment.

Performing train task at a local workstation

In this example, we perform the train task on a local workstation.

```
"train":
{
  "command": "dp",
  "machine": {
    "batch_type": "Shell",
    "context_type": "local",
    "local_root": "./",
    "remote_root": "/home/user1234/work_path"
  },
  "resources": {
    "number_node": 1,
    "cpu_per_node": 4,
    "gpu_per_node": 1,
    "group_size": 1,
    "source_list": ["/home/user1234/deepmd.env"]
  }
},
```

The `command` for the train task in the DeepMD-kit is “dp”.

In machine parameters, `batch_type` specifies the type of job scheduling system. If there is no job scheduling system, we can use the “Shell” to perform the task. `context_type` specifies the method of data transfer, and “local” means copying and moving data via local file storage systems (e.g. cp, mv, etc.). In DP-GEN, the paths of all tasks are

automatically located and set by the software, and therefore `local_root` is always set to `./`. The input file for each task will be sent to the `remote_root` and the task will be performed there, so we need to make sure that the path exists.

In the resources parameter, `number_node`, `cpu_per_node`, and `gpu_per_node` specify the number of nodes, the number of CPUs, and the number of GPUs required for a task respectively. `group_size`, which needs to be highlighted, specifies how many tasks will be packed into a group. In the training tasks, we need to train 4 models. If we only have one GPU, we can set the `group_size` to 4. If `group_size` is set to 1, 4 models will be trained on one GPU at the same time, as there is no job scheduling system. Finally, the environment variables can be activated by `source_list`. In this example, `source /home/user1234/deepmd.env` is executed before `dp` to load the environment variables necessary to perform the training task.

Perform model_devi task at a local Slurm cluster

In this example, we perform the `model_devi` task at a local Slurm workstation.

```
"model_devi":
{
  "command": "lmp",
  "machine": {
    "context_type": "local",
    "batch_type": "Slurm",
    "local_root": "./",
    "remote_root": "/home/user1234/work_path"
  },
  "resources": {
    "number_node": 1,
    "cpu_per_node": 4,
    "gpu_per_node": 1,
    "queue_name": "QueueGPU",
    "custom_flags": ["#SBATCH --mem=32G"],
    "group_size": 10,
    "source_list": ["/home/user1234/lammps.env"]
  }
}
```

The `command` for the `model_devi` task in the LAMMPS is `lmp`.

In the machine parameter, we specify the type of job scheduling system by changing the `batch_type` to `Slurm`.

In the resources parameter, we specify the name of the queue to which the task is submitted by adding `queue_name`. We can add additional lines to the calculation script via the `custom_flags`. In the `model_devi` steps, there are frequently many short tasks, so we usually pack multiple tasks (e.g. 10) into a group for submission. Other parameters are similar to that of the local workstation.

Perform fp task in a remote PBS cluster

In this example, we perform the `fp` task at a remote PBS cluster that can be accessed via SSH.

```
"fp":
{
  "command": "mpirun -n 32 vasp_std",
  "machine": {
    "context_type": "SSHContext",
    "batch_type": "PBS",
```

(continues on next page)

(continued from previous page)

```

    "local_root": "./",
    "remote_root": "/home/user1234/work_path",
    "remote_profile": {
        "hostname": "39.xxx.xx.xx",
        "username": "user1234"
    }
},
"resources": {
    "number_node": 1,
    "cpu_per_node": 32,
    "gpu_per_node": 0,
    "queue_name": "QueueCPU",
    "group_size": 5,
    "source_list": ["/home/user1234/vasp.env"]
}
}

```

VASP code is used for fp task and mpi is used for parallel computing, so “mpirun -n 32” is added to specify the number of parallel threads.

In the machine parameter, *context_type* is modified to “SSHContext” and *batch_type* is modified to “PBS”. It is worth noting that *remote_root* should be set to an accessible path on the remote PBS cluster. *remote_profile* is added to specify the information used to connect the remote cluster, including hostname, username, port, etc.

In the resources parameter, we set *gpu_per_node* to 0 since it is cost-effective to use the CPU for VASP calculations.

Explicit descriptions of keys in machine.json will be given in the following section.

4.4 dpgen run param parameters

Note: One can load, modify, and export the input file by using our effective web-based tool [DP-GUI](#) online or hosted using the *command line interface* `dpgen gui`. All parameters below can be set in DP-GUI. By clicking “SAVE JSON”, one can download the input file.

run_jdata:

type: dict
 argument path: run_jdata
 param.json file

type_map:

type: list[str]
 argument path: run_jdata/type_map

Atom types. Reminder: The elements in param.json, type.raw and data.lmp(when using lammps) should be in the same order.

mass_map:

type: str | list[float], optional, default: auto
 argument path: run_jdata/mass_map

Standard atomic weights (default: “auto”). if one want to use isotopes, or non-standard element names, chemical symbols, or atomic number in the type_map list, please customize the mass_map list instead of using “auto”.

use_ele_temp:

type: int, optional, default: 0
argument path: run_jdata/use_ele_temp

Currently only support fp_style vasp.

- 0: no electron temperature.
- 1: electron temperature as frame parameter.
- 2: electron temperature as atom parameter.

init_data_prefix:

type: str, optional
argument path: run_jdata/init_data_prefix

Prefix of initial data directories.

init_data_sys:

type: list[str]
argument path: run_jdata/init_data_sys

Paths of initial data. The path can be either a system directory containing NumPy files or an HDF5 file. You may use either absolute or relative path here. Systems will be detected recursively in the directories or the HDF5 file.

sys_format:

type: str, optional, default: vasp/poscar
argument path: run_jdata/sys_format

Format of sys_configs.

init_batch_size:

type: str | list[typing.Union[int, str]], optional
argument path: run_jdata/init_batch_size

Each number is the batch_size of corresponding system for training in init_data_sys. One recommended rule for setting the sys_batch_size and init_batch_size is that batch_size multiply number of atoms of the structure should be larger than 32. If set to auto, batch size will be 32 divided by number of atoms. This argument will not override the mixed batch size in *default_training_param*.

sys_configs_prefix:

type: str, optional
argument path: run_jdata/sys_configs_prefix

Prefix of sys_configs.

sys_configs:

type: list[list[str]]
argument path: run_jdata/sys_configs

2D list. Containing directories of structures to be explored in iterations for each system. Wildcard characters are supported here.

sys_batch_size:

type: list[typing.Union[int, str]], optional

argument path: `run_jdata/sys_batch_size`

Each number is the `batch_size` for training of corresponding system in `sys_configs`. If set to `auto`, batch size will be 32 divided by number of atoms. This argument will not override the mixed batch size in *default_training_param*.

numb_models:

type: `int`

argument path: `run_jdata/numb_models`

Number of models to be trained in `00.train`. 4 is recommend.

training_iter0_model_path:

type: `list[str]`, optional

argument path: `run_jdata/training_iter0_model_path`

The model used to init the first iter training. Number of element should be equal to `numb_models`.

training_init_model:

type: `bool`, optional

argument path: `run_jdata/training_init_model`

Iteration > 0, the model parameters will be initilized from the model trained at the previous iteration. Iteration == 0, the model parameters will be initialized from `training_iter0_model_path`.

default_training_param:

type: `dict`

argument path: `run_jdata/default_training_param`

Training parameters for `deepmd-kit` in `00.train`. You can find instructions from [DeePMD-kit documentation](#).

dp_train_skip_neighbor_stat:

type: `bool`, optional, default: `False`

argument path: `run_jdata/dp_train_skip_neighbor_stat`

Append `-skip-neighbor-stat` flag to `dp train`.

dp_compress:

type: `bool`, optional, default: `False`

argument path: `run_jdata/dp_compress`

Use `dp compress` to compress the model.

training_reuse_iter:

type: `int | NoneType`, optional

argument path: `run_jdata/training_reuse_iter`

The minimal index of iteration that continues training models from old models of last iteration.

training_reuse_old_ratio:

type: `str | float`, optional, default: `auto`

argument path: `run_jdata/training_reuse_old_ratio`

The probability proportion of old data during training. It can be:

- `float`: directly assign the probability of old data;

- *auto:f*: automatic probability, where f is the new-to-old ratio;
- *auto*: equivalent to *auto:10*.

This option is only adopted when continuing training models from old models. This option will override default parameters.

training_reuse_num_steps:

type: int | NoneType, optional, default: None, alias: *training_reuse_stop_batch*
argument path: run_jdata/training_reuse_num_steps

Number of training batch. This option is only adopted when continuing training models from old models. This option will override default parameters.

training_reuse_start_lr:

type: float | NoneType, optional, default: None
argument path: run_jdata/training_reuse_start_lr

The learning rate the start of the training. This option is only adopted when continuing training models from old models. This option will override default parameters.

training_reuse_start_pref_e:

type: int | float | NoneType, optional, default: None
argument path: run_jdata/training_reuse_start_pref_e

The prefactor of energy loss at the start of the training. This option is only adopted when continuing training models from old models. This option will override default parameters.

training_reuse_start_pref_f:

type: int | float | NoneType, optional, default: None
argument path: run_jdata/training_reuse_start_pref_f

The prefactor of force loss at the start of the training. This option is only adopted when continuing training models from old models. This option will override default parameters.

model_devi_activation_func:

type: NoneType | list[list[str]], optional
argument path: run_jdata/model_devi_activation_func

The activation function in the model. The shape of list should be (N_models, 2), where 2 represents the embedding and fitting network. This option will override default parameters.

srtab_file_path:

type: str, optional
argument path: run_jdata/srtab_file_path

The path of the table for the short-range pairwise interaction which is needed when using DP-ZBL potential

one_h5:

type: bool, optional, default: False
argument path: run_jdata/one_h5

When using DeePMD-kit, all of the input data will be merged into one HDF5 file.

training_init_frozen_model:

type: list[str], optional

argument path: run_jdata/training_init_frozen_model

At iteration 0, initialize the model parameters from the given frozen models. Number of element should be equal to numb_models.

training_finetune_model:

type: list[str], optional

argument path: run_jdata/training_finetune_model

At iteration 0, finetune the model parameters from the given frozen models. Number of element should be equal to numb_models.

fp_task_max:

type: int

argument path: run_jdata/fp_task_max

Maximum number of structures to be calculated in each system in 02.fp of each iteration. If the number of candidate structures exceeds *fp_task_max*, *fp_task_max* structures will be randomly picked from the candidates and labeled.

fp_task_min:

type: int

argument path: run_jdata/fp_task_min

Skip the training in the next iteration if the number of structures is no more than *fp_task_min*.

fp_accurate_threshold:

type: float, optional

argument path: run_jdata/fp_accurate_threshold

If the accurate ratio is larger than this number, no fp calculation will be performed, i.e. *fp_task_max* = 0.

fp_accurate_soft_threshold:

type: float, optional

argument path: run_jdata/fp_accurate_soft_threshold

If the accurate ratio is between this number and *fp_accurate_threshold*, the *fp_task_max* linearly decays to zero.

fp_cluster_vacuum:

type: float, optional

argument path: run_jdata/fp_cluster_vacuum

If the vacuum size is smaller than this value, this cluster will not be chosen for labeling.

detailed_report_make_fp:

type: bool, optional, default: True

argument path: run_jdata/detailed_report_make_fp

If set to true, a detailed report will be generated for each iteration.

ratio_failed:

type: float, optional

argument path: `run_jdata/ratio_failed`

Check the ratio of unsuccessfully terminated jobs. If too many FP tasks are not converged, `RuntimeError` will be raised.

Depending on the value of `model_devi_engine`, different sub args are accepted.

model_devi_engine:

type: `str` (flag key), default: `lammps`

argument path: `run_jdata/model_devi_engine`

possible choices: `lammps`, `amber`, `calypso`, `gromacs`

Engine for the model deviation task.

When `model_devi_engine` is set to `lammps`:

LAMMPS

model_devi_jobs:

type: `list`

argument path: `run_jdata[model_devi_engine=lammps]/model_devi_jobs`

Settings for exploration in `01.model_devi`. Each dict in the list corresponds to one iteration. The index of `model_devi_jobs` exactly accord with index of iterations

This argument takes a list with each element containing the following:

template:

type: `dict`, optional

argument path:

`run_jdata[model_devi_engine=lammps]/model_devi_jobs/template`

Give an input file template for the supported engine software adopted in `01.model_devi`. Through user-defined template, any freedom (function) that is permitted by the engine software could be inherited (invoked) in the workflow.

lmp:

type: `str`, optional

argument path: `run_jdata[model_devi_engine=lammps]/model_devi_jobs/template/lmp`

The path to input.lammps template. Instructions can be found in [LAMMPS documentation](#).

plm:

type: `str`, optional

argument path: `run_jdata[model_devi_engine=lammps]/model_devi_jobs/template/plm`

The path to input.plumed template. Instructions can be found in [PLUMED documentation](#).

rev_mat:

type: `dict`, optional

argument path: `run_jdata[model_devi_engine=lammps]/model_devi_jobs/rev_mat`

revise matrix for revising variable(s) defined in the template into the specific values (iteration-resolved). Values will be broadcasted for all tasks within the iteration invoking this key.

lmp:

type: dict, optional

argument path:

run_jdata[model_devi_engine=lammps]/
model_devi_jobs/rev_mat/lmp

revise matrix for revising variable(s) defined in the lammps template into the specific values (iteration-resolved).

plm:

type: dict, optional

argument path:

run_jdata[model_devi_engine=lammps]/
model_devi_jobs/rev_mat/plm

revise matrix for revising variable(s) defined in the plumed template into specific values(iteration-resolved)

sys_rev_mat:

type: dict, optional

argument path: run_jdata[model_devi_engine=lammps]/
model_devi_jobs/sys_rev_mat

system-resolved revise matrix for revising variable(s) defined in the template into specific values. Values should be individually assigned to each system adopted by this iteration, through a dictionary where first-level keys are values of sys_idx of this iteration.

sys_idx:

type: list[int]

argument path: run_jdata[model_devi_engine=lammps]/
model_devi_jobs/sys_idx

Systems to be selected as the initial structure of MD and be explored. The index corresponds exactly to the sys_configs.

temps:

type: list[float], optional

argument path: run_jdata[model_devi_engine=lammps]/
model_devi_jobs/temps

Temperature (K) in MD.

press:

type: list[float], optional

argument path: run_jdata[model_devi_engine=lammps]/
model_devi_jobs/press

Pressure (Bar) in MD. Required when ensemble is npt.

trj_freq:

type: int

argument path: run_jdata[model_devi_engine=lammps]/
model_devi_jobs/trj_freq

Frequency of trajectory saved in MD.

nsteps:

type: int, optional
 argument path: `run_jdata[model_devi_engine=lammps]/model_devi_jobs/nsteps`
 Running steps of MD. It is not optional when not using a template.

nbeads:

type: int, optional
 argument path: `run_jdata[model_devi_engine=lammps]/model_devi_jobs/nbeads`
 Number of beads in PIMD. If not given, classical MD will be performed. Only supported for LAMMPS version ≥ 20230615 .

ensemble:

type: str, optional
 argument path: `run_jdata[model_devi_engine=lammps]/model_devi_jobs/ensemble`
 Determining which ensemble used in MD, options include “npt” and “nvt”. It is not optional when not using a template.

neidelay:

type: int, optional
 argument path: `run_jdata[model_devi_engine=lammps]/model_devi_jobs/neidelay`
 delay building until this many steps since last build.

taut:

type: float, optional
 argument path: `run_jdata[model_devi_engine=lammps]/model_devi_jobs/taut`
 Coupling time of thermostat (ps).

taup:

type: float, optional
 argument path: `run_jdata[model_devi_engine=lammps]/model_devi_jobs/taup`
 Coupling time of barostat (ps).

model_devi_f_trust_lo:

type: dict | float, optional
 argument path: `run_jdata[model_devi_engine=lammps]/model_devi_jobs/model_devi_f_trust_lo`
 Lower bound of forces for the selection. If dict, should be set for each index in `sys_idx`, respectively.

model_devi_f_trust_hi:

type: dict | float, optional
argument path: run_jdata[model_devi_engine=lammps]/
model_devi_jobs/model_devi_f_trust_hi

Upper bound of forces for the selection. If dict, should be set for each index in sys_idx, respectively.

model_devi_v_trust_lo:

type: dict | float, optional
argument path: run_jdata[model_devi_engine=lammps]/
model_devi_jobs/model_devi_v_trust_lo

Lower bound of virial for the selection. If dict, should be set for each index in sys_idx, respectively. Should be used with DeePMD-kit v2.x.

model_devi_v_trust_hi:

type: dict | float, optional
argument path: run_jdata[model_devi_engine=lammps]/
model_devi_jobs/model_devi_v_trust_hi

Upper bound of virial for the selection. If dict, should be set for each index in sys_idx, respectively. Should be used with DeePMD-kit v2.x.

model_devi_dt:

type: float
argument path: run_jdata[model_devi_engine=lammps]/model_devi_dt

Timestep for MD. 0.002 is recommend.

model_devi_skip:

type: int
argument path:
run_jdata[model_devi_engine=lammps]/model_devi_skip

Number of structures skipped for fp in each MD.

model_devi_f_trust_lo:

type: dict | list[float] | float
argument path:
run_jdata[model_devi_engine=lammps]/model_devi_f_trust_lo

Lower bound of forces for the selection. If list or dict, should be set for each index in sys_configs, respectively.

model_devi_f_trust_hi:

type: dict | list[float] | float
argument path:
run_jdata[model_devi_engine=lammps]/model_devi_f_trust_hi

Upper bound of forces for the selection. If list or dict, should be set for each index in sys_configs, respectively.

model_devi_v_trust_lo:

type: dict | list[float] | float, optional, default: 10000000000.0

argument path:

run_jdata[model_devi_engine=lammps]/model_devi_v_trust_lo

Lower bound of virial for the selection. If list or dict, should be set for each index in sys_configs, respectively. Should be used with DeePMD-kit v2.x.

model_devi_v_trust_hi:

type: dict | list[float] | float, optional, default: 10000000000.0

argument path:

run_jdata[model_devi_engine=lammps]/model_devi_v_trust_hi

Upper bound of virial for the selection. If list or dict, should be set for each index in sys_configs, respectively. Should be used with DeePMD-kit v2.x.

model_devi_adapt_trust_lo:

type: bool, optional

argument path:

run_jdata[model_devi_engine=lammps]/model_devi_adapt_trust_lo

Adaptively determines the lower trust levels of force and virial. This option should be used together with model_devi_numb_candi_f, model_devi_numb_candi_v and optionally with model_devi_perc_candi_f and model_devi_perc_candi_v. dpngen will make two sets:

- 1. From the frames with force model deviation lower than model_devi_f_trust_hi, select max(model_devi_numb_candi_f, model_devi_perc_candi_f*n_frames) frames with largest force model deviation.
- 2. From the frames with virial model deviation lower than model_devi_v_trust_hi, select max(model_devi_numb_candi_v, model_devi_perc_candi_v*n_frames) frames with largest virial model deviation.

The union of the two sets is made as candidate dataset.

model_devi_numb_candi_f:

type: int, optional

argument path:

run_jdata[model_devi_engine=lammps]/model_devi_numb_candi_f

See model_devi_adapt_trust_lo.

model_devi_numb_candi_v:

type: int, optional

argument path:

run_jdata[model_devi_engine=lammps]/model_devi_numb_candi_v

See model_devi_adapt_trust_lo.

model_devi_perc_candi_f:

type: float, optional

argument path:
run_jdata[model_devi_engine=lammps]/model_devi_perc_candi_f
See model_devi_adapt_trust_lo.

model_devi_perc_candi_v:

type: float, optional
argument path:
run_jdata[model_devi_engine=lammps]/model_devi_perc_candi_v
See model_devi_adapt_trust_lo.

model_devi_f_avg_relative:

type: bool, optional
argument path:
run_jdata[model_devi_engine=lammps]/model_devi_f_avg_relative
Normalized the force model deviations by the RMS force magnitude along the trajectory. This key should not be used with use_relative.

model_devi_clean_traj:

type: bool | int, optional, default: True
argument path:
run_jdata[model_devi_engine=lammps]/model_devi_clean_traj
If type of model_devi_clean_traj is bool type then it denote whether to clean traj folders in MD since they are too large. If it is Int type, then the most recent n iterations of traj folders will be retained, others will be removed.

model_devi_merge_traj:

type: bool, optional, default: False
argument path:
run_jdata[model_devi_engine=lammps]/model_devi_merge_traj
If model_devi_merge_traj is set as True, only all.lammpstrj will be generated, instead of lots of small traj files.

model_devi_nopbc:

type: bool, optional, default: False
argument path:
run_jdata[model_devi_engine=lammps]/model_devi_nopbc
Assume open boundary condition in MD simulations.

model_devi_plumed:

type: bool, optional, default: False
argument path:
run_jdata[model_devi_engine=lammps]/model_devi_plumed

model_devi_plumed_path:

type: bool, optional, default: False
argument path:
run_jdata[model_devi_engine=lammps]/model_devi_plumed_path

shuffle_poscar:

type: bool, optional, default: False

argument path: `run_jdata[model_devi_engine=lammps]/shuffle_poscar`

Shuffle atoms of each frame before running simulations. The purpose is to sample the element occupation of alloys.

use_relative:

type: bool, optional, default: False

argument path: `run_jdata[model_devi_engine=lammps]/use_relative`

Calculate relative force model deviation.

epsilon:

type: float, optional

argument path: `run_jdata[model_devi_engine=lammps]/epsilon`

The level parameter for computing the relative force model deviation.

use_relative_v:

type: bool, optional, default: False

argument path: `run_jdata[model_devi_engine=lammps]/use_relative_v`

Calculate relative virial model deviation.

epsilon_v:

type: float, optional

argument path: `run_jdata[model_devi_engine=lammps]/epsilon_v`

The level parameter for computing the relative virial model deviation.

When `model_devi_engine` is set to `amber`:

Amber DPRc engine. The command argument in the machine file should be path to `sander`.

model_devi_jobs:

type: list

argument path: `run_jdata[model_devi_engine=amber]/model_devi_jobs`

List of dicts. The list including the dict for information of each cycle.

This argument takes a list with each element containing the following:

sys_idx:

type: list[int]

argument path: `run_jdata[model_devi_engine=amber]/model_devi_jobs/sys_idx`

List of ints. List of systems to run.

trj_freq:

type: int

argument path: `run_jdata[model_devi_engine=amber]/model_devi_jobs/trj_freq`

Frequency to dump trajectory.

restart_from_iter:

type: int, optional

argument path: run_jdata[model_devi_engine=amber]/
model_devi_jobs/restart_from_iter

The iteration index to restart the simulation from. If not given, the simulation is restarted from *sys_configs*.

low_level:

type: str

argument path: run_jdata[model_devi_engine=amber]/low_level

Low level method. The value will be filled into mdin file as @qm_theory@.

cutoff:

type: float

argument path: run_jdata[model_devi_engine=amber]/cutoff

Cutoff radius for the DPRc model.

parm7_prefix:

type: str, optional

argument path: run_jdata[model_devi_engine=amber]/parm7_prefix

The path prefix to AMBER PARM7 files.

parm7:

type: list[str]

argument path: run_jdata[model_devi_engine=amber]/parm7

List of paths to AMBER PARM7 files. Each file maps to a system.

mdin_prefix:

type: str, optional

argument path: run_jdata[model_devi_engine=amber]/mdin_prefix

The path prefix to AMBER mdin template files.

mdin:

type: list[str]

argument path: run_jdata[model_devi_engine=amber]/mdin

List of paths to AMBER mdin template files. Each files maps to a system. In the template, the following keywords will be replaced by the actual value: @freq@: freq to dump trajectory; @nstlim@: total time step to run; @qm_region@: AMBER mask of the QM region; @qm_theory@: The low level QM theory, such as DFTB2; @qm_charge@: The total charge of the QM theory, such as -2; @rcut@: cutoff radius of the DPRc model; @GRAPH_FILE0@, @GRAPH_FILE1@, ... : graph files.

qm_region:

type: list[str]

argument path: run_jdata[model_devi_engine=amber]/qm_region

List of strings. AMBER mask of the QM region. Each mask maps to a system.

qm_charge:

type: `list[int]`
 argument path: `run_jdata[model_devi_engine=amber]/qm_charge`
 List of ints. Charge of the QM region. Each charge maps to a system.

nsteps:

type: `list[int]`
 argument path: `run_jdata[model_devi_engine=amber]/nsteps`
 List of ints. The number of steps to run. Each number maps to a system.

r:

type: `list[list[typing.Union[float, list[float]]]]`
 argument path: `run_jdata[model_devi_engine=amber]/r`
 2D or 3D list of floats. Constrict values for the enhanced sampling. The first dimension maps to systems. The second dimension maps to confs in each system. The third dimension is the constrict value. It can be a single float for 1D or list of floats for nD.

disang_prefix:

type: `str`, optional
 argument path: `run_jdata[model_devi_engine=amber]/disang_prefix`
 The path prefix to disang prefix.

disang:

type: `list[str]`
 argument path: `run_jdata[model_devi_engine=amber]/disang`
 List of paths to AMBER disang files. Each file maps to a system. The keyword RVAL will be replaced by the constrict values, or RVAL1, RVAL2, ... for an nD system.

model_devi_f_trust_lo:

type: `dict | list[float] | float`
 argument path:
`run_jdata[model_devi_engine=amber]/model_devi_f_trust_lo`
 Lower bound of forces for the selection. If dict, should be set for each index in `sys_idx`, respectively.

model_devi_f_trust_hi:

type: `dict | list[float] | float`
 argument path:
`run_jdata[model_devi_engine=amber]/model_devi_f_trust_hi`
 Upper bound of forces for the selection. If dict, should be set for each index in `sys_idx`, respectively.

When `model_devi_engine` is set to `calypso`:

TODO: add doc

When `model_devi_engine` is set to `gromacs`:

TODO: add doc

Depending on the value of *fp_style*, different sub args are accepted.

fp_style:

type: str (flag key)
argument path: run_jdata/fp_style
possible choices: *vasp*, *gaussian*, *siesta*, *cp2k*, *abacus*, *amber/diff*,
pwmat, *pwscf*, *custom*
Software for First Principles.

When *fp_style* is set to *vasp*:

fp_pp_path:

type: str
argument path: run_jdata[fp_style=vasp]/fp_pp_path
Directory of psuedo-potential file to be used for 02.fp exists.

fp_pp_files:

type: list[str]
argument path: run_jdata[fp_style=vasp]/fp_pp_files
Psuedo-potential file to be used for 02.fp. Note that the order of elements should correspond to the order in type_map.

fp_incar:

type: str
argument path: run_jdata[fp_style=vasp]/fp_incar
Input file for VASP. INCAR must specify KSPACING and KGAMMA.

fp_aniso_kspacing:

type: list[float], optional
argument path: run_jdata[fp_style=vasp]/fp_aniso_kspacing
Set anisotropic kspacing. Usually useful for 1-D or 2-D materials. Only support VASP. If it is setting the KSPACING key in INCAR will be ignored.

cvasp:

type: bool, optional
argument path: run_jdata[fp_style=vasp]/cvasp
If cvasp is true, DP-GEN will use Custodian to help control VASP calculation.

fp_skip_bad_box:

type: str, optional
argument path: run_jdata[fp_style=vasp]/fp_skip_bad_box
Skip the configurations that are obviously unreasonable before 02.fp

When *fp_style* is set to *gaussian*:

use_clusters:

type: bool, optional, default: False

argument path: `run_jdata[fp_style=gaussian]/use_clusters`

If set to true, clusters will be taken instead of the whole system.

cluster_cutoff:

type: float, optional

argument path: `run_jdata[fp_style=gaussian]/cluster_cutoff`

The soft cutoff radius of clusters if *use_clusters* is set to true. Molecules will be taken as whole even if part of atoms is out of the cluster. Use *cluster_cutoff_hard* to only take atoms within the hard cutoff radius.

cluster_cutoff_hard:

type: float, optional

argument path: `run_jdata[fp_style=gaussian]/cluster_cutoff_hard`

The hard cutoff radius of clusters if *use_clusters* is set to true. Outside the hard cutoff radius, atoms will not be taken even if they are in a molecule where some atoms are within the cutoff radius.

cluster_minify:

type: bool, optional, default: False

argument path: `run_jdata[fp_style=gaussian]/cluster_minify`

If enabled, when an atom within the soft cutoff radius connects a single bond with a non-hydrogen atom out of the soft cutoff radius, the outer atom will be replaced by a hydrogen atom. When the outer atom is a hydrogen atom, the outer atom will be kept. In this case, other atoms out of the soft cutoff radius will be removed.

fp_params:

type: dict

argument path: `run_jdata[fp_style=gaussian]/fp_params`

Parameters for Gaussian calculation.

keywords:

type: str | list[str]

argument path:

`run_jdata[fp_style=gaussian]/fp_params/keywords`

Keywords for Gaussian input, e.g. `force b3lyp/6-31g**`. If a list, run multiple steps.

multiplicity:

type: str | int, optional, default: auto

argument path:

`run_jdata[fp_style=gaussian]/fp_params/multiplicity`

Spin multiplicity for Gaussian input. If *auto*, multiplicity will be detected automatically, with the following rules: when *fragment_guesses=True*, multiplicity will +1 for each radical, and +2 for each oxygen molecule; when *fragment_guesses=False*, multiplicity will be 1 or 2, but +2 for each oxygen molecule.

nproc:

type: int
argument path:
run_jdata[fp_style=gaussian]/fp_params/nproc

The number of processors for Gaussian input.

charge:

type: int, optional, default: 0
argument path:
run_jdata[fp_style=gaussian]/fp_params/charge

Molecule charge. Only used when charge is not provided by the system.

fragment_guesses:

type: bool, optional, default: False
argument path: run_jdata[fp_style=gaussian]/fp_params/
fragment_guesses

Initial guess generated from fragment guesses. If True, *multiplicity* should be *auto*.

basis_set:

type: str, optional
argument path:
run_jdata[fp_style=gaussian]/fp_params/basis_set

Custom basis set.

keywords_high_multiplicity:

type: str, optional
argument path: run_jdata[fp_style=gaussian]/fp_params/
keywords_high_multiplicity

Keywords for points with multiple raicals. *multiplicity* should be *auto*. If not set, fallback to normal keywords.

When `fp_style` is set to `siesta`:

use_clusters:

type: bool, optional
argument path: run_jdata[fp_style=siesta]/use_clusters

If set to true, clusters will be taken instead of the whole system. This option does not work with DeePMD-kit 0.x.

cluster_cutoff:

type: float, optional
argument path: run_jdata[fp_style=siesta]/cluster_cutoff

The cutoff radius of clusters if `use_clusters` is set to true.

fp_params:

type: dict

argument path: `run_jdata[fp_style=siesta]/fp_params`

Parameters for siesta calculation.

ecut:

type: int

argument path:

`run_jdata[fp_style=siesta]/fp_params/ecut`

Define the plane wave cutoff for grid.

ediff:

type: float

argument path:

`run_jdata[fp_style=siesta]/fp_params/ediff`

Tolerance of Density Matrix.

kspacing:

type: float

argument path:

`run_jdata[fp_style=siesta]/fp_params/kspacing`

Sample factor in Brillouin zones.

mixingWeight:

type: float

argument path:

`run_jdata[fp_style=siesta]/fp_params/mixingWeight`

Proportion a of output Density Matrix to be used for the input Density Matrix of next SCF cycle (linear mixing).

NumberPulay:

type: int

argument path:

`run_jdata[fp_style=siesta]/fp_params/NumberPulay`

Controls the Pulay convergence accelerator.

fp_pp_path:

type: str

argument path: `run_jdata[fp_style=siesta]/fp_pp_path`

Directory of psuedo-potential or numerical orbital files to be used for 02.fp exists.

fp_pp_files:

type: list[str]

argument path: `run_jdata[fp_style=siesta]/fp_pp_files`

Psuedo-potential file to be used for 02.fp. Note that the order of elements should correspond to the order in type_map.

When `fp_style` is set to `cp2k`:

user_fp_params:

type: dict, optional, alias: `fp_params`

argument path: `run_jdata[fp_style=cp2k]/user_fp_params`

Parameters for cp2k calculation. find detail in manual.cp2k.org. only the kind section must be set before use. we assume that you have basic knowledge for cp2k input.

external_input_path:

type: str, optional

argument path: `run_jdata[fp_style=cp2k]/external_input_path`

Conflict with key: `user_fp_params`. enable the template input provided by user. some rules should be followed, read the following text in detail:

1. One must present a KEYWORD ABC in the section CELL so that the DP-GEN can replace the cell on-the-fly.
2. One need to add these lines under FORCE_EVAL section to print forces and stresses:

```
STRESS_TENSOR ANALYTICAL
&PRINT
  &FORCES ON
  &END FORCES
  &STRESS_TENSOR ON
  &END STRESS_TENSOR
&END PRINT
```

When `fp_style` is set to `abacus`:

fp_pp_path:

type: str

argument path: `run_jdata[fp_style=abacus]/fp_pp_path`

Directory of psuedo-potential or numerical orbital files to be used for 02.fp exists.

fp_pp_files:

type: list[str]

argument path: `run_jdata[fp_style=abacus]/fp_pp_files`

Psuedo-potential file to be used for 02.fp. Note that the order of elements should correspond to the order in `type_map`.

fp_orb_files:

type: list[str], optional

argument path: `run_jdata[fp_style=abacus]/fp_orb_files`

numerical orbital file to be used for 02.fp when using LCAO basis. Note that the order of elements should correspond to the order in `type_map`.

fp_incar:

type: str, optional

argument path: `run_jdata[fp_style=abacus]/fp_incar`

Input file for ABACUS. This is optional but the priority is lower than `user_fp_params`, and you should not set `user_fp_params` if you want to use `fp_incar`.

fp_kpt_file:

type: str, optional

argument path: `run_jdata[fp_style=abacus]/fp_kpt_file`

KPT file for ABACUS. If the “kspacing” or “gamma_only=1” is defined in INPUT or “k_points” is defined, `fp_kpt_file` will be ignored.

fp_dpks_descriptor:

type: str, optional

argument path: `run_jdata[fp_style=abacus]/fp_dpks_descriptor`

DeePKS descriptor file name. The file should be in pseudopotential directory.

user_fp_params:

type: dict, optional

argument path: `run_jdata[fp_style=abacus]/user_fp_params`

Set the key and value of INPUT.

k_points:

type: list[int], optional

argument path: `run_jdata[fp_style=abacus]/k_points`

Monkhorst-Pack k-grids setting for generating KPT file of ABACUS, such as: [1,1,1,0,0,0]. NB: if “kspacing” or “gamma_only=1” is defined in INPUT, `k_points` will be ignored.

When `fp_style` is set to `amber/diff`:

Amber/diff style for DPRc models. Note: this fp style only supports to be used with `model_devi_engine amber`, where some arguments are reused. The command argument in the machine file should be path to sander. One should also install `dpamber` and make it visible in the PATH.

high_level:

type: str

argument path: `run_jdata[fp_style=amber/diff]/high_level`

High level method. The value will be filled into `mdin` template as `@qm_theory@`.

fp_params:

type: dict

argument path: `run_jdata[fp_style=amber/diff]/fp_params`

Parameters for FP calculation.

high_level_mdin:

type: str

argument path: `run_jdata[fp_style=amber/diff]/
fp_params/high_level_mdin`

Path to high-level AMBER mdin template file. `%qm_theory%`,
`%qm_region%`, and `%qm_charge%` will be replaced.

low_level_mdin:

type: str
argument path: `run_jdata[fp_style=amber/diff]/
fp_params/low_level_mdin`

Path to low-level AMBER mdin template file. `%qm_theory%`,
`%qm_region%`, and `%qm_charge%` will be replaced.

When `fp_style` is set to `pwmat`:

TODO: add doc

When `fp_style` is set to `pwscf`:

fp_pp_path:

type: str
argument path: `run_jdata[fp_style=pwscf]/fp_pp_path`
Directory of psuedo-potential file to be used for 02.fp exists.

fp_pp_files:

type: list[str]
argument path: `run_jdata[fp_style=pwscf]/fp_pp_files`
Psuedo-potential file to be used for 02.fp. Note that the order of elements should
correspond to the order in `type_map`.

fp_params:

type: dict, optional
argument path: `run_jdata[fp_style=pwscf]/fp_params`
Parameters for pwscf calculation. It has lower priority than `user_fp_params`.

ecut:

type: float
argument path:
`run_jdata[fp_style=pwscf]/fp_params/ecut`
`ecutwfc` in pwscf.

ediff:

type: float
argument path:
`run_jdata[fp_style=pwscf]/fp_params/ediff`
`conv_thr` and `ts_vdw_econv_thr` in pwscf.

smearing:

type: str

argument path:
`run_jdata[fp_style=pwscf]/fp_params/smearing`
 smearing in pwscf.

sigma:

type: float
 argument path:
`run_jdata[fp_style=pwscf]/fp_params/sigma`
 degauss in pwscf.

kspacing:

type: float
 argument path:
`run_jdata[fp_style=pwscf]/fp_params/kspacing`
 The spacing between kpoints. Helps to determin KPOINTS in pwscf.

user_fp_params:

type: dict, optional
 argument path: `run_jdata[fp_style=pwscf]/user_fp_params`
 Parameters for pwscf calculation. Find details at https://www.quantum-espresso.org/Doc/INPUT_PW.html. When `user_fp_params` is set, the settings in `fp_params` will be ignored. If one wants to use `user_fp_params`, `kspacing` must be set in `user_fp_params`. `kspacing` is the spacing between kpoints, and helps to determin KPOINTS in pwscf.

When `fp_style` is set to custom:

Custom FP code. You need to provide the input and output file format and name. The command argument in the machine file should be the script to run custom FP codes. The extra forward and backward files can be defined in the machine file.

fp_params:

type: dict
 argument path: `run_jdata[fp_style=custom]/fp_params`
 Parameters for FP calculation.

input_fmt:

type: str
 argument path:
`run_jdata[fp_style=custom]/fp_params/input_fmt`
 Input dpdata format of the custom FP code. Such format should only need the first argument as the file name.

input_fn:

type: str
 argument path:
`run_jdata[fp_style=custom]/fp_params/input_fn`
 Input file name of the custom FP code.

output_fmt:

type: str
argument path:
run_jdata[fp_style=custom]/fp_params/output_fmt
Output dpdata format of the custom FP code. Such format should only need the first argument as the file name.

output_fn:

type: str
argument path:
run_jdata[fp_style=custom]/fp_params/output_fn
Output file name of the custom FP code.

4.5 dpgen run machine parameters

Note: One can load, modify, and export the input file by using our effective web-based tool [DP-GUI](#) online or hosted using the *command line interface* `dpgen gui`. All parameters below can be set in DP-GUI. By clicking “SAVE JSON”, one can download the input file.

run_mdata:

type: dict
argument path: run_mdata
machine.json file

api_version:

type: str, optional, default: 1.0
argument path: run_mdata/api_version
Please set to 1.0

deepmd_version:

type: str, optional, default: 2
argument path: run_mdata/deepmd_version
DeePMD-kit version, e.g. 2.1.3

train:

type: dict
argument path: run_mdata/train
Parameters of command, machine, and resources for train

command:

type: str
argument path: run_mdata/train/command
Command of a program.

machine:

type: dict

argument path: run_mdata/train/machine

batch_type:

type: str

argument path:

run_mdata/train/machine/batch_type

The batch job system type. Option: Bohrium, PBS, OpenAPI, DistributedShell, LSF, SlurmJobArray, Slurm, SGE, Fugaku, Torque, Shell

local_root:

type: str | NoneType

argument path:

run_mdata/train/machine/local_root

The dir where the tasks and relating files locate. Typically the project dir.

remote_root:

type: str | NoneType, optional

argument path:

run_mdata/train/machine/remote_root

The dir where the tasks are executed on the remote machine. Only needed when context is not lazy-local.

clean_asynchronously:

type: bool, optional, default: False

argument path: run_mdata/train/machine/
clean_asynchronously

Clean the remote directory asynchronously after the job finishes.

Depending on the value of *context_type*, different sub args are accepted.

context_type:

type: str (flag key)

argument path:

run_mdata/train/machine/context_type

possible choices: *SSHContext*, *LazyLocalContext*, *HDFSContext*, *BohriumContext*, *OpenAPIContext*, *LocalContext*

The connection used to remote machine. Option: LocalContext, SSHContext, OpenAPIContext, BohriumContext, LazyLocalContext, HDFSContext

When *context_type* is set to *SSHContext* (or its aliases *sshcontext*, *SSH*, *ssh*):

remote_profile:

type: dict

argument path: `run_mdata/train/
machine[SSHContext]/remote_profile`

The information used to maintain the connection with remote machine.

hostname:

type: str
argument path: `run_mdata/train/machine[SSHContext]/
remote_profile/hostname`

hostname or ip of ssh connection.

username:

type: str
argument path: `run_mdata/train/
machine[SSHContext]/remote_profile/
username`

username of target linux system

password:

type: str, optional
argument path: `run_mdata/train/
machine[SSHContext]/remote_profile/
password`

(deprecated) password of linux system. Please use [SSH keys](#) instead to improve security.

port:

type: int, optional, default: 22
argument path: `run_mdata/train/
machine[SSHContext]/remote_profile/
port`

ssh connection port.

key_filename:

type: str | NoneType, optional, default: None
argument path: `run_mdata/train/
machine[SSHContext]/remote_profile/
key_filename`

key filename used by ssh connection. If left None, find key in `~/.ssh` or use password for login

passphrase:

type: str | NoneType, optional, default: None
argument path: `run_mdata/train/
machine[SSHContext]/remote_profile/
passphrase`

passphrase of key used by ssh connection

timeout:

type: int, optional, default: 10
 argument path: run_mdata/train/
 machine[SSHContext]/remote_profile/
 timeout
 timeout of ssh connection

totp_secret:

type: str | NoneType, optional, default: None
 argument path: run_mdata/train/
 machine[SSHContext]/remote_profile/
 totp_secret
 Time-based one time password secret. It
 should be a base32-encoded string extracted
 from the 2D code.

tar_compress:

type: bool, optional, default: True
 argument path: run_mdata/train/
 machine[SSHContext]/remote_profile/
 tar_compress
 The archive will be compressed in upload and
 download if it is True. If not, compression will
 be skipped.

look_for_keys:

type: bool, optional, default: True
 argument path: run_mdata/train/
 machine[SSHContext]/remote_profile/
 look_for_keys
 enable searching for discoverable private key
 files in ~/.ssh/

When `context_type` is set to `LazyLocalContext` (or its aliases
`lazylocalcontext`, `LazyLocal`, `lazylocal`):

remote_profile:

type: dict, optional
 argument path: run_mdata/train/
 machine[LazyLocalContext]/
 remote_profile
 The information used to maintain the connection
 with remote machine. This field is empty for this
 context.

When `context_type` is set to `HDFSContext` (or its aliases
`hdfscontext`, `HDFS`, `hdfs`):

remote_profile:

type: dict, optional

argument path: run_mdata/train/
machine[HDFSContext]/remote_profile

The information used to maintain the connection with remote machine. This field is empty for this context.

When `context_type` is set to `BohriumContext` (or its aliases `bohriumcontext`, `Bohrium`, `bohrium`, `DpCloudServerContext`, `dpcloudservercontext`, `DpCloudServer`, `dpcloudserver`, `LebesgueContext`, `lebesguecontext`, `Lebesgue`, `lebesgue`):

remote_profile:

type: dict
argument path: run_mdata/train/
machine[BohriumContext]/remote_profile

The information used to maintain the connection with remote machine.

email:

type: str, optional
argument path: run_mdata/train/
machine[BohriumContext]/
remote_profile/email

Email

password:

type: str, optional
argument path: run_mdata/train/
machine[BohriumContext]/
remote_profile/password

Password

program_id:

type: int, alias: *project_id*
argument path: run_mdata/train/
machine[BohriumContext]/
remote_profile/program_id

Program ID

retry_count:

type: NoneType | int, optional, default:
2
argument path: run_mdata/train/
machine[BohriumContext]/
remote_profile/retry_count

The retry count when a job is terminated

ignore_exit_code:

type: bool, optional, default: True

argument path: run_mdata/train/
 machine[BohriumContext]/
 remote_profile/ignore_exit_code

The job state will be marked as finished if the exit code is non-zero when set to True. Otherwise,
 the job state will be designated as terminated.

keep_backup:

type: bool, optional
 argument path: run_mdata/train/
 machine[BohriumContext]/
 remote_profile/keep_backup
 keep download and upload zip

input_data:

type: dict
 argument path: run_mdata/train/
 machine[BohriumContext]/
 remote_profile/input_data
 Configuration of job

When `context_type` is set to `OpenAPIContext` (or its aliases `openapicontext`, `OpenAPI`, `openapi`):

remote_profile:

type: dict, optional
 argument path: run_mdata/train/
 machine[OpenAPIContext]/remote_profile
 The information used to maintain the connection with remote machine. This field is empty for this context.

When `context_type` is set to `LocalContext` (or its aliases `localcontext`, `Local`, `local`):

remote_profile:

type: dict, optional
 argument path: run_mdata/train/
 machine[LocalContext]/remote_profile
 The information used to maintain the connection with remote machine. This field is empty for this context.

resources:

type: dict
 argument path: run_mdata/train/resources

number_node:

type: int, optional, default: 1

argument path: `run_mdata/train/resources/number_node`

The number of node need for each *job*

cpu_per_node:

type: int, optional, default: 1

argument path: `run_mdata/train/resources/cpu_per_node`

cpu numbers of each node assigned to each job.

gpu_per_node:

type: int, optional, default: 0

argument path: `run_mdata/train/resources/gpu_per_node`

gpu numbers of each node assigned to each job.

queue_name:

type: str, optional, default: (empty string)

argument path: `run_mdata/train/resources/queue_name`

The queue name of batch job scheduler system.

group_size:

type: int

argument path: `run_mdata/train/resources/group_size`

The number of *tasks* in a *job*. 0 means infinity.

custom_flags:

type: `typing.List[str]`, optional

argument path: `run_mdata/train/resources/custom_flags`

The extra lines pass to job submitting script header

strategy:

type: dict, optional

argument path: `run_mdata/train/resources/strategy`

strategies we use to generation job submitting scripts.

if_cuda_multi_devices:

type: bool, optional, default: False

argument path: `run_mdata/train/resources/strategy/if_cuda_multi_devices`

If there are multiple nvidia GPUS on the node, and we want to assign the tasks to different GPUS. If true, dpdispatcher will manually export environment variable CUDA_VISIBLE_DEVICES to different task. Usually, this option will be used with Task.task_need_resources variable simultaneously.

ratio_unfinished:

type: float, optional, default: 0.0

argument path:

run_mdata/train/resources/
strategy/ratio_unfinished

The ratio of *tasks* that can be unfinished.

customized_script_header_template_file:

type: str, optional

argument path: run_mdata/train/
resources/strategy/
customized_script_header_template_file

The customized template file to generate job submitting script header, which overrides the default file.

para_deg:

type: int, optional, default: 1

argument path:

run_mdata/train/resources/para_deg

Decide how many tasks will be run in parallel.

source_list:

type: typing.List[str], optional, default: []

argument path: run_mdata/train/
resources/source_list

The env file to be sourced before the command execution.

module_purge:

type: bool, optional, default: False

argument path: run_mdata/train/
resources/module_purge

Remove all modules on HPC system before module load (module_list)

module_unload_list:

type: `typing.List[str]`, optional, default:
`[]`

argument path: `run_mdata/train/
resources/module_unload_list`

The modules to be unloaded on HPC system
before submitting jobs

module_list:

type: `typing.List[str]`, optional, default:
`[]`

argument path: `run_mdata/train/
resources/module_list`

The modules to be loaded on HPC system be-
fore submitting jobs

envs:

type: `dict`, optional, default: `{}`

argument path:
`run_mdata/train/resources/envs`

The environment variables to be exported on
before submitting jobs

prepend_script:

type: `typing.List[str]`, optional, default:
`[]`

argument path: `run_mdata/train/
resources/prepend_script`

Optional script run before jobs submitted.

append_script:

type: `typing.List[str]`, optional, default:
`[]`

argument path: `run_mdata/train/
resources/append_script`

Optional script run after jobs submitted.

wait_time:

type: `float | int`, optional, default: `0`

argument path:
`run_mdata/train/resources/wait_time`

The waiting time in second after a single *task*
submitted

Depending on the value of *batch_type*, different sub args are
accepted.

batch_type:

type: `str` (flag key)

argument path: `run_mdata/train/resources/batch_type`
 possible choices: *Shell*, *PBS*,
DistributedShell, *SlurmJobArray*,
Slurm, *OpenAPI*, *Bohrium*, *SGE*, *Torque*,
Fugaku, *LSF*

The batch job system type loaded from machine/batch_type.

When `batch_type` is set to *Shell* (or its alias *shell*):

kwargs:

type: dict, optional
 argument path: `run_mdata/train/resources[Shell]/kwargs`

This field is empty for this batch.

When `batch_type` is set to *PBS* (or its alias *pbs*):

kwargs:

type: dict, optional
 argument path: `run_mdata/train/resources[PBS]/kwargs`

This field is empty for this batch.

When `batch_type` is set to *DistributedShell* (or its alias *distributedshell*):

kwargs:

type: dict, optional
 argument path: `run_mdata/train/resources[DistributedShell]/kwargs`

This field is empty for this batch.

When `batch_type` is set to *SlurmJobArray* (or its alias *slurmjobarray*):

kwargs:

type: dict, optional
 argument path: `run_mdata/train/resources[SlurmJobArray]/kwargs`

Extra arguments.

custom_gpu_line:

type: str | NoneType, optional,
 default: None
 argument path: `run_mdata/train/resources[SlurmJobArray]/kwargs/custom_gpu_line`

Custom GPU configuration, starting with `#SBATCH`

slurm_job_size:

type: int, optional, default: 1
argument path: run_mdata/train/
resources[SlurmJobArray]/
kwargs/slurm_job_size

Number of tasks in a Slurm job

When `batch_type` is set to Slurm (or its alias `slurm`):

kwargs:

type: dict, optional
argument path: run_mdata/train/
resources[Slurm]/kwargs

Extra arguments.

custom_gpu_line:

type: str | NoneType, optional,
default: None
argument path: run_mdata/train/
resources[Slurm]/kwargs/
custom_gpu_line

Custom GPU configuration, starting
with #SBATCH

When `batch_type` is set to OpenAPI (or its alias `openapi`):

kwargs:

type: dict, optional
argument path: run_mdata/train/
resources[OpenAPI]/kwargs

This field is empty for this batch.

When `batch_type` is set to Bohrium (or its aliases `bohrium`, `Lebesgue`, `lebesgue`, `DpCloudServer`, `dpcloudserver`):

kwargs:

type: dict, optional
argument path: run_mdata/train/
resources[Bohrium]/kwargs

This field is empty for this batch.

When `batch_type` is set to SGE (or its alias `sge`):

kwargs:

type: dict, optional
argument path: run_mdata/train/
resources[SGE]/kwargs

This field is empty for this batch.

When `batch_type` is set to Torque (or its alias `torque`):

kwargs:

type: dict, optional
 argument path: run_mdata/train/
 resources[Torque]/kwargs

This field is empty for this batch.

When `batch_type` is set to Fugaku (or its alias `fugaku`):

kwargs:

type: dict, optional
 argument path: run_mdata/train/
 resources[Fugaku]/kwargs

This field is empty for this batch.

When `batch_type` is set to LSF (or its alias `lsf`):

kwargs:

type: dict
 argument path: run_mdata/train/
 resources[LSF]/kwargs

Extra arguments.

gpu_usage:

type: bool, optional, default: False
 argument path: run_mdata/train/
 resources[LSF]/kwargs/
 gpu_usage

Choosing if GPU is used in the calculation step.

gpu_new_syntax:

type: bool, optional, default: False
 argument path: run_mdata/train/
 resources[LSF]/kwargs/
 gpu_new_syntax

For LFS \geq 10.1.0.3, new option `-gpu` for `#BSUB` could be used. If False, and old syntax would be used.

gpu_exclusive:

type: bool, optional, default: True
 argument path: run_mdata/train/
 resources[LSF]/kwargs/
 gpu_exclusive

Only take effect when new syntax enabled. Control whether submit tasks in exclusive way for GPU.

custom_gpu_line:

type: str | NoneType, optional,
 default: None

argument path: `run_mdata/train/
resources[LSF]/kwargs/
custom_gpu_line`

Custom GPU configuration, starting
with `#BSUB`

user_forward_files:

type: `list`, optional

argument path: `run_mdata/train/user_forward_files`

Files to be forwarded to the remote machine.

user_backward_files:

type: `list`, optional

argument path:

`run_mdata/train/user_backward_files`

Files to be backwarded from the remote machine.

model_devi:

type: `dict`

argument path: `run_mdata/model_devi`

Parameters of command, machine, and resources for `model_devi`

command:

type: `str`

argument path: `run_mdata/model_devi/command`

Command of a program.

machine:

type: `dict`

argument path: `run_mdata/model_devi/machine`

batch_type:

type: `str`

argument path: `run_mdata/model_devi/
machine/batch_type`

The batch job system type. Option: Bohrium,
PBS, OpenAPI, DistributedShell, LSF, Slur-
mJobArray, Slurm, SGE, Fugaku, Torque,
Shell

local_root:

type: `str` | `NoneType`

argument path: `run_mdata/model_devi/
machine/local_root`

The dir where the tasks and relating files locate.
Typically the project dir.

remote_root:

type: str | NoneType, optional
 argument path: run_mdata/model_devi/
 machine/remote_root

The dir where the tasks are executed on the remote machine. Only needed when context is not lazy-local.

clean_asynchronously:

type: bool, optional, default: False
 argument path: run_mdata/model_devi/
 machine/clean_asynchronously

Clean the remote directory asynchronously after the job finishes.

Depending on the value of *context_type*, different sub args are accepted.

context_type:

type: str (flag key)
 argument path: run_mdata/model_devi/
 machine/context_type
 possible choices: *SSHContext*,
LazyLocalContext, *HDFSContext*,
BohriumContext, *OpenAPIContext*,
LocalContext

The connection used to remote machine. Option: LocalContext, SSHContext, OpenAPIContext, BohriumContext, LazyLocalContext, HDFSContext

When *context_type* is set to SSHContext (or its aliases sshcontext, SSH, ssh):

remote_profile:

type: dict
 argument path: run_mdata/model_devi/
 machine[SSHContext]/remote_profile

The information used to maintain the connection with remote machine.

hostname:

type: str
 argument path:
 run_mdata/model_devi/
 machine[SSHContext]/
 remote_profile/hostname
 hostname or ip of ssh connection.

username:

type: str
argument path:
run_mdata/model_devi/
machine[SSHContext]/
remote_profile/username
username of target linux system

password:

type: str, optional
argument path:
run_mdata/model_devi/
machine[SSHContext]/
remote_profile/password
(deprecated) password of linux system. Please use [SSH keys](#) instead to improve security.

port:

type: int, optional, default: 22
argument path:
run_mdata/model_devi/
machine[SSHContext]/
remote_profile/port
ssh connection port.

key_filename:

type: str | NoneType, optional,
default: None
argument path:
run_mdata/model_devi/
machine[SSHContext]/
remote_profile/key_filename
key filename used by ssh connection.
If left None, find key in ~/.ssh or use
password for login

passphrase:

type: str | NoneType, optional,
default: None
argument path:
run_mdata/model_devi/
machine[SSHContext]/
remote_profile/passphrase
passphrase of key used by ssh connection

timeout:

type: int, optional, default: 10
argument path:
run_mdata/model_devi/

```
machine[SSHContext]/
remote_profile/timeout

timeout of ssh connection
```

totp_secret:

```
type: str | NoneType, optional,
default: None
argument path:
run_mdata/model_devi/
machine[SSHContext]/
remote_profile/totp_secret

Time-based one time password se-
cret. It should be a base32-encoded
string extracted from the 2D code.
```

tar_compress:

```
type: bool, optional, default: True
argument path:
run_mdata/model_devi/
machine[SSHContext]/
remote_profile/tar_compress

The archive will be compressed in up-
load and download if it is True. If not,
compression will be skipped.
```

look_for_keys:

```
type: bool, optional, default: True
argument path:
run_mdata/model_devi/
machine[SSHContext]/
remote_profile/look_for_keys

enable searching for discoverable pri-
vate key files in ~/.ssh/
```

When `context_type` is set to `LazyLocalContext` (or its aliases `lazylocalcontext`, `LazyLocal`, `lazylocal`):

remote_profile:

```
type: dict, optional
argument path: run_mdata/model_devi/
machine[LazyLocalContext]/
remote_profile

The information used to maintain the connec-
tion with remote machine. This field is empty
for this context.
```

When `context_type` is set to `HDFSContext` (or its aliases `hdfscontext`, `HDFS`, `hdfs`):

remote_profile:

```
type: dict, optional
```

argument path: run_mdata/model_devi/
machine[HDFSContext]/remote_profile

The information used to maintain the connection with remote machine. This field is empty for this context.

When `context_type` is set to `BohriumContext` (or its aliases `bohriumcontext`, `Bohrium`, `bohrium`, `DpCloudServerContext`, `dpcloudservercontext`, `DpCloudServer`, `dpcloudserver`, `LebesgueContext`, `lebesguecontext`, `Lebesgue`, `lebesgue`):

remote_profile:

type: dict
argument path: run_mdata/model_devi/
machine[BohriumContext]/
remote_profile

The information used to maintain the connection with remote machine.

email:

type: str, optional
argument path:
run_mdata/model_devi/
machine[BohriumContext]/
remote_profile/email
Email

password:

type: str, optional
argument path:
run_mdata/model_devi/
machine[BohriumContext]/
remote_profile/password
Password

program_id:

type: int, alias: *project_id*
argument path:
run_mdata/model_devi/
machine[BohriumContext]/
remote_profile/program_id
Program ID

retry_count:

type: NoneType | int, optional,
default: 2
argument path:
run_mdata/model_devi/
machine[BohriumContext]/
remote_profile/retry_count

The retry count when a job is terminated

ignore_exit_code:

type: bool, optional, default: True
 argument path:
 run_mdata/model_devi/
 machine[BohriumContext]/
 remote_profile/
 ignore_exit_code

The job state will be marked as finished if the exit code is non-zero when set to True. Otherwise, the job state will be designated as terminated.

keep_backup:

type: bool, optional
 argument path:
 run_mdata/model_devi/
 machine[BohriumContext]/
 remote_profile/keep_backup
 keep download and upload zip

input_data:

type: dict
 argument path:
 run_mdata/model_devi/
 machine[BohriumContext]/
 remote_profile/input_data

Configuration of job

When `context_type` is set to `OpenAPIContext` (or its aliases `openapicontext`, `OpenAPI`, `openapi`):

remote_profile:

type: dict, optional
 argument path: run_mdata/model_devi/
 machine[OpenAPIContext]/
 remote_profile

The information used to maintain the connection with remote machine. This field is empty for this context.

When `context_type` is set to `LocalContext` (or its aliases `localcontext`, `Local`, `local`):

remote_profile:

type: dict, optional
 argument path: run_mdata/model_devi/
 machine[LocalContext]/
 remote_profile

The information used to maintain the connection with remote machine. This field is empty for this context.

resources:

type: dict

argument path: run_mdata/model_devi/resources

number_node:

type: int, optional, default: 1

argument path: run_mdata/model_devi/
resources/number_node

The number of node need for each *job*

cpu_per_node:

type: int, optional, default: 1

argument path: run_mdata/model_devi/
resources/cpu_per_node

cpu numbers of each node assigned to each job.

gpu_per_node:

type: int, optional, default: 0

argument path: run_mdata/model_devi/
resources/gpu_per_node

gpu numbers of each node assigned to each job.

queue_name:

type: str, optional, default: (empty string)

argument path: run_mdata/model_devi/
resources/queue_name

The queue name of batch job scheduler system.

group_size:

type: int

argument path: run_mdata/model_devi/
resources/group_size

The number of *tasks* in a *job*. 0 means infinity.

custom_flags:

type: typing.List[str], optional

argument path: run_mdata/model_devi/
resources/custom_flags

The extra lines pass to job submitting script
header

strategy:

type: dict, optional

argument path: `run_mdata/model_devi/
resources/strategy`

strategies we use to generation job submitting scripts.

if_cuda_multi_devices:

type: bool, optional, default: False

argument path:
`run_mdata/model_devi/
resources/strategy/
if_cuda_multi_devices`

If there are multiple nvidia GPUS on the node, and we want to assign the tasks to different GPUS. If true, dpdispatcher will manually export environment variable `CUDA_VISIBLE_DEVICES` to different task. Usually, this option will be used with `Task.task_need_resources` variable simultaneously.

ratio_unfinished:

type: float, optional, default: 0.0

argument path: `run_mdata/
model_devi/resources/
strategy/ratio_unfinished`

The ratio of *tasks* that can be unfinished.

customized_script_header_template_file:

type: str, optional

argument path:
`run_mdata/model_devi/
resources/strategy/
customized_script_header_template_file`

The customized template file to generate job submitting script header, which overrides the default file.

para_deg:

type: int, optional, default: 1

argument path: `run_mdata/model_devi/
resources/para_deg`

Decide how many tasks will be run in parallel.

source_list:

type: `typing.List[str]`, optional, default: []

argument path: `run_mdata/model_devi/
resources/source_list`

The env file to be sourced before the command execution.

module_purge:

type: bool, optional, default: False
argument path: run_mdata/model_devi/
resources/module_purge

Remove all modules on HPC system before module load (module_list)

module_unload_list:

type: typing.List[str], optional, default: []
argument path: run_mdata/model_devi/
resources/module_unload_list

The modules to be unloaded on HPC system before submitting jobs

module_list:

type: typing.List[str], optional, default: []
argument path: run_mdata/model_devi/
resources/module_list

The modules to be loaded on HPC system before submitting jobs

envs:

type: dict, optional, default: {}
argument path:
run_mdata/model_devi/resources/envs

The environment variables to be exported on before submitting jobs

prepend_script:

type: typing.List[str], optional, default: []
argument path: run_mdata/model_devi/
resources/prepend_script

Optional script run before jobs submitted.

append_script:

type: typing.List[str], optional, default: []
argument path: run_mdata/model_devi/
resources/append_script

Optional script run after jobs submitted.

wait_time:

type: float | int, optional, default: 0
 argument path: run_mdata/model_devi/
 resources/wait_time

The waiting time in second after a single *task* submitted

Depending on the value of *batch_type*, different sub args are accepted.

batch_type:

type: str (flag key)
 argument path: run_mdata/model_devi/
 resources/batch_type
 possible choices: *Shell*, *PBS*,
DistributedShell, *SlurmJobArray*,
Slurm, *OpenAPI*, *Bohrium*, *SGE*, *Torque*,
Fugaku, *LSF*

The batch job system type loaded from machine/batch_type.

When *batch_type* is set to *Shell* (or its alias *shell*):

kwargs:

type: dict, optional
 argument path: run_mdata/model_devi/
 resources[Shell]/kwargs

This field is empty for this batch.

When *batch_type* is set to *PBS* (or its alias *pbs*):

kwargs:

type: dict, optional
 argument path: run_mdata/model_devi/
 resources[PBS]/kwargs

This field is empty for this batch.

When *batch_type* is set to *DistributedShell* (or its alias *distributedshell*):

kwargs:

type: dict, optional
 argument path: run_mdata/model_devi/
 resources[DistributedShell]/kwargs

This field is empty for this batch.

When *batch_type* is set to *SlurmJobArray* (or its alias *slurmjobarray*):

kwargs:

type: dict, optional

argument path: `run_mdata/model_devi/
resources[SlurmJobArray]/kwargs`

Extra arguments.

custom_gpu_line:

type: `str | NoneType`, optional,
default: `None`

argument path:
`run_mdata/model_devi/
resources[SlurmJobArray]/
kwargs/custom_gpu_line`

Custom GPU configuration, starting
with `#SBATCH`

slurm_job_size:

type: `int`, optional, default: `1`

argument path:
`run_mdata/model_devi/
resources[SlurmJobArray]/
kwargs/slurm_job_size`

Number of tasks in a Slurm job

When `batch_type` is set to `Slurm` (or its alias `slurm`):

kwargs:

type: `dict`, optional

argument path: `run_mdata/model_devi/
resources[Slurm]/kwargs`

Extra arguments.

custom_gpu_line:

type: `str | NoneType`, optional,
default: `None`

argument path: `run_mdata/
model_devi/resources[Slurm]/
kwargs/custom_gpu_line`

Custom GPU configuration, starting
with `#SBATCH`

When `batch_type` is set to `OpenAPI` (or its alias `openapi`):

kwargs:

type: `dict`, optional

argument path: `run_mdata/model_devi/
resources[OpenAPI]/kwargs`

This field is empty for this batch.

When `batch_type` is set to `Bohrium` (or its aliases `bohrium`,
`Lebesgue`, `lebesgue`, `DpCloudServer`, `dpcloudserver`):

kwargs:

type: `dict`, optional

argument path: `run_mdata/model_devi/resources[Bohrium]/kwargs`

This field is empty for this batch.

When `batch_type` is set to SGE (or its alias `sge`):

kwargs:

type: dict, optional

argument path: `run_mdata/model_devi/resources[SGE]/kwargs`

This field is empty for this batch.

When `batch_type` is set to Torque (or its alias `torque`):

kwargs:

type: dict, optional

argument path: `run_mdata/model_devi/resources[Torque]/kwargs`

This field is empty for this batch.

When `batch_type` is set to Fugaku (or its alias `fugaku`):

kwargs:

type: dict, optional

argument path: `run_mdata/model_devi/resources[Fugaku]/kwargs`

This field is empty for this batch.

When `batch_type` is set to LSF (or its alias `lsf`):

kwargs:

type: dict

argument path: `run_mdata/model_devi/resources[LSF]/kwargs`

Extra arguments.

gpu_usage:

type: bool, optional, default: False

argument path: `run_mdata/model_devi/resources[LSF]/kwargs/gpu_usage`

Choosing if GPU is used in the calculation step.

gpu_new_syntax:

type: bool, optional, default: False

argument path: `run_mdata/model_devi/resources[LSF]/kwargs/gpu_new_syntax`

For LFS \geq 10.1.0.3, new option -gpu for #BSUB could be used. If False, and old syntax would be used.

gpu_exclusive:

type: bool, optional, default: True
argument path: run_mdata/
model_devi/resources[LSF]/
kwargs/gpu_exclusive

Only take effect when new syntax enabled. Control whether submit tasks in exclusive way for GPU.

custom_gpu_line:

type: str | NoneType, optional,
default: None
argument path: run_mdata/
model_devi/resources[LSF]/
kwargs/custom_gpu_line

Custom GPU configuration, starting with #BSUB

user_forward_files:

type: list, optional
argument path:
run_mdata/model_devi/user_forward_files

Files to be forwarded to the remote machine.

user_backward_files:

type: list, optional
argument path:
run_mdata/model_devi/user_backward_files

Files to be backwarded from the remote machine.

fp:

type: dict
argument path: run_mdata/fp

Parameters of command, machine, and resources for fp

command:

type: str
argument path: run_mdata/fp/command

Command of a program.

machine:

type: dict
argument path: run_mdata/fp/machine

batch_type:

type: str

argument path:

run_mdata/fp/machine/batch_type

The batch job system type. Option: Bohrium, PBS, OpenAPI, DistributedShell, LSF, SlurmJobArray, Slurm, SGE, Fugaku, Torque, Shell

local_root:

type: str | NoneType

argument path:

run_mdata/fp/machine/local_root

The dir where the tasks and relating files locate. Typically the project dir.

remote_root:

type: str | NoneType, optional

argument path:

run_mdata/fp/machine/remote_root

The dir where the tasks are executed on the remote machine. Only needed when context is not lazy-local.

clean_asynchronously:

type: bool, optional, default: False

argument path: run_mdata/fp/machine/
clean_asynchronously

Clean the remote directory asynchronously after the job finishes.

Depending on the value of *context_type*, different sub args are accepted.

context_type:

type: str (flag key)

argument path:

run_mdata/fp/machine/context_type

possible choices: *SSHContext*,
LazyLocalContext, *HDFSContext*,
BohriumContext, *OpenAPIContext*,
LocalContext

The connection used to remote machine. Option: LocalContext, SSHContext, OpenAPIContext, BohriumContext, LazyLocalContext, HDFSContext

When *context_type* is set to *SSHContext* (or its aliases *sshcontext*, *SSH*, *ssh*):

remote_profile:

type: dict

argument path: run_mdata/fp/

machine[SSHContext]/remote_profile

The information used to maintain the connection with remote machine.

hostname:

type: str

argument path: run_mdata/fp/

machine[SSHContext]/

remote_profile/hostname

hostname or ip of ssh connection.

username:

type: str

argument path: run_mdata/fp/

machine[SSHContext]/

remote_profile/username

username of target linux system

password:

type: str, optional

argument path: run_mdata/fp/

machine[SSHContext]/

remote_profile/password

(deprecated) password of linux system. Please use [SSH keys](#) instead to improve security.

port:

type: int, optional, default: 22

argument path: run_mdata/fp/

machine[SSHContext]/

remote_profile/port

ssh connection port.

key_filename:

type: str | NoneType, optional,

default: None

argument path: run_mdata/fp/

machine[SSHContext]/

remote_profile/key_filename

key filename used by ssh connection. If left None, find key in ~/.ssh or use password for login

passphrase:

type: str | NoneType, optional,

default: None

argument path: `run_mdata/fp/
machine[SSHContext]/
remote_profile/passphrase`

passphrase of key used by ssh connection

timeout:

type: `int`, optional, default: `10`
argument path: `run_mdata/fp/
machine[SSHContext]/
remote_profile/timeout`

timeout of ssh connection

totp_secret:

type: `str | NoneType`, optional,
default: `None`
argument path: `run_mdata/fp/
machine[SSHContext]/
remote_profile/totp_secret`

Time-based one time password secret. It should be a base32-encoded string extracted from the 2D code.

tar_compress:

type: `bool`, optional, default: `True`
argument path: `run_mdata/fp/
machine[SSHContext]/
remote_profile/tar_compress`

The archive will be compressed in upload and download if it is `True`. If not, compression will be skipped.

look_for_keys:

type: `bool`, optional, default: `True`
argument path: `run_mdata/fp/
machine[SSHContext]/
remote_profile/look_for_keys`

enable searching for discoverable private key files in `~/.ssh/`

When `context_type` is set to `LazyLocalContext` (or its aliases `lazylocalcontext`, `LazyLocal`, `lazylocal`):

remote_profile:

type: `dict`, optional
argument path: `run_mdata/fp/
machine[LazyLocalContext]/
remote_profile`

The information used to maintain the connection with remote machine. This field is empty for this context.

When `context_type` is set to `HDFSContext` (or its aliases `hdfscontext`, `HDFS`, `hdfs`):

remote_profile:

type: dict, optional
argument path: `run_mdata/fp/machine[HDFSContext]/remote_profile`

The information used to maintain the connection with remote machine. This field is empty for this context.

When `context_type` is set to `BohriumContext` (or its aliases `bohriumcontext`, `Bohrium`, `bohrium`, `DpCloudServerContext`, `dpcloudservercontext`, `DpCloudServer`, `dpcloudserver`, `LebesgueContext`, `lebesguecontext`, `Lebesgue`, `lebesgue`):

remote_profile:

type: dict
argument path: `run_mdata/fp/machine[BohriumContext]/remote_profile`

The information used to maintain the connection with remote machine.

email:

type: str, optional
argument path: `run_mdata/fp/machine[BohriumContext]/remote_profile/email`

Email

password:

type: str, optional
argument path: `run_mdata/fp/machine[BohriumContext]/remote_profile/password`

Password

program_id:

type: int, alias: *project_id*
argument path: `run_mdata/fp/machine[BohriumContext]/remote_profile/program_id`

Program ID

retry_count:

type: `NoneType` | int, optional,
default: 2
argument path: `run_mdata/fp/machine[BohriumContext]/remote_profile/retry_count`

The retry count when a job is terminated

ignore_exit_code:

type: bool, optional, default: True
 argument path: run_mdata/fp/
 machine[BohriumContext]/
 remote_profile/
 ignore_exit_code

The job state will be marked as finished if the exit code is non-zero when set to True. Otherwise, the job state will be designated as terminated.

keep_backup:

type: bool, optional
 argument path: run_mdata/fp/
 machine[BohriumContext]/
 remote_profile/keep_backup
 keep download and upload zip

input_data:

type: dict
 argument path: run_mdata/fp/
 machine[BohriumContext]/
 remote_profile/input_data

Configuration of job

When `context_type` is set to `OpenAPIContext` (or its aliases `openapicontext`, `OpenAPI`, `openapi`):

remote_profile:

type: dict, optional
 argument path: run_mdata/fp/
 machine[OpenAPIContext]/
 remote_profile

The information used to maintain the connection with remote machine. This field is empty for this context.

When `context_type` is set to `LocalContext` (or its aliases `localcontext`, `Local`, `local`):

remote_profile:

type: dict, optional
 argument path:
 run_mdata/fp/machine[LocalContext]/
 remote_profile

The information used to maintain the connection with remote machine. This field is empty for this context.

resources:

type: dict

argument path: run_mdata/fp/resources

number_node:

type: int, optional, default: 1

argument path:

run_mdata/fp/resources/number_node

The number of node need for each *job*

cpu_per_node:

type: int, optional, default: 1

argument path:

run_mdata/fp/resources/cpu_per_node

cpu numbers of each node assigned to each job.

gpu_per_node:

type: int, optional, default: 0

argument path:

run_mdata/fp/resources/gpu_per_node

gpu numbers of each node assigned to each job.

queue_name:

type: str, optional, default: (empty string)

argument path:

run_mdata/fp/resources/queue_name

The queue name of batch job scheduler system.

group_size:

type: int

argument path:

run_mdata/fp/resources/group_size

The number of *tasks* in a *job*. 0 means infinity.

custom_flags:

type: typing.List[str], optional

argument path:

run_mdata/fp/resources/custom_flags

The extra lines pass to job submitting script header

strategy:

type: dict, optional

argument path:

run_mdata/fp/resources/strategy

strategies we use to generation job submitting scripts.

if_cuda_multi_devices:

type: bool, optional, default: False
 argument path: run_mdata/fp/
 resources/strategy/
 if_cuda_multi_devices

If there are multiple nvidia GPUS on the node, and we want to assign the tasks to different GPUS. If true, dpdispatcher will manually export environment variable CUDA_VISIBLE_DEVICES to different task. Usually, this option will be used with Task.task_need_resources variable simultaneously.

ratio_unfinished:

type: float, optional, default: 0.0
 argument path:
 run_mdata/fp/resources/
 strategy/ratio_unfinished

The ratio of *tasks* that can be unfinished.

customized_script_header_template_file:

type: str, optional
 argument path: run_mdata/fp/
 resources/strategy/
 customized_script_header_template_file

The customized template file to generate job submitting script header, which overrides the default file.

para_deg:

type: int, optional, default: 1
 argument path:
 run_mdata/fp/resources/para_deg

Decide how many tasks will be run in parallel.

source_list:

type: typing.List[str], optional, default: []
 argument path:
 run_mdata/fp/resources/source_list

The env file to be sourced before the command execution.

module_purge:

type: bool, optional, default: False
 argument path:
 run_mdata/fp/resources/module_purge

Remove all modules on HPC system before module load (module_list)

module_unload_list:

type: `typing.List[str]`, optional, default: `[]`

argument path: `run_mdata/fp/resources/module_unload_list`

The modules to be unloaded on HPC system before submitting jobs

module_list:

type: `typing.List[str]`, optional, default: `[]`

argument path: `run_mdata/fp/resources/module_list`

The modules to be loaded on HPC system before submitting jobs

envs:

type: `dict`, optional, default: `{}`

argument path: `run_mdata/fp/resources/envs`

The environment variables to be exported on before submitting jobs

prepend_script:

type: `typing.List[str]`, optional, default: `[]`

argument path: `run_mdata/fp/resources/prepend_script`

Optional script run before jobs submitted.

append_script:

type: `typing.List[str]`, optional, default: `[]`

argument path: `run_mdata/fp/resources/append_script`

Optional script run after jobs submitted.

wait_time:

type: `float | int`, optional, default: `0`

argument path: `run_mdata/fp/resources/wait_time`

The waiting time in second after a single *task* submitted

Depending on the value of *batch_type*, different sub args are accepted.

batch_type:

type: str (flag key)

argument path:

run_mdata/fp/resources/batch_type

possible choices: *Shell*, *PBS*,
DistributedShell, *SlurmJobArray*,
Slurm, *OpenAPI*, *Bohrium*, *SGE*, *Torque*,
Fugaku, *LSF*

The batch job system type loaded from machine/batch_type.

When *batch_type* is set to *Shell* (or its alias *shell*):

kwargs:

type: dict, optional

argument path: run_mdata/fp/

resources[Shell]/kwargs

This field is empty for this batch.

When *batch_type* is set to *PBS* (or its alias *pbs*):

kwargs:

type: dict, optional

argument path:

run_mdata/fp/resources[PBS]/kwargs

This field is empty for this batch.

When *batch_type* is set to *DistributedShell* (or its alias *distributedshell*):

kwargs:

type: dict, optional

argument path: run_mdata/fp/

resources[DistributedShell]/kwargs

This field is empty for this batch.

When *batch_type* is set to *SlurmJobArray* (or its alias *slurmjobarray*):

kwargs:

type: dict, optional

argument path: run_mdata/fp/

resources[SlurmJobArray]/kwargs

Extra arguments.

custom_gpu_line:

type: str | NoneType, optional,

default: None

argument path: run_mdata/fp/

resources[SlurmJobArray]/

kwargs/custom_gpu_line

Custom GPU configuration, starting with #SBATCH

slurm_job_size:

type: int, optional, default: 1
argument path: run_mdata/fp/
resources[SlurmJobArray]/
kwargs/slurm_job_size

Number of tasks in a Slurm job

When `batch_type` is set to Slurm (or its alias `slurm`):

kwargs:

type: dict, optional
argument path: run_mdata/fp/
resources[Slurm]/kwargs

Extra arguments.

custom_gpu_line:

type: str | NoneType, optional,
default: None
argument path: run_mdata/fp/
resources[Slurm]/kwargs/
custom_gpu_line

Custom GPU configuration, starting with #SBATCH

When `batch_type` is set to OpenAPI (or its alias `openapi`):

kwargs:

type: dict, optional
argument path: run_mdata/fp/
resources[OpenAPI]/kwargs

This field is empty for this batch.

When `batch_type` is set to Bohrium (or its aliases `bohrium`, `Lebesgue`, `lebesgue`, `DpCloudServer`, `dpcloudserver`):

kwargs:

type: dict, optional
argument path: run_mdata/fp/
resources[Bohrium]/kwargs

This field is empty for this batch.

When `batch_type` is set to SGE (or its alias `sge`):

kwargs:

type: dict, optional
argument path:
run_mdata/fp/resources[SGE]/kwargs

This field is empty for this batch.

When `batch_type` is set to Torque (or its alias `torque`):

kwargs:

type: dict, optional
 argument path: `run_mdata/fp/resources[Torque]/kwargs`

This field is empty for this batch.

When `batch_type` is set to Fugaku (or its alias `fugaku`):

kwargs:

type: dict, optional
 argument path: `run_mdata/fp/resources[Fugaku]/kwargs`

This field is empty for this batch.

When `batch_type` is set to LSF (or its alias `lsf`):

kwargs:

type: dict
 argument path:
`run_mdata/fp/resources[LSF]/kwargs`

Extra arguments.

gpu_usage:

type: bool, optional, default: False
 argument path:
`run_mdata/fp/resources[LSF]/kwargs/gpu_usage`

Choosing if GPU is used in the calculation step.

gpu_new_syntax:

type: bool, optional, default: False
 argument path:
`run_mdata/fp/resources[LSF]/kwargs/gpu_new_syntax`

For LFS >= 10.1.0.3, new option `-gpu` for `#BSUB` could be used. If False, and old syntax would be used.

gpu_exclusive:

type: bool, optional, default: True
 argument path:
`run_mdata/fp/resources[LSF]/kwargs/gpu_exclusive`

Only take effect when new syntax enabled. Control whether submit tasks in exclusive way for GPU.

custom_gpu_line:

type: str | NoneType, optional,
default: None

argument path:

run_mdata/fp/resources[LSF]/
kwargs/custom_gpu_line

Custom GPU configuration, starting
with #BSUB

user_forward_files:

type: list, optional

argument path: run_mdata/fp/user_forward_files

Files to be forwarded to the remote machine.

user_backward_files:

type: list, optional

argument path: run_mdata/fp/user_backward_files

Files to be backwarded from the remote machine.

5.1 Init_bulk

You may prepare initial data for bulk systems with VASP by:

```
dpngen init_bulk PARAM [MACHINE]
```

The MACHINE configure file is optional. If this parameter exists, then the optimization tasks or MD tasks will be submitted automatically according to MACHINE.json.

Basically init_bulk can be divided into four parts, denoted as stages in PARAM:

1. Relax in folder 00.place_ele
2. Perturb and scale in folder 01.scale_pert
3. Run a short AIMD in folder 02.md
4. Collect data in folder 02.md.

All stages must be **in order**. One doesn't need to run all stages. For example, you may run stage 1 and 2, generating supercells as starting point of exploration in dpngen run.

If MACHINE is None, there should be only one stage in stages. Corresponding tasks will be generated, but user's intervention should be involved in, to manually run the scripts.

Following is an example for PARAM, which generates data from a typical structure hcp.

```
{
  "stages" : [1,2,3,4],
  "cell_type": "hcp",
  "latt": 4.479,
  "super_cell": [2, 2, 2],
  "elements": ["Mg"],
  "potcars": ["...../POTCAR"],
  "relax_incar": "...../INCAR_metal_rlx",
  "md_incar" : "...../INCAR_metal_md",
  "scale": [1.00],
  "skip_relax": false,
  "pert_numb": 2,
  "md_nstep" : 5,
  "pert_box": 0.03,
  "pert_atom": 0.01,
  "coll_ndata": 5000,
  "type_map" : [ "Mg", "Al"],
```

(continues on next page)

(continued from previous page)

```

    "_comment":      "that's all"
}

```

If you want to specify a structure as starting point for `init_bulk`, you may set in `PARAM` as follows.

```

"from_poscar":      true,
"from_poscar_path": "...../C_mp-47_conventional.POSCAR",

```

`init_bulk` supports both VASP and ABACUS for first-principle calculation. You can choose the software by specifying the key `init_fp_style`. If `init_fp_style` is not specified, the default software will be VASP.

When using ABACUS for `init_fp_style`, the keys of the paths of INPUT files for relaxation and MD simulations are the same as INCAR for VASP, which are `relax_incar` and `md_incar` respectively. Use `relax_kpt` and `md_kpt` for the relative path for KPT files of relaxation and MD simulations. They two can be omitted if `kspacing` (in unit of 1/Bohr) or `gamma_only` has been set in corresponding INPUT files. If `from_poscar` is set to `false`, you have to specify `atom_masses` in the same order as elements.

5.2 dpngen init_bulk parameters

Note: One can load, modify, and export the input file by using our effective web-based tool [DP-GUI](#) online or hosted using the *command line interface* `dpngen gui`. All parameters below can be set in DP-GUI. By clicking “SAVE JSON”, one can download the input file.

`init_bulk_jdata:`

type: dict

argument path: `init_bulk_jdata`

Generate initial data for bulk systems.

`stages:`

type: list[int]

argument path: `init_bulk_jdata/stages`

Stages for *init_bulk*.

`elements:`

type: list[str]

argument path: `init_bulk_jdata/elements`

Atom types.

`potcars:`

type: list[str], optional

argument path: `init_bulk_jdata/potcars`

Path of POTCAR.

`cell_type:`

type: str, optional

argument path: `init_bulk_jdata/cell_type`

Specifying which typical structure to be generated. **Options** include `fcc`, `hcp`, `bcc`, `sc`, `diamond`.

super_cell:

type: list[int]
 argument path: init_bulk_jdata/super_cell
 Size of supercell.

from_poscar:

type: bool, optional, default: False
 argument path: init_bulk_jdata/from_poscar
 Deciding whether to use a given poscar as the beginning of relaxation. If it's true, keys (*cell_type*, *latt*) will be aborted. Otherwise, these two keys are **necessary**.

from_poscar_path:

type: str, optional
 argument path: init_bulk_jdata/from_poscar_path
 Path of POSCAR for VASP or STRU for ABACUS. **Necessary** if *from_poscar* is true.

relax_incar:

type: str, optional
 argument path: init_bulk_jdata/relax_incar
 Path of INCAR for VASP or INPUT for ABACUS for relaxation in VASP. **Necessary** if *stages* include 1.

md_incar:

type: str, optional
 argument path: init_bulk_jdata/md_incar
 Path of INCAR for VASP or INPUT for ABACUS for MD in VASP. **Necessary** if *stages* include 3.

scale:

type: list[float]
 argument path: init_bulk_jdata/scale
 Scales for isotropic transforming cells.

skip_relax:

type: bool
 argument path: init_bulk_jdata/skip_relax
 If it's true, you may directly run stage 2 (perturb and scale) using an unrelaxed POSCAR.

pert_numb:

type: int
 argument path: init_bulk_jdata/pert_numb
 Number of perturbations for each scaled (key *scale*) POSCAR.

pert_box:

type: float
 argument path: init_bulk_jdata/pert_box

Anisotropic Perturbation for cells (independent changes of lengths of three box vectors as well as angle among) in decimal formats. 9 elements of the 3x3 perturbation matrix will be randomly sampled from a uniform distribution (default) in the range $[-\text{pert_box}, \text{pert_box}]$. Such a perturbation matrix adds the identity matrix gives the actual transformation matrix for this perturbation operation.

pert_atom:

type: float

argument path: `init_bulk_jdata/pert_atom`

Perturbation of atom coordinates (Angstrom). Random perturbations are performed on three coordinates of each atom by adding values randomly sampled from a uniform distribution in the range $[-\text{pert_atom}, \text{pert_atom}]$.

md_nstep:

type: int

argument path: `init_bulk_jdata/md_nstep`

Steps of AIMD in stage 3. If it's not equal to settings via *NSW* in *md_incar*, DP-GEN will follow *NSW*.

coll_ndata:

type: int

argument path: `init_bulk_jdata/coll_ndata`

Maximal number of collected data.

type_map:

type: `list[str]`, optional

argument path: `init_bulk_jdata/type_map`

The indices of elements in deepmd formats will be set in this order.

Depending on the value of *init_fp_style*, different sub args are accepted.

init_fp_style:

type: `str` (flag key), default: VASP

argument path: `init_bulk_jdata/init_fp_style`

possible choices: [VASP](#), [ABACUS](#)

First-principle software. If this key is absent.

When *init_fp_style* is set to VASP:

No more parameters is needed to be added.

When *init_fp_style* is set to ABACUS:

ABACUS

relax_kpt:

type: `str`, optional

argument path: `init_bulk_jdata[ABACUS]/relax_kpt`

Path of *KPT* file for relaxation in stage 1. Only useful if *init_fp_style* is "ABACUS".

md_kpt:

type: `str`, optional

argument path: `init_bulk_jdata[ABACUS]/md_kpt`

Path of *KPT* file for MD simulations in stage 3. Only useful if *init_fp_style* is “ABACUS”.

atom_masses:

type: `list[float]`, optional

argument path: `init_bulk_jdata[ABACUS]/atom_masses`

List of atomic masses of elements. The order should be the same as *Elements*.

Only useful if *init_fp_style* is “ABACUS”.

5.3 dpngen init_bulk machine parameters

Note: One can load, modify, and export the input file by using our effective web-based tool [DP-GUI](#) online or hosted using the *command line interface* `dpngen gui`. All parameters below can be set in DP-GUI. By clicking “SAVE JSON”, one can download the input file.

init_bulk_mdata:

type: `dict`

argument path: `init_bulk_mdata`

machine.json file

api_version:

type: `str`, optional, default: 1.0

argument path: `init_bulk_mdata/api_version`

Please set to 1.0

deepmd_version:

type: `str`, optional, default: 2

argument path: `init_bulk_mdata/deepmd_version`

DeePMD-kit version, e.g. 2.1.3

fp:

type: `dict`

argument path: `init_bulk_mdata/fp`

Parameters of command, machine, and resources for fp

command:

type: `str`

argument path: `init_bulk_mdata/fp/command`

Command of a program.

machine:

type: `dict`

argument path: `init_bulk_mdata/fp/machine`

batch_type:

type: str
argument path: `init_bulk_mdata/fp/
machine/batch_type`

The batch job system type. Option: Bohrium, PBS, OpenAPI, DistributedShell, LSF, SlurmJobArray, Slurm, SGE, Fugaku, Torque, Shell

local_root:

type: str | NoneType
argument path: `init_bulk_mdata/fp/
machine/local_root`

The dir where the tasks and relating files locate. Typically the project dir.

remote_root:

type: str | NoneType, optional
argument path: `init_bulk_mdata/fp/
machine/remote_root`

The dir where the tasks are executed on the remote machine. Only needed when context is not lazy-local.

clean_asynchronously:

type: bool, optional, default: False
argument path: `init_bulk_mdata/fp/
machine/clean_asynchronously`

Clean the remote directory asynchronously after the job finishes.

Depending on the value of *context_type*, different sub args are accepted.

context_type:

type: str (flag key)
argument path: `init_bulk_mdata/fp/
machine/context_type`
possible choices: *SSHContext*,
LazyLocalContext, *HDFSContext*,
BohriumContext, *OpenAPIContext*,
LocalContext

The connection used to remote machine. Option: LocalContext, SSHContext, OpenAPIContext, BohriumContext, LazyLocalContext, HDFSContext

When *context_type* is set to *SSHContext* (or its aliases *sshcontext*, *SSH*, *ssh*):

remote_profile:

type: dict

argument path: `init_bulk_mdata/fp/machine[SSHContext]/remote_profile`

The information used to maintain the connection with remote machine.

hostname:

type: str

argument path: `init_bulk_mdata/fp/machine[SSHContext]/remote_profile/hostname`

hostname or ip of ssh connection.

username:

type: str

argument path: `init_bulk_mdata/fp/machine[SSHContext]/remote_profile/username`

username of target linux system

password:

type: str, optional

argument path: `init_bulk_mdata/fp/machine[SSHContext]/remote_profile/password`

(deprecated) password of linux system. Please use [SSH keys](#) instead to improve security.

port:

type: int, optional, default: 22

argument path: `init_bulk_mdata/fp/machine[SSHContext]/remote_profile/port`

ssh connection port.

key_filename:

type: str | NoneType, optional,
default: None

argument path: `init_bulk_mdata/fp/machine[SSHContext]/remote_profile/key_filename`

key filename used by ssh connection.
If left None, find key in `~/.ssh` or use password for login

passphrase:

type: str | NoneType, optional,
default: None

argument path: `init_bulk_mdata/
fp/machine[SSHContext]/
remote_profile/passphrase`

passphrase of key used by ssh connection

timeout:

type: `int`, optional, default: `10`

argument path: `init_bulk_mdata/
fp/machine[SSHContext]/
remote_profile/timeout`

timeout of ssh connection

totp_secret:

type: `str | NoneType`, optional,
default: `None`

argument path: `init_bulk_mdata/
fp/machine[SSHContext]/
remote_profile/totp_secret`

Time-based one time password secret. It should be a base32-encoded string extracted from the 2D code.

tar_compress:

type: `bool`, optional, default: `True`

argument path: `init_bulk_mdata/
fp/machine[SSHContext]/
remote_profile/tar_compress`

The archive will be compressed in upload and download if it is `True`. If not, compression will be skipped.

look_for_keys:

type: `bool`, optional, default: `True`

argument path: `init_bulk_mdata/
fp/machine[SSHContext]/
remote_profile/look_for_keys`

enable searching for discoverable private key files in `~/.ssh/`

When `context_type` is set to `LazyLocalContext` (or its aliases `lazylocalcontext`, `LazyLocal`, `lazylocal`):

remote_profile:

type: `dict`, optional

argument path: `init_bulk_mdata/fp/
machine[LazyLocalContext]/
remote_profile`

The information used to maintain the connection with remote machine. This field is empty for this context.

When `context_type` is set to `HDFSContext` (or its aliases `hdfscontext`, `HDFS`, `hdfs`):

remote_profile:

type: dict, optional
 argument path: `init_bulk_mdata/fp/machine[HDFSContext]/remote_profile`

The information used to maintain the connection with remote machine. This field is empty for this context.

When `context_type` is set to `BohriumContext` (or its aliases `bohriumcontext`, `Bohrium`, `bohrium`, `DpCloudServerContext`, `dpcloudservercontext`, `DpCloudServer`, `dpcloudserver`, `LebesgueContext`, `lebesguecontext`, `Lebesgue`, `lebesgue`):

remote_profile:

type: dict
 argument path: `init_bulk_mdata/fp/machine[BohriumContext]/remote_profile`

The information used to maintain the connection with remote machine.

email:

type: str, optional
 argument path: `init_bulk_mdata/fp/machine[BohriumContext]/remote_profile/email`

Email

password:

type: str, optional
 argument path: `init_bulk_mdata/fp/machine[BohriumContext]/remote_profile/password`

Password

program_id:

type: int, alias: *project_id*
 argument path: `init_bulk_mdata/fp/machine[BohriumContext]/remote_profile/program_id`

Program ID

retry_count:

type: `NoneType` | int, optional,
 default: 2
 argument path: `init_bulk_mdata/fp/machine[BohriumContext]/remote_profile/retry_count`

The retry count when a job is terminated

ignore_exit_code:

type: bool, optional, default: True
argument path: init_bulk_mdata/
fp/machine[BohriumContext]/
remote_profile/
ignore_exit_code

The job state will be marked as finished if the exit code is non-zero when set to True. Otherwise, the job state will be designated as terminated.

keep_backup:

type: bool, optional
argument path: init_bulk_mdata/
fp/machine[BohriumContext]/
remote_profile/keep_backup

keep download and upload zip

input_data:

type: dict
argument path: init_bulk_mdata/
fp/machine[BohriumContext]/
remote_profile/input_data

Configuration of job

When `context_type` is set to `OpenAPIContext` (or its aliases `openapicontext`, `OpenAPI`, `openapi`):

remote_profile:

type: dict, optional
argument path: init_bulk_mdata/fp/
machine[OpenAPIContext]/
remote_profile

The information used to maintain the connection with remote machine. This field is empty for this context.

When `context_type` is set to `LocalContext` (or its aliases `localcontext`, `Local`, `local`):

remote_profile:

type: dict, optional
argument path: init_bulk_mdata/fp/
machine[LocalContext]/
remote_profile

The information used to maintain the connection with remote machine. This field is empty for this context.

resources:

type: dict

argument path: `init_bulk_mdata/fp/resources`

number_node:

type: int, optional, default: 1

argument path: `init_bulk_mdata/fp/resources/number_node`

The number of node need for each *job*

cpu_per_node:

type: int, optional, default: 1

argument path: `init_bulk_mdata/fp/resources/cpu_per_node`

cpu numbers of each node assigned to each job.

gpu_per_node:

type: int, optional, default: 0

argument path: `init_bulk_mdata/fp/resources/gpu_per_node`

gpu numbers of each node assigned to each job.

queue_name:

type: str, optional, default: (empty string)

argument path: `init_bulk_mdata/fp/resources/queue_name`

The queue name of batch job scheduler system.

group_size:

type: int

argument path: `init_bulk_mdata/fp/resources/group_size`

The number of *tasks* in a *job*. 0 means infinity.

custom_flags:

type: typing.List[str], optional

argument path: `init_bulk_mdata/fp/resources/custom_flags`

The extra lines pass to job submitting script header

strategy:

type: dict, optional

argument path: `init_bulk_mdata/fp/resources/strategy`

strategies we use to generation job submitting scripts.

if_cuda_multi_devices:

type: bool, optional, default: False
argument path: `init_bulk_mdata/
fp/resources/strategy/
if_cuda_multi_devices`

If there are multiple nvidia GPUS on the node, and we want to assign the tasks to different GPUS. If true, `dpdispatcher` will manually export environment variable `CUDA_VISIBLE_DEVICES` to different task. Usually, this option will be used with `Task.task_need_resources` variable simultaneously.

ratio_unfinished:

type: float, optional, default: 0.0
argument path: `init_bulk_mdata/
fp/resources/strategy/
ratio_unfinished`

The ratio of *tasks* that can be unfinished.

customized_script_header_template_file:

type: str, optional
argument path: `init_bulk_mdata/
fp/resources/strategy/
customized_script_header_template_file`

The customized template file to generate job submitting script header, which overrides the default file.

para_deg:

type: int, optional, default: 1
argument path: `init_bulk_mdata/fp/
resources/para_deg`

Decide how many tasks will be run in parallel.

source_list:

type: `typing.List[str]`, optional, default: []
argument path: `init_bulk_mdata/fp/
resources/source_list`

The env file to be sourced before the command execution.

module_purge:

type: bool, optional, default: False
argument path: `init_bulk_mdata/fp/
resources/module_purge`

Remove all modules on HPC system before module load (module_list)

module_unload_list:

type: `typing.List[str]`, optional, default: `[]`

argument path: `init_bulk_mdata/fp/resources/module_unload_list`

The modules to be unloaded on HPC system before submitting jobs

module_list:

type: `typing.List[str]`, optional, default: `[]`

argument path: `init_bulk_mdata/fp/resources/module_list`

The modules to be loaded on HPC system before submitting jobs

envs:

type: `dict`, optional, default: `{}`

argument path: `init_bulk_mdata/fp/resources/envs`

The environment variables to be exported on before submitting jobs

prepend_script:

type: `typing.List[str]`, optional, default: `[]`

argument path: `init_bulk_mdata/fp/resources/prepend_script`

Optional script run before jobs submitted.

append_script:

type: `typing.List[str]`, optional, default: `[]`

argument path: `init_bulk_mdata/fp/resources/append_script`

Optional script run after jobs submitted.

wait_time:

type: `float | int`, optional, default: `0`

argument path: `init_bulk_mdata/fp/resources/wait_time`

The waiting time in second after a single *task* submitted

Depending on the value of *batch_type*, different sub args are accepted.

batch_type:

type: str (flag key)
argument path: `init_bulk_mdata/fp/resources/batch_type`
possible choices: *Shell*, *PBS*,
DistributedShell, *SlurmJobArray*,
Slurm, *OpenAPI*, *Bohrium*, *SGE*, *Torque*,
Fugaku, *LSF*
The batch job system type loaded from machine/batch_type.

When `batch_type` is set to *Shell* (or its alias *shell*):

kwargs:

type: dict, optional
argument path: `init_bulk_mdata/fp/resources[Shell]/kwargs`
This field is empty for this batch.

When `batch_type` is set to *PBS* (or its alias *pbs*):

kwargs:

type: dict, optional
argument path: `init_bulk_mdata/fp/resources[PBS]/kwargs`
This field is empty for this batch.

When `batch_type` is set to *DistributedShell* (or its alias *distributedshell*):

kwargs:

type: dict, optional
argument path: `init_bulk_mdata/fp/resources[DistributedShell]/kwargs`
This field is empty for this batch.

When `batch_type` is set to *SlurmJobArray* (or its alias *slurmjobarray*):

kwargs:

type: dict, optional
argument path: `init_bulk_mdata/fp/resources[SlurmJobArray]/kwargs`
Extra arguments.

custom_gpu_line:

type: str | NoneType, optional,
default: None
argument path: `init_bulk_mdata/fp/resources[SlurmJobArray]/kwargs/custom_gpu_line`

Custom GPU configuration, starting with #SBATCH

slurm_job_size:

type: int, optional, default: 1
 argument path: `init_bulk_mdata/fp/resources[SlurmJobArray]/kwargs/slurm_job_size`

Number of tasks in a Slurm job

When `batch_type` is set to Slurm (or its alias `slurm`):

kwargs:

type: dict, optional
 argument path: `init_bulk_mdata/fp/resources[Slurm]/kwargs`

Extra arguments.

custom_gpu_line:

type: str | NoneType, optional,
 default: None
 argument path: `init_bulk_mdata/fp/resources[Slurm]/kwargs/custom_gpu_line`

Custom GPU configuration, starting with #SBATCH

When `batch_type` is set to OpenAPI (or its alias `openapi`):

kwargs:

type: dict, optional
 argument path: `init_bulk_mdata/fp/resources[OpenAPI]/kwargs`

This field is empty for this batch.

When `batch_type` is set to Bohrium (or its aliases `bohrium`, `Lebesgue`, `lebesgue`, `DpCloudServer`, `dpcloudserver`):

kwargs:

type: dict, optional
 argument path: `init_bulk_mdata/fp/resources[Bohrium]/kwargs`

This field is empty for this batch.

When `batch_type` is set to SGE (or its alias `sge`):

kwargs:

type: dict, optional
 argument path: `init_bulk_mdata/fp/resources[SGE]/kwargs`

This field is empty for this batch.

When `batch_type` is set to Torque (or its alias `torque`):

kwargs:

type: dict, optional
argument path: `init_bulk_mdata/fp/resources[Torque]/kwargs`

This field is empty for this batch.

When `batch_type` is set to Fugaku (or its alias `fugaku`):

kwargs:

type: dict, optional
argument path: `init_bulk_mdata/fp/resources[Fugaku]/kwargs`

This field is empty for this batch.

When `batch_type` is set to LSF (or its alias `lsf`):

kwargs:

type: dict
argument path: `init_bulk_mdata/fp/resources[LSF]/kwargs`

Extra arguments.

gpu_usage:

type: bool, optional, default: False
argument path: `init_bulk_mdata/fp/resources[LSF]/kwargs/gpu_usage`

Choosing if GPU is used in the calculation step.

gpu_new_syntax:

type: bool, optional, default: False
argument path: `init_bulk_mdata/fp/resources[LSF]/kwargs/gpu_new_syntax`

For LFS \geq 10.1.0.3, new option `-gpu` for `#BSUB` could be used. If False, and old syntax would be used.

gpu_exclusive:

type: bool, optional, default: True
argument path: `init_bulk_mdata/fp/resources[LSF]/kwargs/gpu_exclusive`

Only take effect when new syntax enabled. Control whether submit tasks in exclusive way for GPU.

custom_gpu_line:

type: str | NoneType, optional,
 default: None
 argument path: `init_bulk_mdata/
 fp/resources[LSF]/kwargs/
 custom_gpu_line`
 Custom GPU configuration, starting
 with #BSUB

user_forward_files:

type: list, optional
 argument path:
`init_bulk_mdata/fp/user_forward_files`
 Files to be forwarded to the remote machine.

user_backward_files:

type: list, optional
 argument path:
`init_bulk_mdata/fp/user_backward_files`
 Files to be backwarded from the remote machine.

5.4 Init_surf

You may prepare initial data for surface systems with VASP by:

```
dpngen init_surf PARAM [MACHINE]
```

The MACHINE configure file is optional. If this parameter exists, then the optimization tasks or MD tasks will be submitted automatically according to MACHINE.json. That is to say, if one only wants to prepare `surf-xxx/sys-xxx` folders for the second stage but wants to skip relaxation, `dpngen init_surf PARAM` should be used (without MACHINE). “stages” and “skip_relax” in PARAM should be set as:

```
"stages": [1,2],  

"skip_relax": true,
```

Basically `init_surf` can be divided into two parts, denoted as `stages` in PARAM:

1. Build specific surface in folder `00.place_ele`
2. Pertub and scale in folder `01.scale_pert`

All stages must be **in order**.

Generally, `init_surf` does not run AIMD but only generates a lot of configurations. Compared with `init_bulk`, which runs DFT calculations twice, `init_surf` does once. Usually, we do `init_bulk`, run many rounds of DP-GEN iterations, collect enough data for the bulk system, and do `init_surf` after that. At this point, the lattice constant has been determined, and the lattice constant required for the initial configuration of `init_surf` can be used directly. These configurations made by `init_surf` are prepared for `01.model_devi`. Candidates will do DFT calculation in `02.fp`.

- Generate vacuum layers

According to [the source code of `pert_scaled`](#), `init_surf` will generate a series of surface structures with specified separations between the sample layer and its periodic image. There are two ways to specify the interval in generating the vacuum layers: 1) to set the interval value and 2) to set the number of intervals.

You can use `layer_numb` (the number of layers of the slab) or `z_min` (the total thickness) to specify the thickness of the atoms below. Then `vacuum_*` parameters specify the vacuum layers above. `dpgen init_surf` will make a series of structures with the thickness of vacuum layers from `vacuum_min` to `vacuum_max`. The number of vacuum layers is controlled by the parameter `vacuum_resol`.

The layers will be generated even when the size of `vacuum_resol` is 1. When the size of `vacuum_resol` is 2 or it is empty, the whole interval range is divided into the nearby region with denser intervals (head region) and the far-away region with sparser intervals (tail region), which are divided by `mid_point`.

When the size of `vacuum_resol` is 2, two elements respectively decide the number of intervals in head region and tail region.

When `vacuum_resol` is empty, the number of intervals in the head region = `vacuum_num` * `head_ratio`. `vacuum_num` and `head_ratio` are both keys in `param.json`.

- Attach files in the task path

One can use the machine parameter `forward_files` to upload other files besides POSCAR, INCAR, and POTCAR. For example, “`vdw_kernal.bindat`” for each task.

See [the document of task parameters](#).

Following is an example for `PARAM`, which generates data from a typical structure `fcc`.

```
{
  "stages": [
    1,
    2
  ],
  "cell_type": "fcc",
  "latt": 4.034,
  "super_cell": [
    2,
    2,
    2
  ],
  "layer_numb": 3,
  "vacuum_max": 9.0,
  "vacuum_resol": [
    0.5,
    1
  ],
  "mid_point": 4.0,
  "millers": [
    [
      1,
      0,
      0
    ],
    [
      1,
      1,
      0
    ]
  ]
}
```

(continues on next page)

(continued from previous page)

```

    ],
    [
        1,
        1,
        1
    ]
],
"elements": [
    "Al"
],
"potcars": [
    "...../POTCAR"
],
"relax_incar": "...../INCAR_metal_rlx_low",
"scale": [
    1.0
],
"skip_relax": true,
"pert_numb": 2,
"pert_box": 0.03,
"pert_atom": 0.01,
"_comment": "that's all"
}

```

Another example is `from_poscar` method. Here you need to specify the POSCAR file.

```

{
    "stages": [
        1,
        2
    ],
    "cell_type": "fcc",
    "from_poscar": true,
    "from_poscar_path": "POSCAR",
    "super_cell": [
        1,
        1,
        1
    ],
    "layer_numb": 3,
    "vacuum_max": 5.0,
    "vacuum_resol": [0.5, 2],
    "mid_point": 2.0,
    "millers": [
        [
            1,
            0,
            0
        ]
    ],
    "elements": [
        "Al"
    ]
}

```

(continues on next page)

(continued from previous page)

```

],
"potcars": [
  "./POTCAR"
],
"relax_incar" : "INCAR_metal_rlx_low",
"scale": [
  1.0
],
"skip_relax": true,
"pert_numb": 5,
"pert_box": 0.03,
"pert_atom": 0.01,
"coll_ndata": 5000,
"_comment": "that's all"
}

```

5.5 dpngen init_surf parameters

Note: One can load, modify, and export the input file by using our effective web-based tool [DP-GUI](#) online or hosted using the *command line interface* `dpngen gui`. All parameters below can be set in DP-GUI. By clicking “SAVE JSON”, one can download the input file.

init_surf_jdata:

type: dict

argument path: `init_surf_jdata`

Generate initial data for surface systems.

stages:

type: `list[int]`

argument path: `init_surf_jdata/stages`

Stages for *init_surf*.

elements:

type: `list[str]`

argument path: `init_surf_jdata/elements`

Atom types.

potcars:

type: `list[str]`, optional

argument path: `init_surf_jdata/potcars`

Path of POTCAR.

cell_type:

type: `str`, optional

argument path: `init_surf_jdata/cell_type`

Specifying which typical structure to be generated. **Options** include `fcc`, `hcp`, `bcc`, `sc`, `diamond`.

super_cell:

type: list[int]

argument path: `init_surf_jdata/super_cell`

Size of supercell.

from_poscar:

type: bool, optional, default: False

argument path: `init_surf_jdata/from_poscar`Deciding whether to use a given poscar as the beginning of relaxation. If it's true, keys (*cell_type*, *latt*) will be aborted. Otherwise, these two keys are **necessary**.**from_poscar_path:**

type: str, optional

argument path: `init_surf_jdata/from_poscar_path`Path of POSCAR for VASP or STRU for ABACUS. **Necessary** if *from_poscar* is true.**latt:**

type: float

argument path: `init_surf_jdata/latt`

Lattice constant for single cell.

layer_numb:

type: int, optional

argument path: `init_surf_jdata/layer_numb`

Number of atom layers constructing the slab.

z_min:

type: int, optional

argument path: `init_surf_jdata/z_min`Thickness of slab without vacuum (Angstrom). If *layer_numb* is set, *z_min* will be ignored.**vacuum_max:**

type: float

argument path: `init_surf_jdata/vacuum_max`

Maximal thickness of vacuum (Angstrom).

vacuum_min:

type: float, optional

argument path: `init_surf_jdata/vacuum_min`

Minimal thickness of vacuum (Angstrom). Default value is 2 times atomic radius.

vacuum_resol:

type: list[float]

argument path: `init_surf_jdata/vacuum_resol`Interval of thickness of vacuum. If size of *vacuum_resol* is 1, the interval is fixed to its value. If size of *vacuum_resol* is 2, the interval is *vacuum_resol*[0] before *mid_point*, otherwise *vacuum_resol*[1] after *mid_point*.

vacuum_numb:

type: int, optional

argument path: `init_surf_jdata/vacuum_numb`

The total number of vacuum layers **Necessary** if `vacuum_resol` is empty.

mid_point:

type: float, optional

argument path: `init_surf_jdata/mid_point`

The mid point separating head region and tail region. **Necessary** if the size of `vacuum_resol` is 2 or 0.

head_ratio:

type: float, optional

argument path: `init_surf_jdata/head_ratio`

Ratio of vacuum layers in the nearby region with denser intervals(head region). **Necessary** if `vacuum_resol` is empty.

millers:

type: `list[list[int]]`

argument path: `init_surf_jdata/millers`

Miller indices.

relax_incar:

type: str, optional

argument path: `init_surf_jdata/relax_incar`

Path of INCAR for relaxation in VASP. **Necessary** if *stages* include 1.

scale:

type: `list[float]`

argument path: `init_surf_jdata/scale`

Scales for isotropic transforming cells.

skip_relax:

type: bool

argument path: `init_surf_jdata/skip_relax`

If it's true, you may directly run stage 2 (perturb and scale) using an unrelaxed POSCAR.

pert_numb:

type: int

argument path: `init_surf_jdata/pert_numb`

Number of perturbations for each scaled (key *scale*) POSCAR.

pert_box:

type: float

argument path: `init_surf_jdata/pert_box`

Anisotropic Perturbation for cells (independent changes of lengths of three box vectors as well as angle among) in decimal formats. 9 elements of the 3x3 perturbation matrix will be randomly sampled from a uniform distribution (default)

in the range $[-\text{pert_box}, \text{pert_box}]$. Such a perturbation matrix adds the identity matrix gives the actual transformation matrix for this perturbation operation.

pert_atom:

type: float

argument path: `init_surf_jdata/pert_atom`

Perturbation of atom coordinates (Angstrom). Random perturbations are performed on three coordinates of each atom by adding values randomly sampled from a uniform distribution in the range $[-\text{pert_atom}, \text{pert_atom}]$.

coll_ndata:

type: int

argument path: `init_surf_jdata/coll_ndata`

Maximal number of collected data.

5.6 dpngen init_surf machine parameters

Note: One can load, modify, and export the input file by using our effective web-based tool [DP-GUI](#) online or hosted using the *command line interface* `dpngen gui`. All parameters below can be set in DP-GUI. By clicking “SAVE JSON”, one can download the input file.

init_surf_mdata:

type: dict

argument path: `init_surf_mdata`

machine.json file

api_version:

type: str, optional, default: 1.0

argument path: `init_surf_mdata/api_version`

Please set to 1.0

deepmd_version:

type: str, optional, default: 2

argument path: `init_surf_mdata/deepmd_version`

DeePMD-kit version, e.g. 2.1.3

fp:

type: dict

argument path: `init_surf_mdata/fp`

Parameters of command, machine, and resources for fp

command:

type: str

argument path: `init_surf_mdata/fp/command`

Command of a program.

machine:

type: dict

argument path: `init_surf_mdata/fp/machine`

batch_type:

type: str

argument path: `init_surf_mdata/fp/machine/batch_type`

The batch job system type. Option: Bohrium, PBS, OpenAPI, DistributedShell, LSF, SlurmJobArray, Slurm, SGE, Fugaku, Torque, Shell

local_root:

type: str | NoneType

argument path: `init_surf_mdata/fp/machine/local_root`

The dir where the tasks and relating files locate. Typically the project dir.

remote_root:

type: str | NoneType, optional

argument path: `init_surf_mdata/fp/machine/remote_root`

The dir where the tasks are executed on the remote machine. Only needed when context is not lazy-local.

clean_asynchronously:

type: bool, optional, default: False

argument path: `init_surf_mdata/fp/machine/clean_asynchronously`

Clean the remote directory asynchronously after the job finishes.

Depending on the value of *context_type*, different sub args are accepted.

context_type:

type: str (flag key)

argument path: `init_surf_mdata/fp/machine/context_type`

possible choices: *SSHContext*, *LazyLocalContext*, *HDFSContext*, *BohriumContext*, *OpenAPIContext*, *LocalContext*

The connection used to remote machine. Option: LocalContext, SSHContext, OpenAPI-

Context, BohriumContext, LazyLocalContext,
HDFSContext

When `context_type` is set to `SSHContext` (or its aliases `sshcontext`, `SSH`, `ssh`):

remote_profile:

type: dict
argument path: `init_surf_mdata/fp/machine[SSHContext]/remote_profile`

The information used to maintain the connection with remote machine.

hostname:

type: str
argument path: `init_surf_mdata/fp/machine[SSHContext]/remote_profile/hostname`
hostname or ip of ssh connection.

username:

type: str
argument path: `init_surf_mdata/fp/machine[SSHContext]/remote_profile/username`
username of target linux system

password:

type: str, optional
argument path: `init_surf_mdata/fp/machine[SSHContext]/remote_profile/password`
(deprecated) password of linux system. Please use [SSH keys](#) instead to improve security.

port:

type: int, optional, default: 22
argument path: `init_surf_mdata/fp/machine[SSHContext]/remote_profile/port`
ssh connection port.

key_filename:

type: str | NoneType, optional,
default: None
argument path: `init_surf_mdata/fp/machine[SSHContext]/remote_profile/key_filename`
key filename used by ssh connection.
If left None, find key in `~/.ssh` or use password for login

passphrase:

type: str | NoneType, optional,
default: None
argument path: init_surf_mdata/
fp/machine[SSHContext]/
remote_profile/passphrase
passphrase of key used by ssh connection

timeout:

type: int, optional, default: 10
argument path: init_surf_mdata/
fp/machine[SSHContext]/
remote_profile/timeout
timeout of ssh connection

totp_secret:

type: str | NoneType, optional,
default: None
argument path: init_surf_mdata/
fp/machine[SSHContext]/
remote_profile/totp_secret
Time-based one time password secret. It should be a base32-encoded string extracted from the 2D code.

tar_compress:

type: bool, optional, default: True
argument path: init_surf_mdata/
fp/machine[SSHContext]/
remote_profile/tar_compress
The archive will be compressed in upload and download if it is True. If not, compression will be skipped.

look_for_keys:

type: bool, optional, default: True
argument path: init_surf_mdata/
fp/machine[SSHContext]/
remote_profile/look_for_keys
enable searching for discoverable private key files in ~/.ssh/

When `context_type` is set to `LazyLocalContext` (or its aliases `lazylocalcontext`, `LazyLocal`, `lazylocal`):

remote_profile:

type: dict, optional
argument path: init_surf_mdata/fp/
machine[LazyLocalContext]/
remote_profile

The information used to maintain the connection with remote machine. This field is empty for this context.

When `context_type` is set to `HDFSContext` (or its aliases `hdfscontext`, `HDFS`, `hdfs`):

remote_profile:

type: dict, optional
 argument path: `init_surf_mdata/fp/machine[HDFSContext]/remote_profile`

The information used to maintain the connection with remote machine. This field is empty for this context.

When `context_type` is set to `BohriumContext` (or its aliases `bohriumcontext`, `Bohrium`, `bohrium`, `DpCloudServerContext`, `dpcloudservercontext`, `DpCloudServer`, `dpcloudserver`, `LebesgueContext`, `lebesguecontext`, `Lebesgue`, `lebesgue`):

remote_profile:

type: dict
 argument path: `init_surf_mdata/fp/machine[BohriumContext]/remote_profile`

The information used to maintain the connection with remote machine.

email:

type: str, optional
 argument path: `init_surf_mdata/fp/machine[BohriumContext]/remote_profile/email`

Email

password:

type: str, optional
 argument path: `init_surf_mdata/fp/machine[BohriumContext]/remote_profile/password`

Password

program_id:

type: int, alias: *project_id*
 argument path: `init_surf_mdata/fp/machine[BohriumContext]/remote_profile/program_id`

Program ID

retry_count:

type: NoneType | int, optional,
default: 2
argument path: init_surf_mdata/
fp/machine[BohriumContext]/
remote_profile/retry_count

The retry count when a job is terminated

ignore_exit_code:

type: bool, optional, default: True
argument path: init_surf_mdata/
fp/machine[BohriumContext]/
remote_profile/
ignore_exit_code

The job state will be marked as finished if the exit code is non-zero when set to True. Otherwise,
the job state will be designated as terminated.

keep_backup:

type: bool, optional
argument path: init_surf_mdata/
fp/machine[BohriumContext]/
remote_profile/keep_backup

keep download and upload zip

input_data:

type: dict
argument path: init_surf_mdata/
fp/machine[BohriumContext]/
remote_profile/input_data

Configuration of job

When `context_type` is set to `OpenAPIContext` (or its aliases `openapicontext`, `OpenAPI`, `openapi`):

remote_profile:

type: dict, optional
argument path: init_surf_mdata/fp/
machine[OpenAPIContext]/
remote_profile

The information used to maintain the connection with remote machine. This field is empty for this context.

When `context_type` is set to `LocalContext` (or its aliases `localcontext`, `Local`, `local`):

remote_profile:

type: dict, optional

argument path: `init_surf_mdata/fp/machine[LocalContext]/remote_profile`

The information used to maintain the connection with remote machine. This field is empty for this context.

resources:

type: dict

argument path: `init_surf_mdata/fp/resources`

number_node:

type: int, optional, default: 1

argument path: `init_surf_mdata/fp/resources/number_node`

The number of node need for each *job*

cpu_per_node:

type: int, optional, default: 1

argument path: `init_surf_mdata/fp/resources/cpu_per_node`

cpu numbers of each node assigned to each job.

gpu_per_node:

type: int, optional, default: 0

argument path: `init_surf_mdata/fp/resources/gpu_per_node`

gpu numbers of each node assigned to each job.

queue_name:

type: str, optional, default: (empty string)

argument path: `init_surf_mdata/fp/resources/queue_name`

The queue name of batch job scheduler system.

group_size:

type: int

argument path: `init_surf_mdata/fp/resources/group_size`

The number of *tasks* in a *job*. 0 means infinity.

custom_flags:

type: typing.List[str], optional

argument path: `init_surf_mdata/fp/resources/custom_flags`

The extra lines pass to job submitting script header

strategy:

type: dict, optional

argument path: `init_surf_mdata/fp/resources/strategy`

strategies we use to generation job submitting scripts.

if_cuda_multi_devices:

type: bool, optional, default: False

argument path: `init_surf_mdata/fp/resources/strategy/if_cuda_multi_devices`

If there are multiple nvidia GPUS on the node, and we want to assign the tasks to different GPUS. If true, `dpdispatcher` will manually export environment variable `CUDA_VISIBLE_DEVICES` to different task. Usually, this option will be used with `Task.task_need_resources` variable simultaneously.

ratio_unfinished:

type: float, optional, default: 0.0

argument path: `init_surf_mdata/fp/resources/strategy/ratio_unfinished`

The ratio of *tasks* that can be unfinished.

customized_script_header_template_file:

type: str, optional

argument path: `init_surf_mdata/fp/resources/strategy/customized_script_header_template_file`

The customized template file to generate job submitting script header, which overrides the default file.

para_deg:

type: int, optional, default: 1

argument path: `init_surf_mdata/fp/resources/para_deg`

Decide how many tasks will be run in parallel.

source_list:

type: `typing.List[str]`, optional, default: []

argument path: `init_surf_mdata/fp/resources/source_list`

The env file to be sourced before the command execution.

module_purge:

type: bool, optional, default: False
 argument path: init_surf_mdata/fp/
 resources/module_purge

Remove all modules on HPC system before module load (module_list)

module_unload_list:

type: typing.List[str], optional, default: []
 argument path: init_surf_mdata/fp/
 resources/module_unload_list

The modules to be unloaded on HPC system before submitting jobs

module_list:

type: typing.List[str], optional, default: []
 argument path: init_surf_mdata/fp/
 resources/module_list

The modules to be loaded on HPC system before submitting jobs

envs:

type: dict, optional, default: {}
 argument path:
 init_surf_mdata/fp/resources/envs

The environment variables to be exported on before submitting jobs

prepend_script:

type: typing.List[str], optional, default: []
 argument path: init_surf_mdata/fp/
 resources/prepend_script

Optional script run before jobs submitted.

append_script:

type: typing.List[str], optional, default: []
 argument path: init_surf_mdata/fp/
 resources/append_script

Optional script run after jobs submitted.

wait_time:

type: float | int, optional, default: 0
argument path: `init_surf_mdata/fp/
resources/wait_time`

The waiting time in second after a single *task* submitted

Depending on the value of *batch_type*, different sub args are accepted.

batch_type:

type: str (flag key)
argument path: `init_surf_mdata/fp/
resources/batch_type`
possible choices: *Shell*, *PBS*,
DistributedShell, *SlurmJobArray*,
Slurm, *OpenAPI*, *Bohrium*, *SGE*, *Torque*,
Fugaku, *LSF*

The batch job system type loaded from machine/*batch_type*.

When *batch_type* is set to *Shell* (or its alias *shell*):

kwargs:

type: dict, optional
argument path: `init_surf_mdata/fp/
resources[Shell]/kwargs`

This field is empty for this batch.

When *batch_type* is set to *PBS* (or its alias *pbs*):

kwargs:

type: dict, optional
argument path: `init_surf_mdata/fp/
resources[PBS]/kwargs`

This field is empty for this batch.

When *batch_type* is set to *DistributedShell* (or its alias *distributedshell*):

kwargs:

type: dict, optional
argument path: `init_surf_mdata/fp/
resources[DistributedShell]/kwargs`

This field is empty for this batch.

When *batch_type* is set to *SlurmJobArray* (or its alias *slurmjobarray*):

kwargs:

type: dict, optional

argument path: `init_surf_mdata/fp/resources[SlurmJobArray]/kwargs`

Extra arguments.

custom_gpu_line:

type: `str | NoneType`, optional,
default: `None`

argument path: `init_surf_mdata/fp/resources[SlurmJobArray]/kwargs/custom_gpu_line`

Custom GPU configuration, starting with `#SBATCH`

slurm_job_size:

type: `int`, optional, default: `1`

argument path: `init_surf_mdata/fp/resources[SlurmJobArray]/kwargs/slurm_job_size`

Number of tasks in a Slurm job

When `batch_type` is set to `Slurm` (or its alias `slurm`):

kwargs:

type: `dict`, optional

argument path: `init_surf_mdata/fp/resources[Slurm]/kwargs`

Extra arguments.

custom_gpu_line:

type: `str | NoneType`, optional,
default: `None`

argument path: `init_surf_mdata/fp/resources[Slurm]/kwargs/custom_gpu_line`

Custom GPU configuration, starting with `#SBATCH`

When `batch_type` is set to `OpenAPI` (or its alias `openapi`):

kwargs:

type: `dict`, optional

argument path: `init_surf_mdata/fp/resources[OpenAPI]/kwargs`

This field is empty for this batch.

When `batch_type` is set to `Bohrium` (or its aliases `bohrium`, `Lebesgue`, `lebesgue`, `DpCloudServer`, `dpcloudserver`):

kwargs:

type: `dict`, optional

argument path: `init_surf_mdata/fp/resources[Bohrium]/kwargs`

This field is empty for this batch.

When `batch_type` is set to SGE (or its alias `sge`):

kwargs:

type: dict, optional

argument path: `init_surf_mdata/fp/resources[SGE]/kwargs`

This field is empty for this batch.

When `batch_type` is set to Torque (or its alias `torque`):

kwargs:

type: dict, optional

argument path: `init_surf_mdata/fp/resources[Torque]/kwargs`

This field is empty for this batch.

When `batch_type` is set to Fugaku (or its alias `fugaku`):

kwargs:

type: dict, optional

argument path: `init_surf_mdata/fp/resources[Fugaku]/kwargs`

This field is empty for this batch.

When `batch_type` is set to LSF (or its alias `lsf`):

kwargs:

type: dict

argument path: `init_surf_mdata/fp/resources[LSF]/kwargs`

Extra arguments.

gpu_usage:

type: bool, optional, default: False

argument path: `init_surf_mdata/fp/resources[LSF]/kwargs/gpu_usage`

Choosing if GPU is used in the calculation step.

gpu_new_syntax:

type: bool, optional, default: False

argument path: `init_surf_mdata/fp/resources[LSF]/kwargs/gpu_new_syntax`

For LFS \geq 10.1.0.3, new option `-gpu` for `#BSUB` could be used. If False, and old syntax would be used.

gpu_exclusive:

type: bool, optional, default: True
 argument path: `init_surf_mdata/`
`fp/resources[LSF]/kwargs/`
`gpu_exclusive`

Only take effect when new syntax enabled. Control whether submit tasks in exclusive way for GPU.

custom_gpu_line:

type: str | NoneType, optional,
 default: None
 argument path: `init_surf_mdata/`
`fp/resources[LSF]/kwargs/`
`custom_gpu_line`

Custom GPU configuration, starting with `#BSUB`

user_forward_files:

type: list, optional
 argument path:
`init_surf_mdata/fp/user_forward_files`

Files to be forwarded to the remote machine.

user_backward_files:

type: list, optional
 argument path:
`init_surf_mdata/fp/user_backward_files`

Files to be backwarded from the remote machine.

5.7 init_reaction

`dpgen init_reaction` is a workflow to initialize data for reactive systems of small gas-phase molecules. The workflow was introduced in the “Initialization” section of [Energy & Fuels, 2021, 35 \(1\), 762–769](#).

To start the workflow, one needs a box containing reactive systems. The following packages are required for each of the step:

- Exploring: [LAMMPS](#)
- Sampling: [MDDatasetBuilder](#)
- Labeling: [Gaussian](#)

The Exploring step uses LAMMPS `pair_style reaxff` to run a short ReaxMD NVT MD simulation. In the Sampling step, molecular clusters are taken and k-means clustering algorithm is applied to remove the redundancy, which is described in [Nature Communications, 11, 5713 \(2020\)](#). The Labeling step calculates energies and forces using the Gaussian package.

An example of `reaction.json` is given below:

```

1 {
2   "type_map": [
3     "H",
4     "O"
5   ],
6   "reaxff": {
7     "data": "data.hydrogen",
8     "ff": "ffield.reax.cho",
9     "control": "lmp_control",
10    "temp": 3000,
11    "tau_t": 100,
12    "dt": 0.1,
13    "nstep": 100000,
14    "dump_freq": 100
15  },
16  "cutoff": 3.5,
17  "dataset_size": 100,
18  "qmkeywords": "b3lyp/6-31g** force Geom=PrintInputOrient"
19 }
```

For detailed parameters, see *parameters* and *machine parameters*.

The generated data can be used to continue DP-GEN concurrent learning workflow. Read *Energy & Fuels*, 2021, 35 (1), 762–769 for details.

5.8 dpgen init_reaction parameters

Note: One can load, modify, and export the input file by using our effective web-based tool [DP-GUI](#) online or hosted using the *command line interface* `dpgen gui`. All parameters below can be set in DP-GUI. By clicking “SAVE JSON”, one can download the input file.

init_reaction_jdata:

type: dict

argument path: `init_reaction_jdata`

Generate initial data for reactive systems for small gas-phase molecules, from a ReaxFF NVT MD trajectory.

type_map:

type: list[str]

argument path: `init_reaction_jdata/type_map`

Type map, which should match types in the initial data. e.g. [“C”, “H”, “O”]

reaxff:

type: dict

argument path: `init_reaction_jdata/reaxff`

Parameters for ReaxFF NVT MD.

data:

type: str

argument path: init_reaction_jdata/reaxff/data

Path to initial LAMMPS data file. The atom_style should be charge.

ff:

type: str

argument path: init_reaction_jdata/reaxff/ff

Path to ReaxFF force field file. Available in the lammmps/potentials directory.

control:

type: str

argument path: init_reaction_jdata/reaxff/control

Path to ReaxFF control file.

temp:

type: int | float

argument path: init_reaction_jdata/reaxff/temp

Target Temperature for the NVT MD simulation. Unit: K.

dt:

type: int | float

argument path: init_reaction_jdata/reaxff/dt

Real time for every time step. Unit: fs.

tau_t:

type: int | float

argument path: init_reaction_jdata/reaxff/tau_t

Time to determine how rapidly the temperature. Unit: fs.

dump_freq:

type: int

argument path:

init_reaction_jdata/reaxff/dump_freq

Frequency of time steps to collect trajectory.

nstep:

type: int

argument path: init_reaction_jdata/reaxff/nstep

Total steps to run the ReaxFF MD simulation.

cutoff:

type: float

argument path: `init_reaction_jdata/cutoff`

Cutoff radius to take clusters from the trajectory. Note that only a complete molecule or free radical will be taken.

dataset_size:

type: `int`

argument path: `init_reaction_jdata/dataset_size`

Collected dataset size for each bond type.

qmkeywords:

type: `str`

argument path: `init_reaction_jdata/qmkeywords`

Gaussian keywords for first-principle calculations. e.g. `force mn15/6-31g**`
`Geom=PrintInputOrient`. Note that “force” job is necessary to collect data.
`Geom=PrintInputOrient` should be used when there are more than 50 atoms in a cluster.

5.9 dpgen `init_reaction` machine parameters

Note: One can load, modify, and export the input file by using our effective web-based tool [DP-GUI](#) online or hosted using the *command line interface* `dpgen gui`. All parameters below can be set in DP-GUI. By clicking “SAVE JSON”, one can download the input file.

init_reaction_mdata:

type: `dict`

argument path: `init_reaction_mdata`

machine.json file

api_version:

type: `str`, optional, default: `1.0`

argument path: `init_reaction_mdata/api_version`

Please set to `1.0`

deepmd_version:

type: `str`, optional, default: `2`

argument path: `init_reaction_mdata/deepmd_version`

DeePMD-kit version, e.g. `2.1.3`

reaxff:

type: `dict`

argument path: `init_reaction_mdata/reaxff`

Parameters of command, machine, and resources for reaxff

command:

type: `str`

argument path: `init_reaction_mdata/reaxff/command`

Command of a program.

machine:

type: dict

argument path: `init_reaction_mdata/reaxff/machine`

batch_type:

type: str

argument path: `init_reaction_mdata/reaxff/machine/batch_type`

The batch job system type. Option: Bohrium, PBS, OpenAPI, DistributedShell, LSF, SlurmJobArray, Slurm, SGE, Fugaku, Torque, Shell

local_root:

type: str | NoneType

argument path: `init_reaction_mdata/reaxff/machine/local_root`

The dir where the tasks and relating files locate. Typically the project dir.

remote_root:

type: str | NoneType, optional

argument path: `init_reaction_mdata/reaxff/machine/remote_root`

The dir where the tasks are executed on the remote machine. Only needed when context is not lazy-local.

clean_asynchronously:

type: bool, optional, default: False

argument path: `init_reaction_mdata/reaxff/machine/clean_asynchronously`

Clean the remote directory asynchronously after the job finishes.

Depending on the value of *context_type*, different sub args are accepted.

context_type:

type: str (flag key)

argument path: `init_reaction_mdata/reaxff/machine/context_type`

possible choices: *SSHContext*, *LazyLocalContext*, *HDFSContext*, *BohriumContext*, *OpenAPIContext*, *LocalContext*

The connection used to remote machine. Option: LocalContext, SSHContext, OpenAPI-

Context, BohriumContext, LazyLocalContext,
HDFSContext

When `context_type` is set to `SSHContext` (or its aliases
`sshcontext`, `SSH`, `ssh`):

remote_profile:

type: dict
argument path:
`init_reaction_mdata/reaxff/
machine[SSHContext]/remote_profile`

The information used to maintain the connection with remote machine.

hostname:

type: str
argument path:
`init_reaction_mdata/reaxff/
machine[SSHContext]/
remote_profile/hostname`
hostname or ip of ssh connection.

username:

type: str
argument path:
`init_reaction_mdata/reaxff/
machine[SSHContext]/
remote_profile/username`
username of target linux system

password:

type: str, optional
argument path:
`init_reaction_mdata/reaxff/
machine[SSHContext]/
remote_profile/password`
(deprecated) password of linux system. Please use [SSH keys](#) instead to improve security.

port:

type: int, optional, default: 22
argument path:
`init_reaction_mdata/reaxff/
machine[SSHContext]/
remote_profile/port`
ssh connection port.

key_filename:

type: str | NoneType, optional,
default: None

argument path:
 init_reaction_mdata/reaxff/
 machine[SSHContext]/
 remote_profile/key_filename
 key filename used by ssh connection.
 If left None, find key in ~/.ssh or use
 password for login

passphrase:

type: str | NoneType, optional,
 default: None
 argument path:
 init_reaction_mdata/reaxff/
 machine[SSHContext]/
 remote_profile/passphrase
 passphrase of key used by ssh connection

timeout:

type: int, optional, default: 10
 argument path:
 init_reaction_mdata/reaxff/
 machine[SSHContext]/
 remote_profile/timeout
 timeout of ssh connection

totp_secret:

type: str | NoneType, optional,
 default: None
 argument path:
 init_reaction_mdata/reaxff/
 machine[SSHContext]/
 remote_profile/totp_secret
 Time-based one time password secret. It should be a base32-encoded string extracted from the 2D code.

tar_compress:

type: bool, optional, default: True
 argument path:
 init_reaction_mdata/reaxff/
 machine[SSHContext]/
 remote_profile/tar_compress
 The archive will be compressed in upload and download if it is True. If not, compression will be skipped.

look_for_keys:

type: bool, optional, default: True
 argument path:
 init_reaction_mdata/reaxff/

```
machine[SSHContext]/
remote_profile/look_for_keys

enable searching for discoverable private key files in ~/.ssh/
```

When `context_type` is set to `LazyLocalContext` (or its aliases `lazylocalcontext`, `LazyLocal`, `lazylocal`):

remote_profile:

```
type: dict, optional
argument path: init_reaction_mdata/
reaxff/machine[LazyLocalContext]/
remote_profile
```

The information used to maintain the connection with remote machine. This field is empty for this context.

When `context_type` is set to `HDFSContext` (or its aliases `hdfscontext`, `HDFS`, `hdfs`):

remote_profile:

```
type: dict, optional
argument path:
init_reaction_mdata/reaxff/
machine[HDFSContext]/remote_profile
```

The information used to maintain the connection with remote machine. This field is empty for this context.

When `context_type` is set to `BohriumContext` (or its aliases `bohriumcontext`, `Bohrium`, `bohrium`, `DpCloudServerContext`, `dpcloudservercontext`, `DpCloudServer`, `dpcloudserver`, `LebesgueContext`, `lebesguecontext`, `Lebesgue`, `lebesgue`):

remote_profile:

```
type: dict
argument path: init_reaction_mdata/
reaxff/machine[BohriumContext]/
remote_profile
```

The information used to maintain the connection with remote machine.

email:

```
type: str, optional
argument path:
init_reaction_mdata/reaxff/
machine[BohriumContext]/
remote_profile/email
```

Email

password:

type: str, optional
 argument path:
 init_reaction_mdata/reaxff/
 machine[BohriumContext]/
 remote_profile/password
 Password

program_id:

type: int, alias: *project_id*
 argument path:
 init_reaction_mdata/reaxff/
 machine[BohriumContext]/
 remote_profile/program_id
 Program ID

retry_count:

type: NoneType | int, optional,
 default: 2
 argument path:
 init_reaction_mdata/reaxff/
 machine[BohriumContext]/
 remote_profile/retry_count
 The retry count when a job is terminated

ignore_exit_code:

type: bool, optional, default: True
 argument path:
 init_reaction_mdata/reaxff/
 machine[BohriumContext]/
 remote_profile/
 ignore_exit_code

The job state will be marked as finished if the exit code is non-zero when set to True. Otherwise,
 the job state will be designated as terminated.

keep_backup:

type: bool, optional
 argument path:
 init_reaction_mdata/reaxff/
 machine[BohriumContext]/
 remote_profile/keep_backup
 keep download and upload zip

input_data:

type: dict
 argument path:
 init_reaction_mdata/reaxff/

```
machine[BohriumContext]/  
remote_profile/input_data
```

Configuration of job

When `context_type` is set to `OpenAPIContext` (or its aliases `openapicontext`, `OpenAPI`, `openapi`):

remote_profile:

```
type: dict, optional  
argument path: init_reaction_mdata/  
reaxff/machine[OpenAPIContext]/  
remote_profile
```

The information used to maintain the connection with remote machine. This field is empty for this context.

When `context_type` is set to `LocalContext` (or its aliases `localcontext`, `Local`, `local`):

remote_profile:

```
type: dict, optional  
argument path: init_reaction_mdata/  
reaxff/machine[LocalContext]/  
remote_profile
```

The information used to maintain the connection with remote machine. This field is empty for this context.

resources:

```
type: dict  
argument path:  
init_reaction_mdata/reaxff/resources
```

number_node:

```
type: int, optional, default: 1  
argument path: init_reaction_mdata/  
reaxff/resources/number_node
```

The number of node need for each *job*

cpu_per_node:

```
type: int, optional, default: 1  
argument path: init_reaction_mdata/  
reaxff/resources/cpu_per_node
```

cpu numbers of each node assigned to each job.

gpu_per_node:

```
type: int, optional, default: 0  
argument path: init_reaction_mdata/  
reaxff/resources/gpu_per_node
```

gpu numbers of each node assigned to each job.

queue_name:

type: str, optional, default: (empty string)
 argument path: init_reaction_mdata/
 reaxff/resources/queue_name

The queue name of batch job scheduler system.

group_size:

type: int
 argument path: init_reaction_mdata/
 reaxff/resources/group_size

The number of *tasks* in a *job*. 0 means infinity.

custom_flags:

type: typing.List[str], optional
 argument path: init_reaction_mdata/
 reaxff/resources/custom_flags

The extra lines pass to job submitting script header

strategy:

type: dict, optional
 argument path: init_reaction_mdata/
 reaxff/resources/strategy

strategies we use to generation job submitting scripts.

if_cuda_multi_devices:

type: bool, optional, default: False
 argument path:
 init_reaction_mdata/reaxff/
 resources/strategy/
 if_cuda_multi_devices

If there are multiple nvidia GPUS on the node, and we want to assign the tasks to different GPUS. If true, dpdispatcher will manually export environment variable CUDA_VISIBLE_DEVICES to different task. Usually, this option will be used with Task.task_need_resources variable simultaneously.

ratio_unfinished:

type: float, optional, default: 0.0
 argument path:
 init_reaction_mdata/reaxff/
 resources/strategy/
 ratio_unfinished

The ratio of *tasks* that can be unfinished.

customized_script_header_template_file:

type: str, optional
argument path:
init_reaction_mdata/reaxff/
resources/strategy/
customized_script_header_template_file

The customized template file to generate job submitting script header, which overrides the default file.

para_deg:

type: int, optional, default: 1
argument path: init_reaction_mdata/
reaxff/resources/para_deg

Decide how many tasks will be run in parallel.

source_list:

type: typing.List[str], optional, default:
[]
argument path: init_reaction_mdata/
reaxff/resources/source_list

The env file to be sourced before the command execution.

module_purge:

type: bool, optional, default: False
argument path: init_reaction_mdata/
reaxff/resources/module_purge

Remove all modules on HPC system before module load (module_list)

module_unload_list:

type: typing.List[str], optional, default:
[]
argument path: init_reaction_mdata/
reaxff/resources/module_unload_list

The modules to be unloaded on HPC system before submitting jobs

module_list:

type: typing.List[str], optional, default:
[]
argument path: init_reaction_mdata/
reaxff/resources/module_list

The modules to be loaded on HPC system before submitting jobs

envs:

type: dict, optional, default: {}
 argument path: `init_reaction_mdata/`
`reaxff/resources/envs`

The environment variables to be exported on
 before submitting jobs

prepend_script:

type: `typing.List[str]`, optional, default:
`[]`

argument path: `init_reaction_mdata/`
`reaxff/resources/prepend_script`

Optional script run before jobs submitted.

append_script:

type: `typing.List[str]`, optional, default:
`[]`

argument path: `init_reaction_mdata/`
`reaxff/resources/append_script`

Optional script run after jobs submitted.

wait_time:

type: `float | int`, optional, default: `0`
 argument path: `init_reaction_mdata/`
`reaxff/resources/wait_time`

The waiting time in second after a single *task*
 submitted

Depending on the value of *batch_type*, different sub args are
 accepted.

batch_type:

type: str (flag key)
 argument path: `init_reaction_mdata/`
`reaxff/resources/batch_type`
 possible choices: *Shell*, *PBS*,
DistributedShell, *SlurmJobArray*,
Slurm, *OpenAPI*, *Bohrium*, *SGE*, *Torque*,
Fugaku, *LSF*

The batch job system type loaded from ma-
 chine/batch_type.

When *batch_type* is set to *Shell* (or its alias *shell*):

kwargs:

type: dict, optional
 argument path: `init_reaction_mdata/`
`reaxff/resources[Shell]/kwargs`

This field is empty for this batch.

When `batch_type` is set to PBS (or its alias pbs):

kwargs:

type: dict, optional
argument path: `init_reaction_mdata/
reaxff/resources[PBS]/kwargs`

This field is empty for this batch.

When `batch_type` is set to DistributedShell (or its alias distributedshell):

kwargs:

type: dict, optional
argument path:
`init_reaction_mdata/reaxff/
resources[DistributedShell]/kwargs`

This field is empty for this batch.

When `batch_type` is set to SlurmJobArray (or its alias slurmjobarray):

kwargs:

type: dict, optional
argument path:
`init_reaction_mdata/reaxff/
resources[SlurmJobArray]/kwargs`

Extra arguments.

custom_gpu_line:

type: str | NoneType, optional,
default: None
argument path:
`init_reaction_mdata/reaxff/
resources[SlurmJobArray]/
kwargs/custom_gpu_line`

Custom GPU configuration, starting
with #SBATCH

slurm_job_size:

type: int, optional, default: 1
argument path:
`init_reaction_mdata/reaxff/
resources[SlurmJobArray]/
kwargs/slurm_job_size`

Number of tasks in a Slurm job

When `batch_type` is set to Slurm (or its alias slurm):

kwargs:

type: dict, optional

argument path: `init_reaction_mdata/
reaxff/resources[Slurm]/kwargs`

Extra arguments.

custom_gpu_line:

type: `str | NoneType`, optional,
default: `None`

argument path:
`init_reaction_mdata/reaxff/
resources[Slurm]/kwargs/
custom_gpu_line`

Custom GPU configuration, starting
with `#SBATCH`

When `batch_type` is set to `OpenAPI` (or its alias `openapi`):

kwargs:

type: `dict`, optional

argument path: `init_reaction_mdata/
reaxff/resources[OpenAPI]/kwargs`

This field is empty for this batch.

When `batch_type` is set to `Bohrium` (or its aliases `bohrium`,
`Lebesgue`, `lebesgue`, `DpCloudServer`, `dpcloudserver`):

kwargs:

type: `dict`, optional

argument path: `init_reaction_mdata/
reaxff/resources[Bohrium]/kwargs`

This field is empty for this batch.

When `batch_type` is set to `SGE` (or its alias `sge`):

kwargs:

type: `dict`, optional

argument path: `init_reaction_mdata/
reaxff/resources[SGE]/kwargs`

This field is empty for this batch.

When `batch_type` is set to `Torque` (or its alias `torque`):

kwargs:

type: `dict`, optional

argument path: `init_reaction_mdata/
reaxff/resources[Torque]/kwargs`

This field is empty for this batch.

When `batch_type` is set to `Fugaku` (or its alias `fugaku`):

kwargs:

type: `dict`, optional

argument path: `init_reaction_mdata/
reaxff/resources[Fugaku]/kwargs`

This field is empty for this batch.

When `batch_type` is set to LSF (or its alias `lsf`):

kwargs:

type: dict

argument path: `init_reaction_mdata/
reaxff/resources[LSF]/kwargs`

Extra arguments.

gpu_usage:

type: bool, optional, default: False

argument path:

`init_reaction_mdata/reaxff/
resources[LSF]/kwargs/
gpu_usage`

Choosing if GPU is used in the calculation step.

gpu_new_syntax:

type: bool, optional, default: False

argument path:

`init_reaction_mdata/reaxff/
resources[LSF]/kwargs/
gpu_new_syntax`

For LFS $\geq 10.1.0.3$, new option `-gpu` for `#BSUB` could be used. If False, and old syntax would be used.

gpu_exclusive:

type: bool, optional, default: True

argument path:

`init_reaction_mdata/reaxff/
resources[LSF]/kwargs/
gpu_exclusive`

Only take effect when new syntax enabled. Control whether submit tasks in exclusive way for GPU.

custom_gpu_line:

type: str | NoneType, optional,
default: None

argument path:

`init_reaction_mdata/reaxff/
resources[LSF]/kwargs/
custom_gpu_line`

Custom GPU configuration, starting with `#BSUB`

user_forward_files:

type: list, optional
 argument path: `init_reaction_mdata/reaxff/user_forward_files`
 Files to be forwarded to the remote machine.

user_backward_files:

type: list, optional
 argument path: `init_reaction_mdata/reaxff/user_backward_files`
 Files to be backwarded from the remote machine.

build:

type: dict
 argument path: `init_reaction_mdata/build`
 Parameters of command, machine, and resources for build

command:

type: str
 argument path: `init_reaction_mdata/build/command`
 Command of a program.

machine:

type: dict
 argument path: `init_reaction_mdata/build/machine`

batch_type:

type: str
 argument path: `init_reaction_mdata/build/machine/batch_type`
 The batch job system type. Option: Bohrium, PBS, OpenAPI, DistributedShell, LSF, SlurmJobArray, Slurm, SGE, Fugaku, Torque, Shell

local_root:

type: str | NoneType
 argument path: `init_reaction_mdata/build/machine/local_root`
 The dir where the tasks and relating files locate.
 Typically the project dir.

remote_root:

type: str | NoneType, optional
 argument path: `init_reaction_mdata/build/machine/remote_root`

The dir where the tasks are executed on the remote machine. Only needed when context is not lazy-local.

clean_asynchronously:

type: bool, optional, default: False
argument path: `init_reaction_mdata/build/machine/clean_asynchronously`

Clean the remote directory asynchronously after the job finishes.

Depending on the value of *context_type*, different sub args are accepted.

context_type:

type: str (flag key)
argument path: `init_reaction_mdata/build/machine/context_type`
possible choices: *SSHContext*,
LazyLocalContext, *HDFSContext*,
BohriumContext, *OpenAPIContext*,
LocalContext

The connection used to remote machine. Option: LocalContext, SSHContext, OpenAPIContext, BohriumContext, LazyLocalContext, HDFSContext

When *context_type* is set to SSHContext (or its aliases *sshcontext*, *SSH*, *ssh*):

remote_profile:

type: dict
argument path:
`init_reaction_mdata/build/machine[SSHContext]/remote_profile`

The information used to maintain the connection with remote machine.

hostname:

type: str
argument path:
`init_reaction_mdata/build/machine[SSHContext]/remote_profile/hostname`
hostname or ip of ssh connection.

username:

type: str
argument path:
`init_reaction_mdata/build/machine[SSHContext]/remote_profile/username`

username of target linux system

password:

type: str, optional

argument path:

init_reaction_mdata/build/

machine[SSHContext]/

remote_profile/password

(deprecated) password of linux system. Please use [SSH keys](#) instead to improve security.

port:

type: int, optional, default: 22

argument path:

init_reaction_mdata/build/

machine[SSHContext]/

remote_profile/port

ssh connection port.

key_filename:

type: str | NoneType, optional,

default: None

argument path:

init_reaction_mdata/build/

machine[SSHContext]/

remote_profile/key_filename

key filename used by ssh connection.

If left None, find key in ~/.ssh or use password for login

passphrase:

type: str | NoneType, optional,

default: None

argument path:

init_reaction_mdata/build/

machine[SSHContext]/

remote_profile/passphrase

passphrase of key used by ssh connection

timeout:

type: int, optional, default: 10

argument path:

init_reaction_mdata/build/

machine[SSHContext]/

remote_profile/timeout

timeout of ssh connection

totp_secret:

type: str | NoneType, optional,

default: None

argument path:
init_reaction_mdata/build/
machine[SSHContext]/
remote_profile/totp_secret

Time-based one time password secret. It should be a base32-encoded string extracted from the 2D code.

tar_compress:

type: bool, optional, default: True
argument path:
init_reaction_mdata/build/
machine[SSHContext]/
remote_profile/tar_compress

The archive will be compressed in upload and download if it is True. If not, compression will be skipped.

look_for_keys:

type: bool, optional, default: True
argument path:
init_reaction_mdata/build/
machine[SSHContext]/
remote_profile/look_for_keys

enable searching for discoverable private key files in ~/.ssh/

When `context_type` is set to `LazyLocalContext` (or its aliases `lazylocalcontext`, `LazyLocal`, `lazylocal`):

remote_profile:

type: dict, optional
argument path: init_reaction_mdata/
build/machine[LazyLocalContext]/
remote_profile

The information used to maintain the connection with remote machine. This field is empty for this context.

When `context_type` is set to `HDFSContext` (or its aliases `hdfscontext`, `HDFS`, `hdfs`):

remote_profile:

type: dict, optional
argument path:
init_reaction_mdata/build/
machine[HDFSContext]/remote_profile

The information used to maintain the connection with remote machine. This field is empty for this context.

When `context_type` is set to `BohriumContext` (or its aliases `bohriumcontext`, `Bohrium`, `bohrium`, `DpCloudServerContext`, `dpcloudservercontext`, `DpCloudServer`, `dpcloudserver`, `LebesgueContext`, `lebesguecontext`, `Lebesgue`, `lebesgue`):

remote_profile:

type: dict
 argument path: `init_reaction_mdata/build/machine[BohriumContext]/remote_profile`

The information used to maintain the connection with remote machine.

email:

type: str, optional
 argument path:
`init_reaction_mdata/build/machine[BohriumContext]/remote_profile/email`

Email

password:

type: str, optional
 argument path:
`init_reaction_mdata/build/machine[BohriumContext]/remote_profile/password`

Password

program_id:

type: int, alias: *project_id*
 argument path:
`init_reaction_mdata/build/machine[BohriumContext]/remote_profile/program_id`

Program ID

retry_count:

type: `NoneType` | int, optional,
 default: 2
 argument path:
`init_reaction_mdata/build/machine[BohriumContext]/remote_profile/retry_count`

The retry count when a job is terminated

ignore_exit_code:

type: bool, optional, default: True

argument path:
init_reaction_mdata/build/
machine[BohriumContext]/
remote_profile/
ignore_exit_code

The job state will be marked as finished if the exit code is non-zero when set to True. Otherwise,
the job state will be designated as terminated.

keep_backup:

type: bool, optional
argument path:
init_reaction_mdata/build/
machine[BohriumContext]/
remote_profile/keep_backup

keep download and upload zip

input_data:

type: dict
argument path:
init_reaction_mdata/build/
machine[BohriumContext]/
remote_profile/input_data

Configuration of job

When `context_type` is set to `OpenAPIContext` (or its aliases `openapicontext`, `OpenAPI`, `openapi`):

remote_profile:

type: dict, optional
argument path: init_reaction_mdata/
build/machine[OpenAPIContext]/
remote_profile

The information used to maintain the connection with remote machine. This field is empty for this context.

When `context_type` is set to `LocalContext` (or its aliases `localcontext`, `Local`, `local`):

remote_profile:

type: dict, optional
argument path: init_reaction_mdata/
build/machine[LocalContext]/
remote_profile

The information used to maintain the connection with remote machine. This field is empty for this context.

resources:

type: dict
 argument path:
 init_reaction_mdata/build/resources

number_node:

type: int, optional, default: 1
 argument path: init_reaction_mdata/
 build/resources/number_node

The number of node need for each *job*

cpu_per_node:

type: int, optional, default: 1
 argument path: init_reaction_mdata/
 build/resources/cpu_per_node

cpu numbers of each node assigned to each job.

gpu_per_node:

type: int, optional, default: 0
 argument path: init_reaction_mdata/
 build/resources/gpu_per_node

gpu numbers of each node assigned to each job.

queue_name:

type: str, optional, default: (empty string)
 argument path: init_reaction_mdata/
 build/resources/queue_name

The queue name of batch job scheduler system.

group_size:

type: int
 argument path: init_reaction_mdata/
 build/resources/group_size

The number of *tasks* in a *job*. 0 means infinity.

custom_flags:

type: typing.List[str], optional
 argument path: init_reaction_mdata/
 build/resources/custom_flags

The extra lines pass to job submitting script header

strategy:

type: dict, optional
 argument path: init_reaction_mdata/
 build/resources/strategy

strategies we use to generation job submitting scripts.

if_cuda_multi_devices:

type: bool, optional, default: False
argument path:
init_reaction_mdata/build/
resources/strategy/
if_cuda_multi_devices

If there are multiple nvidia GPUS on the node, and we want to assign the tasks to different GPUS. If true, dpdispatcher will manually export environment variable CUDA_VISIBLE_DEVICES to different task. Usually, this option will be used with Task.task_need_resources variable simultaneously.

ratio_unfinished:

type: float, optional, default: 0.0
argument path:
init_reaction_mdata/build/
resources/strategy/
ratio_unfinished

The ratio of *tasks* that can be unfinished.

customized_script_header_template_file:

type: str, optional
argument path:
init_reaction_mdata/build/
resources/strategy/
customized_script_header_template_file

The customized template file to generate job submitting script header, which overrides the default file.

para_deg:

type: int, optional, default: 1
argument path: init_reaction_mdata/
build/resources/para_deg

Decide how many tasks will be run in parallel.

source_list:

type: typing.List[str], optional, default: []
argument path: init_reaction_mdata/
build/resources/source_list

The env file to be sourced before the command execution.

module_purge:

type: bool, optional, default: False
argument path: init_reaction_mdata/
build/resources/module_purge
Remove all modules on HPC system before
module load (module_list)

module_unload_list:

type: typing.List[str], optional, default:
[]
argument path: init_reaction_mdata/
build/resources/module_unload_list
The modules to be unloaded on HPC system
before submitting jobs

module_list:

type: typing.List[str], optional, default:
[]
argument path: init_reaction_mdata/
build/resources/module_list
The modules to be loaded on HPC system be-
fore submitting jobs

envs:

type: dict, optional, default: {}
argument path: init_reaction_mdata/
build/resources/envs
The environment variables to be exported on
before submitting jobs

prepend_script:

type: typing.List[str], optional, default:
[]
argument path: init_reaction_mdata/
build/resources/prepend_script
Optional script run before jobs submitted.

append_script:

type: typing.List[str], optional, default:
[]
argument path: init_reaction_mdata/
build/resources/append_script
Optional script run after jobs submitted.

wait_time:

type: float | int, optional, default: 0

argument path: `init_reaction_mdata/
build/resources/wait_time`

The waiting time in second after a single *task* submitted

Depending on the value of *batch_type*, different sub args are accepted.

batch_type:

type: str (flag key)
argument path: `init_reaction_mdata/
build/resources/batch_type`
possible choices: *Shell*, *PBS*,
DistributedShell, *SlurmJobArray*,
Slurm, *OpenAPI*, *Bohrrium*, *SGE*, *Torque*,
Fugaku, *LSF*

The batch job system type loaded from machine/*batch_type*.

When *batch_type* is set to *Shell* (or its alias *shell*):

kwargs:

type: dict, optional
argument path: `init_reaction_mdata/
build/resources[Shell]/kwargs`

This field is empty for this batch.

When *batch_type* is set to *PBS* (or its alias *pbs*):

kwargs:

type: dict, optional
argument path: `init_reaction_mdata/
build/resources[PBS]/kwargs`

This field is empty for this batch.

When *batch_type* is set to *DistributedShell* (or its alias *distributedshell*):

kwargs:

type: dict, optional
argument path:
`init_reaction_mdata/build/
resources[DistributedShell]/kwargs`

This field is empty for this batch.

When *batch_type* is set to *SlurmJobArray* (or its alias *slurmjobarray*):

kwargs:

type: dict, optional

argument path:
 init_reaction_mdata/build/
 resources[SlurmJobArray]/kwargs

Extra arguments.

custom_gpu_line:

type: str | NoneType, optional,
 default: None

argument path:
 init_reaction_mdata/build/
 resources[SlurmJobArray]/
 kwargs/custom_gpu_line

Custom GPU configuration, starting
 with #SBATCH

slurm_job_size:

type: int, optional, default: 1

argument path:
 init_reaction_mdata/build/
 resources[SlurmJobArray]/
 kwargs/slurm_job_size

Number of tasks in a Slurm job

When `batch_type` is set to Slurm (or its alias slurm):

kwargs:

type: dict, optional

argument path: init_reaction_mdata/
 build/resources[Slurm]/kwargs

Extra arguments.

custom_gpu_line:

type: str | NoneType, optional,
 default: None

argument path:
 init_reaction_mdata/build/
 resources[Slurm]/kwargs/
 custom_gpu_line

Custom GPU configuration, starting
 with #SBATCH

When `batch_type` is set to OpenAPI (or its alias openapi):

kwargs:

type: dict, optional

argument path: init_reaction_mdata/
 build/resources[OpenAPI]/kwargs

This field is empty for this batch.

When `batch_type` is set to Bohrium (or its aliases bohrium,
 Lebesgue, lebesgue, DpCloudServer, dpcloudserver):

kwargs:

type: dict, optional
argument path: `init_reaction_mdata/
build/resources[Bohrium]/kwargs`

This field is empty for this batch.

When `batch_type` is set to SGE (or its alias `sge`):

kwargs:

type: dict, optional
argument path: `init_reaction_mdata/
build/resources[SGE]/kwargs`

This field is empty for this batch.

When `batch_type` is set to Torque (or its alias `torque`):

kwargs:

type: dict, optional
argument path: `init_reaction_mdata/
build/resources[Torque]/kwargs`

This field is empty for this batch.

When `batch_type` is set to Fugaku (or its alias `fugaku`):

kwargs:

type: dict, optional
argument path: `init_reaction_mdata/
build/resources[Fugaku]/kwargs`

This field is empty for this batch.

When `batch_type` is set to LSF (or its alias `lsf`):

kwargs:

type: dict
argument path: `init_reaction_mdata/
build/resources[LSF]/kwargs`

Extra arguments.

gpu_usage:

type: bool, optional, default: False
argument path:
`init_reaction_mdata/build/
resources[LSF]/kwargs/
gpu_usage`

Choosing if GPU is used in the calculation step.

gpu_new_syntax:

type: bool, optional, default: False

argument path:
 init_reaction_mdata/build/
 resources[LSF]/kwargs/
 gpu_new_syntax

For LFS >= 10.1.0.3, new option -gpu
 for #BSUB could be used. If False,
 and old syntax would be used.

gpu_exclusive:

type: bool, optional, default: True
 argument path:
 init_reaction_mdata/build/
 resources[LSF]/kwargs/
 gpu_exclusive

Only take effect when new syntax en-
 abled. Control whether submit tasks
 in exclusive way for GPU.

custom_gpu_line:

type: str | NoneType, optional,
 default: None
 argument path:
 init_reaction_mdata/build/
 resources[LSF]/kwargs/
 custom_gpu_line

Custom GPU configuration, starting
 with #BSUB

user_forward_files:

type: list, optional
 argument path:
 init_reaction_mdata/build/user_forward_files
 Files to be forwarded to the remote machine.

user_backward_files:

type: list, optional
 argument path:
 init_reaction_mdata/build/user_backward_files
 Files to be backwarded from the remote machine.

fp:

type: dict
 argument path: init_reaction_mdata/fp
 Parameters of command, machine, and resources for fp

command:

type: str
 argument path: init_reaction_mdata/fp/command
 Command of a program.

machine:

type: dict

argument path: `init_reaction_mdata/fp/machine`

batch_type:

type: str

argument path: `init_reaction_mdata/fp/machine/batch_type`

The batch job system type. Option: Bohrium, PBS, OpenAPI, DistributedShell, LSF, SlurmJobArray, Slurm, SGE, Fugaku, Torque, Shell

local_root:

type: str | NoneType

argument path: `init_reaction_mdata/fp/machine/local_root`

The dir where the tasks and relating files locate. Typically the project dir.

remote_root:

type: str | NoneType, optional

argument path: `init_reaction_mdata/fp/machine/remote_root`

The dir where the tasks are executed on the remote machine. Only needed when context is not lazy-local.

clean_asynchronously:

type: bool, optional, default: False

argument path: `init_reaction_mdata/fp/machine/clean_asynchronously`

Clean the remote directory asynchronously after the job finishes.

Depending on the value of *context_type*, different sub args are accepted.

context_type:

type: str (flag key)

argument path: `init_reaction_mdata/fp/machine/context_type`

possible choices: *SSHContext*, *LazyLocalContext*, *HDFSContext*, *BohriumContext*, *OpenAPIContext*, *LocalContext*

The connection used to remote machine. Option: LocalContext, SSHContext, OpenAPI-

Context, BohriumContext, LazyLocalContext,
HDFSContext

When `context_type` is set to `SSHContext` (or its aliases `sshcontext`, `SSH`, `ssh`):

remote_profile:

type: dict
argument path: `init_reaction_mdata/fp/
machine[SSHContext]/remote_profile`

The information used to maintain the connection with remote machine.

hostname:

type: str
argument path:
`init_reaction_mdata/fp/
machine[SSHContext]/
remote_profile/hostname`
hostname or ip of ssh connection.

username:

type: str
argument path:
`init_reaction_mdata/fp/
machine[SSHContext]/
remote_profile/username`
username of target linux system

password:

type: str, optional
argument path:
`init_reaction_mdata/fp/
machine[SSHContext]/
remote_profile/password`
(deprecated) password of linux system. Please use [SSH keys](#) instead to improve security.

port:

type: int, optional, default: 22
argument path:
`init_reaction_mdata/fp/
machine[SSHContext]/
remote_profile/port`
ssh connection port.

key_filename:

type: str | NoneType, optional,
default: None

argument path:
init_reaction_mdata/fp/
machine[SSHContext]/
remote_profile/key_filename

key filename used by ssh connection.
If left None, find key in ~/.ssh or use
password for login

passphrase:

type: str | NoneType, optional,
default: None

argument path:
init_reaction_mdata/fp/
machine[SSHContext]/
remote_profile/passphrase

passphrase of key used by ssh connection

timeout:

type: int, optional, default: 10

argument path:
init_reaction_mdata/fp/
machine[SSHContext]/
remote_profile/timeout

timeout of ssh connection

totp_secret:

type: str | NoneType, optional,
default: None

argument path:
init_reaction_mdata/fp/
machine[SSHContext]/
remote_profile/totp_secret

Time-based one time password secret. It should be a base32-encoded string extracted from the 2D code.

tar_compress:

type: bool, optional, default: True

argument path:
init_reaction_mdata/fp/
machine[SSHContext]/
remote_profile/tar_compress

The archive will be compressed in upload and download if it is True. If not, compression will be skipped.

look_for_keys:

type: bool, optional, default: True

argument path:
init_reaction_mdata/fp/

```

machine[SSHContext]/
remote_profile/look_for_keys

enable searching for discoverable private key files in ~/.ssh/

```

When `context_type` is set to `LazyLocalContext` (or its aliases `lazylocalcontext`, `LazyLocal`, `lazylocal`):

remote_profile:

```

type: dict, optional
argument path: init_reaction_mdata/fp/
machine[LazyLocalContext]/
remote_profile

```

The information used to maintain the connection with remote machine. This field is empty for this context.

When `context_type` is set to `HDFSContext` (or its aliases `hdfscontext`, `HDFS`, `hdfs`):

remote_profile:

```

type: dict, optional
argument path: init_reaction_mdata/fp/
machine[HDFSContext]/remote_profile

```

The information used to maintain the connection with remote machine. This field is empty for this context.

When `context_type` is set to `BohriumContext` (or its aliases `bohriumcontext`, `Bohrium`, `bohrium`, `DpCloudServerContext`, `dpcloudservercontext`, `DpCloudServer`, `dpcloudserver`, `LebesgueContext`, `lebesguecontext`, `Lebesgue`, `lebesgue`):

remote_profile:

```

type: dict
argument path: init_reaction_mdata/fp/
machine[BohriumContext]/
remote_profile

```

The information used to maintain the connection with remote machine.

email:

```

type: str, optional
argument path:
init_reaction_mdata/fp/
machine[BohriumContext]/
remote_profile/email

```

Email

password:

```

type: str, optional

```

argument path:
init_reaction_mdata/fp/
machine[BohriumContext]/
remote_profile/password

Password

program_id:

type: int, alias: *project_id*
argument path:
init_reaction_mdata/fp/
machine[BohriumContext]/
remote_profile/program_id

Program ID

retry_count:

type: NoneType | int, optional,
default: 2
argument path:
init_reaction_mdata/fp/
machine[BohriumContext]/
remote_profile/retry_count

The retry count when a job is terminated

ignore_exit_code:

type: bool, optional, default: True
argument path:
init_reaction_mdata/fp/
machine[BohriumContext]/
remote_profile/
ignore_exit_code

The job state will be marked as finished if the exit code is non-zero when set to True. Otherwise,
the job state will be designated as terminated.

keep_backup:

type: bool, optional
argument path:
init_reaction_mdata/fp/
machine[BohriumContext]/
remote_profile/keep_backup

keep download and upload zip

input_data:

type: dict
argument path:
init_reaction_mdata/fp/
machine[BohriumContext]/
remote_profile/input_data

Configuration of job

When `context_type` is set to `OpenAPIContext` (or its aliases `openapicontext`, `OpenAPI`, `openapi`):

remote_profile:

type: dict, optional
 argument path: `init_reaction_mdata/fp/machine[OpenAPIContext]/remote_profile`

The information used to maintain the connection with remote machine. This field is empty for this context.

When `context_type` is set to `LocalContext` (or its aliases `localcontext`, `Local`, `local`):

remote_profile:

type: dict, optional
 argument path: `init_reaction_mdata/fp/machine[LocalContext]/remote_profile`

The information used to maintain the connection with remote machine. This field is empty for this context.

resources:

type: dict
 argument path: `init_reaction_mdata/fp/resources`

number_node:

type: int, optional, default: 1
 argument path: `init_reaction_mdata/fp/resources/number_node`

The number of node need for each *job*

cpu_per_node:

type: int, optional, default: 1
 argument path: `init_reaction_mdata/fp/resources/cpu_per_node`

cpu numbers of each node assigned to each job.

gpu_per_node:

type: int, optional, default: 0
 argument path: `init_reaction_mdata/fp/resources/gpu_per_node`

gpu numbers of each node assigned to each job.

queue_name:

type: str, optional, default: (empty string)

argument path: `init_reaction_mdata/fp/resources/queue_name`

The queue name of batch job scheduler system.

group_size:

type: `int`

argument path: `init_reaction_mdata/fp/resources/group_size`

The number of *tasks* in a *job*. 0 means infinity.

custom_flags:

type: `typing.List[str]`, optional

argument path: `init_reaction_mdata/fp/resources/custom_flags`

The extra lines pass to job submitting script header

strategy:

type: `dict`, optional

argument path: `init_reaction_mdata/fp/resources/strategy`

strategies we use to generation job submitting scripts.

if_cuda_multi_devices:

type: `bool`, optional, default: `False`

argument path:

`init_reaction_mdata/fp/resources/strategy/if_cuda_multi_devices`

If there are multiple nvidia GPUS on the node, and we want to assign the tasks to different GPUS. If true, `dpdispatcher` will manually export environment variable `CUDA_VISIBLE_DEVICES` to different task. Usually, this option will be used with `Task.task_need_resources` variable simultaneously.

ratio_unfinished:

type: `float`, optional, default: `0.0`

argument path:

`init_reaction_mdata/fp/resources/strategy/ratio_unfinished`

The ratio of *tasks* that can be unfinished.

customized_script_header_template_file:

type: str, optional
 argument path:
 init_reaction_mdata/fp/
 resources/strategy/
 customized_script_header_template_file

The customized template file to generate job submitting script header, which overrides the default file.

para_deg:

type: int, optional, default: 1
 argument path: init_reaction_mdata/fp/
 resources/para_deg

Decide how many tasks will be run in parallel.

source_list:

type: typing.List[str], optional, default:
 []
 argument path: init_reaction_mdata/fp/
 resources/source_list

The env file to be sourced before the command execution.

module_purge:

type: bool, optional, default: False
 argument path: init_reaction_mdata/fp/
 resources/module_purge

Remove all modules on HPC system before module load (module_list)

module_unload_list:

type: typing.List[str], optional, default:
 []
 argument path: init_reaction_mdata/fp/
 resources/module_unload_list

The modules to be unloaded on HPC system before submitting jobs

module_list:

type: typing.List[str], optional, default:
 []
 argument path: init_reaction_mdata/fp/
 resources/module_list

The modules to be loaded on HPC system before submitting jobs

envs:

type: dict, optional, default: {}
argument path: `init_reaction_mdata/fp/resources/envs`

The environment variables to be exported on before submitting jobs

prepend_script:

type: `typing.List[str]`, optional, default: []
argument path: `init_reaction_mdata/fp/resources/prepend_script`

Optional script run before jobs submitted.

append_script:

type: `typing.List[str]`, optional, default: []
argument path: `init_reaction_mdata/fp/resources/append_script`

Optional script run after jobs submitted.

wait_time:

type: `float | int`, optional, default: 0
argument path: `init_reaction_mdata/fp/resources/wait_time`

The waiting time in second after a single *task* submitted

Depending on the value of *batch_type*, different sub args are accepted.

batch_type:

type: str (flag key)
argument path: `init_reaction_mdata/fp/resources/batch_type`
possible choices: *Shell*, *PBS*, *DistributedShell*, *SlurmJobArray*, *Slurm*, *OpenAPI*, *Bohrium*, *SGE*, *Torque*, *Fugaku*, *LSF*

The batch job system type loaded from machine/*batch_type*.

When *batch_type* is set to *Shell* (or its alias *shell*):

kwargs:

type: dict, optional
argument path: `init_reaction_mdata/fp/resources[Shell]/kwargs`

This field is empty for this batch.

When `batch_type` is set to PBS (or its alias pbs):

kwargs:

type: dict, optional
 argument path: `init_reaction_mdata/fp/resources[PBS]/kwargs`

This field is empty for this batch.

When `batch_type` is set to DistributedShell (or its alias distributedshell):

kwargs:

type: dict, optional
 argument path: `init_reaction_mdata/fp/resources[DistributedShell]/kwargs`

This field is empty for this batch.

When `batch_type` is set to SlurmJobArray (or its alias slurmjobarray):

kwargs:

type: dict, optional
 argument path: `init_reaction_mdata/fp/resources[SlurmJobArray]/kwargs`

Extra arguments.

custom_gpu_line:

type: str | NoneType, optional,
 default: None

argument path:
`init_reaction_mdata/fp/resources[SlurmJobArray]/kwargs/custom_gpu_line`

Custom GPU configuration, starting with #SBATCH

slurm_job_size:

type: int, optional, default: 1

argument path:
`init_reaction_mdata/fp/resources[SlurmJobArray]/kwargs/slurm_job_size`

Number of tasks in a Slurm job

When `batch_type` is set to Slurm (or its alias slurm):

kwargs:

type: dict, optional
 argument path: `init_reaction_mdata/fp/resources[Slurm]/kwargs`

Extra arguments.

custom_gpu_line:

type: str | NoneType, optional,
default: None

argument path:

init_reaction_mdata/fp/
resources[Slurm]/kwargs/
custom_gpu_line

Custom GPU configuration, starting
with #SBATCH

When `batch_type` is set to OpenAPI (or its alias `openapi`):

kwargs:

type: dict, optional

argument path: init_reaction_mdata/fp/
resources[OpenAPI]/kwargs

This field is empty for this batch.

When `batch_type` is set to Bohrium (or its aliases `bohrium`, `Lebesgue`, `lebesgue`, `DpCloudServer`, `dpcloudserver`):

kwargs:

type: dict, optional

argument path: init_reaction_mdata/fp/
resources[Bohrium]/kwargs

This field is empty for this batch.

When `batch_type` is set to SGE (or its alias `sge`):

kwargs:

type: dict, optional

argument path: init_reaction_mdata/fp/
resources[SGE]/kwargs

This field is empty for this batch.

When `batch_type` is set to Torque (or its alias `torque`):

kwargs:

type: dict, optional

argument path: init_reaction_mdata/fp/
resources[Torque]/kwargs

This field is empty for this batch.

When `batch_type` is set to Fugaku (or its alias `fugaku`):

kwargs:

type: dict, optional

argument path: init_reaction_mdata/fp/
resources[Fugaku]/kwargs

This field is empty for this batch.

When `batch_type` is set to LSF (or its alias `lsf`):

kwargs:

type: dict
 argument path: `init_reaction_mdata/fp/resources[LSF]/kwargs`

Extra arguments.

gpu_usage:

type: bool, optional, default: False
 argument path:
`init_reaction_mdata/fp/resources[LSF]/kwargs/gpu_usage`

Choosing if GPU is used in the calculation step.

gpu_new_syntax:

type: bool, optional, default: False
 argument path:
`init_reaction_mdata/fp/resources[LSF]/kwargs/gpu_new_syntax`

For LFS \geq 10.1.0.3, new option -gpu for #BSUB could be used. If False, and old syntax would be used.

gpu_exclusive:

type: bool, optional, default: True
 argument path:
`init_reaction_mdata/fp/resources[LSF]/kwargs/gpu_exclusive`

Only take effect when new syntax enabled. Control whether submit tasks in exclusive way for GPU.

custom_gpu_line:

type: str | NoneType, optional,
 default: None
 argument path:
`init_reaction_mdata/fp/resources[LSF]/kwargs/custom_gpu_line`

Custom GPU configuration, starting with #BSUB

user_forward_files:

type: list, optional
 argument path:
`init_reaction_mdata/fp/user_forward_files`

Files to be forwarded to the remote machine.

user_backward_files:

type: list, optional

argument path:

init_reaction_mdata/fp/user_backward_files

Files to be backwarded from the remote machine.

SIMPLIFY

6.1 Simplify

When you have a dataset containing lots of repeated data, this step will help you simplify your dataset. The workflow contains three stages: train, model_devi, and fp. The train stage and the fp stage are as the same as the run step, and the model_devi stage will calculate model deviations of the rest data that has not been confirmed accurate. Data with small model deviations will be confirmed accurate, while the program will pick data from those with large model deviations to the new dataset.

Use the following script to start the workflow:

```
dpngen simplify param.json machine.json
```

Here is an example of param.json for QM7 dataset:

```
{
  "type_map": [
    "C",
    "H",
    "N",
    "O",
    "S"
  ],
  "mass_map": [
    12.011,
    1.008,
    14.007,
    15.999,
    32.065
  ],
  "pick_data": "/scratch/jz748/simplify/qm7",
  "init_data_prefix": "",
  "init_data_sys": [],
  "sys_batch_size": [
    "auto"
  ],
  "numb_models": 4,
  "default_training_param": {
    "model": {
      "type_map": [
        "C",
```

(continues on next page)

(continued from previous page)

```

        "H",
        "N",
        "O",
        "S"
    ],
    "descriptor": {
        "type": "se_a",
        "sel": [
            7,
            16,
            3,
            3,
            1
        ],
        "rcut_smth": 1.00,
        "rcut": 6.00,
        "neuron": [
            25,
            50,
            100
        ],
        "resnet_dt": false,
        "axis_neuron": 12
    },
    "fitting_net": {
        "neuron": [
            240,
            240,
            240
        ],
        "resnet_dt": true
    }
},
"learning_rate": {
    "type": "exp",
    "start_lr": 0.001,
    "stop_lr": 5e-8,
    "decay_rate": 0.99
},
"loss": {
    "start_pref_e": 0.02,
    "limit_pref_e": 1,
    "start_pref_f": 1000,
    "limit_pref_f": 1,
    "start_pref_v": 0,
    "limit_pref_v": 0,
    "start_pref_pf": 0,
    "limit_pref_pf": 0
},
"training": {
    "set_prefix": "set",
    "numb_steps": 10000,

```

(continues on next page)

(continued from previous page)

```

        "disp_file": "lcurve.out",
        "disp_freq": 1000,
        "numb_test": 1,
        "save_freq": 1000,
        "disp_training": true,
        "time_training": true,
        "profiling": false,
        "profiling_file": "timeline.json"
    },
    "_comment": "that's all"
},
"fp_style": "gaussian",
"shuffle_poscar": false,
"fp_task_max": 1000,
"fp_task_min": 10,
"fp_pp_path": "/home/jzzeng/",
"fp_pp_files": [],
"fp_params": {
    "keywords": "mn15/6-31g** force nosymm scf(maxcyc=512)",
    "nproc": 28,
    "multiplicity": 1,
    "_comment": " that's all "
},
"init_pick_number": 100,
"iter_pick_number": 100,
"model_devi_f_trust_lo": 0.25,
"model_devi_f_trust_hi": 0.45,
"_comment": " that's all "
}

```

Here *pick_data* is the directory to data to simplify where the program recursively detects systems System with deepmd/npz format. *init_pick_number* and *iter_pick_number* are the numbers of picked frames. *model_devi_f_trust_lo* and *model_devi_f_trust_hi* mean the range of the max deviation of atomic forces in a frame. *fp_style* can be either gaussian or vasp currently. Other parameters are as the same as those of generator.

6.2 dpngen simplify parameters

Note: One can load, modify, and export the input file by using our effective web-based tool [DP-GUI](#) online or hosted using the *command line interface* `dpngen gui`. All parameters below can be set in DP-GUI. By clicking “SAVE JSON”, one can download the input file.

simplify_jdata:

type: dict

argument path: simplify_jdata

Parameters for simplify.json, the first argument of *dpngen simplify*.

type_map:

type: list[str]

argument path: simplify_jdata/type_map

Atom types. Reminder: The elements in param.json, type.raw and data.lmp(when using lammmps) should be in the same order.

mass_map:

type: str | list[float], optional, default: auto

argument path: simplify_jdata/mass_map

Standard atomic weights (default: "auto"). if one want to use isotopes, or non-standard element names, chemical symbols, or atomic number in the type_map list, please customize the mass_map list instead of using "auto".

use_ele_temp:

type: int, optional, default: 0

argument path: simplify_jdata/use_ele_temp

Currently only support fp_style vasp.

- 0: no electron temperature.
- 1: eletron temperature as frame parameter.
- 2: electron temperature as atom parameter.

init_data_prefix:

type: str, optional

argument path: simplify_jdata/init_data_prefix

Prefix of initial data directories.

init_data_sys:

type: list[str]

argument path: simplify_jdata/init_data_sys

Paths of initial data. The path can be either a system diretory containing NumPy files or an HDF5 file. You may use either absolute or relative path here. Systems will be detected recursively in the directories or the HDF5 file.

sys_format:

type: str, optional, default: vasp/poscar

argument path: simplify_jdata/sys_format

Format of sys_configs.

init_batch_size:

type: str | list[typing.Union[int, str]], optional

argument path: simplify_jdata/init_batch_size

Each number is the batch_size of corresponding system for training in init_data_sys. One recommended rule for setting the sys_batch_size and init_batch_size is that batch_size mutiply number of atoms ot the stucture should be larger than 32. If set to auto, batch size will be 32 divided by number of atoms. This argument will not override the mixed batch size in *default_training_param*.

sys_configs_prefix:

type: str, optional

argument path: simplify_jdata/sys_configs_prefix

Prefix of sys_configs.

sys_configs:

type: list[list[str]]

argument path: `simplify_jdata/sys_configs`

2D list. Containing directories of structures to be explored in iterations for each system. Wildcard characters are supported here.

sys_batch_size:

type: `list[typing.Union[int, str]]`, optional

argument path: `simplify_jdata/sys_batch_size`

Each number is the `batch_size` for training of corresponding system in `sys_configs`. If set to `auto`, batch size will be 32 divided by number of atoms. This argument will not override the mixed batch size in `default_training_param`.

labeled:

type: `bool`, optional, default: `False`

argument path: `simplify_jdata/labeled`

If true, the initial data is labeled.

pick_data:

type: `str | list[str]`

argument path: `simplify_jdata/pick_data`

(List of) Path to the directory with the pick data with the `deepmd/npz` or the `HDF5` file with `deepmd/hdf5` format. Systems are detected recursively.

init_pick_number:

type: `int`

argument path: `simplify_jdata/init_pick_number`

The number of initial pick data.

iter_pick_number:

type: `int`

argument path: `simplify_jdata/iter_pick_number`

The number of pick data in each iteration.

model_devi_f_trust_lo:

type: `float`

argument path: `simplify_jdata/model_devi_f_trust_lo`

The lower bound of forces for the selection for the model deviation.

model_devi_f_trust_hi:

type: `float`

argument path: `simplify_jdata/model_devi_f_trust_hi`

The higher bound of forces for the selection for the model deviation.

model_devi_e_trust_lo:

type: `float`, optional, default: `100000000000.0`

argument path: `simplify_jdata/model_devi_e_trust_lo`

The lower bound of energy per atom for the selection for the model deviation. Requires DeePMD-kit version $\geq 2.2.2$.

model_devi_e_trust_hi:

type: `float`, optional, default: `100000000000.0`

argument path: `simplify_jdata/model_devi_e_trust_hi`

The higher bound of energy per atom for the selection for the model deviation. Requires DeePMD-kit version $\geq 2.2.2$.

true_error_f_trust_lo:

type: float, optional, default: `10000000000.0`

argument path: `simplify_jdata/true_error_f_trust_lo`

The lower bound of forces for the selection for the true error. Requires DeePMD-kit version $\geq 2.2.4$.

true_error_f_trust_hi:

type: float, optional, default: `10000000000.0`

argument path: `simplify_jdata/true_error_f_trust_hi`

The higher bound of forces for the selection for the true error. Requires DeePMD-kit version $\geq 2.2.4$.

true_error_e_trust_lo:

type: float, optional, default: `10000000000.0`

argument path: `simplify_jdata/true_error_e_trust_lo`

The lower bound of energy per atom for the selection for the true error. Requires DeePMD-kit version $\geq 2.2.4$.

true_error_e_trust_hi:

type: float, optional, default: `10000000000.0`

argument path: `simplify_jdata/true_error_e_trust_hi`

The higher bound of energy per atom for the selection for the true error. Requires DeePMD-kit version $\geq 2.2.4$.

numb_models:

type: int

argument path: `simplify_jdata/numb_models`

Number of models to be trained in `00.train`. 4 is recommend.

training_iter0_model_path:

type: list[str], optional

argument path: `simplify_jdata/training_iter0_model_path`

The model used to init the first iter training. Number of element should be equal to `numb_models`.

training_init_model:

type: bool, optional

argument path: `simplify_jdata/training_init_model`

Iteration > 0 , the model parameters will be initilized from the model trained at the previous iteration. Iteration $== 0$, the model parameters will be initialized from `training_iter0_model_path`.

default_training_param:

type: dict

argument path: `simplify_jdata/default_training_param`

Training parameters for deepmd-kit in `00.train`. You can find instructions from [DeePMD-kit documentation](#).

dp_train_skip_neighbor_stat:

type: bool, optional, default: False

argument path: simplify_jdata/dp_train_skip_neighbor_stat

Append `--skip-neighbor-stat` flag to dp train.

dp_compress:

type: bool, optional, default: False

argument path: simplify_jdata/dp_compress

Use dp compress to compress the model.

training_reuse_iter:

type: int | NoneType, optional

argument path: simplify_jdata/training_reuse_iter

The minimal index of iteration that continues training models from old models of last iteration.

training_reuse_old_ratio:

type: str | float, optional, default: auto

argument path: simplify_jdata/training_reuse_old_ratio

The probability proportion of old data during training. It can be:

- float: directly assign the probability of old data;
- *auto:f*: automatic probability, where f is the new-to-old ratio;
- *auto*: equivalent to *auto:10*.

This option is only adopted when continuing training models from old models. This option will override default parameters.

training_reuse_num_steps:

type: int | NoneType, optional, default: None, alias:

training_reuse_stop_batch

argument path: simplify_jdata/training_reuse_num_steps

Number of training batch. This option is only adopted when continuing training models from old models. This option will override default parameters.

training_reuse_start_lr:

type: float | NoneType, optional, default: None

argument path: simplify_jdata/training_reuse_start_lr

The learning rate the start of the training. This option is only adopted when continuing training models from old models. This option will override default parameters.

training_reuse_start_pref_e:

type: int | float | NoneType, optional, default: None

argument path: simplify_jdata/training_reuse_start_pref_e

The prefactor of energy loss at the start of the training. This option is only adopted when continuing training models from old models. This option will override default parameters.

training_reuse_start_pref_f:

type: int | float | NoneType, optional, default: None

argument path: `simplify_jdata/training_reuse_start_pref_f`

The prefactor of force loss at the start of the training. This option is only adopted when continuing training models from old models. This option will override default parameters.

model_devi_activation_func:

type: `NoneType | list[list[str]]`, optional

argument path: `simplify_jdata/model_devi_activation_func`

The activation function in the model. The shape of list should be `(N_models, 2)`, where 2 represents the embedding and fitting network. This option will override default parameters.

srtab_file_path:

type: `str`, optional

argument path: `simplify_jdata/srtab_file_path`

The path of the table for the short-range pairwise interaction which is needed when using DP-ZBL potential

one_h5:

type: `bool`, optional, default: `False`

argument path: `simplify_jdata/one_h5`

When using DeePMD-kit, all of the input data will be merged into one HDF5 file.

training_init_frozen_model:

type: `list[str]`, optional

argument path: `simplify_jdata/training_init_frozen_model`

At iteration 0, initialize the model parameters from the given frozen models. Number of element should be equal to `numb_models`.

training_finetune_model:

type: `list[str]`, optional

argument path: `simplify_jdata/training_finetune_model`

At iteration 0, finetune the model parameters from the given frozen models. Number of element should be equal to `numb_models`.

fp_task_max:

type: `int`, optional

argument path: `simplify_jdata/fp_task_max`

Maximum of structures to be calculated in 02.fp of each iteration.

fp_task_min:

type: `int`, optional

argument path: `simplify_jdata/fp_task_min`

Minimum of structures to be calculated in 02.fp of each iteration.

fp_accurate_threshold:

type: `float`, optional

argument path: `simplify_jdata/fp_accurate_threshold`

If the accurate ratio is larger than this number, no fp calculation will be performed, i.e. `fp_task_max = 0`.

fp_accurate_soft_threshold:

type: float, optional

argument path: `simplify_jdata/fp_accurate_soft_threshold`

If the accurate ratio is between this number and `fp_accurate_threshold`, the `fp_task_max` linearly decays to zero.

ratio_failed:

type: float, optional

argument path: `simplify_jdata/ratio_failed`

Check the ratio of unsuccessfully terminated jobs. If too many FP tasks are not converged, `RuntimeError` will be raised.

Depending on the value of `fp_style`, different sub args are accepted.

fp_style:

type: str (flag key), default: none

argument path: `simplify_jdata/fp_style`

possible choices: *none*, *vasp*, *gaussian*, *siesta*, *cp2k*, *abacus*, *pwmat*, *pwscf*, *custom*

Software for First Principles, if *labeled* is false.

When `fp_style` is set to *none*:

No fp.

When `fp_style` is set to *vasp*:

VASP.

fp_pp_path:

type: str

argument path: `simplify_jdata[vasp]/fp_pp_path`

Directory of psuedo-potential file to be used for 02.fp exists.

fp_pp_files:

type: list[str]

argument path: `simplify_jdata[vasp]/fp_pp_files`

Pseudo-potential file to be used for 02.fp. Note that the order of elements should correspond to the order in `type_map`.

fp_incar:

type: str

argument path: `simplify_jdata[vasp]/fp_incar`

Input file for VASP. INCAR must specify KSPACING and KGAMMA.

fp_aniso_kspacing:

type: list[float], optional

argument path: `simplify_jdata[vasp]/fp_aniso_kspacing`

Set anisotropic kspacing. Usually useful for 1-D or 2-D materials. Only support VASP. If it is setting the KSPACING key in INCAR will be ignored.

cvasp:

type: bool, optional

argument path: `simplify_jdata[vasp]/cvasp`

If `cvasp` is true, DP-GEN will use Custodian to help control VASP calculation.

fp_skip_bad_box:

type: str, optional

argument path: `simplify_jdata[vasp]/fp_skip_bad_box`

Skip the configurations that are obviously unreasonable before 02.fp

When `fp_style` is set to gaussian:

Gaussian. The command should be set as *g16 < input*.

use_clusters:

type: bool, optional, default: False

argument path: `simplify_jdata[gaussian]/use_clusters`

If set to true, clusters will be taken instead of the whole system.

cluster_cutoff:

type: float, optional

argument path: `simplify_jdata[gaussian]/cluster_cutoff`

The soft cutoff radius of clusters if *use_clusters* is set to true. Molecules will be taken as whole even if part of atoms is out of the cluster. Use *cluster_cutoff_hard* to only take atoms within the hard cutoff radius.

cluster_cutoff_hard:

type: float, optional

argument path: `simplify_jdata[gaussian]/cluster_cutoff_hard`

The hard cutoff radius of clusters if *use_clusters* is set to true. Outside the hard cutoff radius, atoms will not be taken even if they are in a molecule where some atoms are within the cutoff radius.

cluster_minify:

type: bool, optional, default: False

argument path: `simplify_jdata[gaussian]/cluster_minify`

If enabled, when an atom within the soft cutoff radius connects a single bond with a non-hydrogen atom out of the soft cutoff radius, the outer atom will be replaced by a hydrogen atom. When the outer atom is a hydrogen atom, the outer atom will be kept. In this case, other atoms out of the soft cutoff radius will be removed.

fp_params:

type: dict

argument path: `simplify_jdata[gaussian]/fp_params`

Parameters for Gaussian calculation.

keywords:

type: str | list[str]

argument path:

`simplify_jdata[gaussian]/fp_params/keywords`

Keywords for Gaussian input, e.g. force b3lyp/6-31g**. If a list, run multiple steps.

multiplicity:

type: str | int, optional, default: auto
 argument path: simplify_jdata[gaussian]/
 fp_params/multiplicity

Spin multiplicity for Gaussian input. If *auto*, multiplicity will be detected automatically, with the following rules: when fragment_guesses=True, multiplicity will +1 for each radical, and +2 for each oxygen molecule; when fragment_guesses=False, multiplicity will be 1 or 2, but +2 for each oxygen molecule.

nproc:

type: int
 argument path:
 simplify_jdata[gaussian]/fp_params/nproc

The number of processors for Gaussian input.

charge:

type: int, optional, default: 0
 argument path:
 simplify_jdata[gaussian]/fp_params/charge

Molecule charge. Only used when charge is not provided by the system.

fragment_guesses:

type: bool, optional, default: False
 argument path: simplify_jdata[gaussian]/
 fp_params/fragment_guesses

Initial guess generated from fragment guesses. If True, *multiplicity* should be *auto*.

basis_set:

type: str, optional
 argument path:
 simplify_jdata[gaussian]/fp_params/basis_set

Custom basis set.

keywords_high_multiplicity:

type: str, optional
 argument path: simplify_jdata[gaussian]/
 fp_params/keywords_high_multiplicity

Keywords for points with multiple radicals. *multiplicity* should be *auto*. If not set, fallback to normal keywords.

When `fp_style` is set to `siesta`:

use_clusters:

type: bool, optional

argument path: `simplify_jdata[siesta]/use_clusters`

If set to true, clusters will be taken instead of the whole system. This option does not work with DeePMD-kit 0.x.

cluster_cutoff:

type: float, optional

argument path: `simplify_jdata[siesta]/cluster_cutoff`

The cutoff radius of clusters if `use_clusters` is set to true.

fp_params:

type: dict

argument path: `simplify_jdata[siesta]/fp_params`

Parameters for siesta calculation.

ecut:

type: int

argument path:

`simplify_jdata[siesta]/fp_params/ecut`

Define the plane wave cutoff for grid.

ediff:

type: float

argument path:

`simplify_jdata[siesta]/fp_params/ediff`

Tolerance of Density Matrix.

kspacing:

type: float

argument path:

`simplify_jdata[siesta]/fp_params/kspacing`

Sample factor in Brillouin zones.

mixingWeight:

type: float

argument path:

`simplify_jdata[siesta]/fp_params/mixingWeight`

Proportion a of output Density Matrix to be used for the input Density Matrix of next SCF cycle (linear mixing).

NumberPulay:

type: int

argument path:

`simplify_jdata[siesta]/fp_params/NumberPulay`

Controls the Pulay convergence accelerator.

fp_pp_path:

type: str

argument path: `simplify_jdata[siesta]/fp_pp_path`

Directory of psuedo-potential or numerical orbital files to be used for 02.fp exists.

fp_pp_files:

type: list[str]

argument path: `simplify_jdata[siesta]/fp_pp_files`Psuedo-potential file to be used for 02.fp. Note that the order of elements should correspond to the order in `type_map`.When `fp_style` is set to `cp2k`:**user_fp_params:**type: dict, optional, alias: `fp_params`argument path: `simplify_jdata[cp2k]/user_fp_params`Parameters for cp2k calculation. find detail in manual.cp2k.org. only the kind section must be set before use. we assume that you have basic knowledge for cp2k input.**external_input_path:**

type: str, optional

argument path: `simplify_jdata[cp2k]/external_input_path`Conflict with key: `user_fp_params`. enable the template input provided by user. some rules should be followed, read the following text in detail:

1. One must present a KEYWORD ABC in the section CELL so that the DP-GEN can replace the cell on-the-fly.
2. One need to add these lines under FORCE_EVAL section to print forces and stresses:

```
STRESS_TENSOR ANALYTICAL
&PRINT
  &FORCES ON
  &END FORCES
  &STRESS_TENSOR ON
  &END STRESS_TENSOR
&END PRINT
```

When `fp_style` is set to `abacus`:**fp_pp_path:**

type: str

argument path: `simplify_jdata[abacus]/fp_pp_path`

Directory of psuedo-potential or numerical orbital files to be used for 02.fp exists.

fp_pp_files:

type: list[str]

argument path: `simplify_jdata[abacus]/fp_pp_files`Psuedo-potential file to be used for 02.fp. Note that the order of elements should correspond to the order in `type_map`.

fp_orb_files:

type: list[str], optional

argument path: simplify_jdata[abacus]/fp_orb_files

numerical orbital file to be used for 02.fp when using LCAO basis. Note that the order of elements should correspond to the order in type_map.

fp_incar:

type: str, optional

argument path: simplify_jdata[abacus]/fp_incar

Input file for ABACUS. This is optional but the priority is lower than user_fp_params, and you should not set user_fp_params if you want to use fp_incar.

fp_kpt_file:

type: str, optional

argument path: simplify_jdata[abacus]/fp_kpt_file

KPT file for ABACUS. If the “kspacing” or “gamma_only=1” is defined in INPUT or “k_points” is defined, fp_kpt_file will be ignored.

fp_dpks_descriptor:

type: str, optional

argument path: simplify_jdata[abacus]/fp_dpks_descriptor

DeePKS descriptor file name. The file should be in pseudopotential directory.

user_fp_params:

type: dict, optional

argument path: simplify_jdata[abacus]/user_fp_params

Set the key and value of INPUT.

k_points:

type: list[int], optional

argument path: simplify_jdata[abacus]/k_points

Monkhorst-Pack k-grids setting for generating KPT file of ABACUS, such as: [1,1,1,0,0,0]. NB: if “kspacing” or “gamma_only=1” is defined in INPUT, k_points will be ignored.

When `fp_style` is set to `pwmat`:

TODO: add doc

When `fp_style` is set to `pwscf`:

`pwscf` (Quantum Espresso).

fp_pp_path:

type: str

argument path: simplify_jdata[pwscf]/fp_pp_path

Directory of psuedo-potential file to be used for 02.fp exists.

fp_pp_files:

type: list[str]

argument path: `simplify_jdata[pwscf]/fp_pp_files`

Pseudo-potential file to be used for 02.fp. Note that the order of elements should correspond to the order in `type_map`.

fp_params:

type: dict, optional

argument path: `simplify_jdata[pwscf]/fp_params`

Parameters for pwscf calculation. It has lower priority than `user_fp_params`.

ecut:

type: float

argument path:

`simplify_jdata[pwscf]/fp_params/ecut`

`ecutwfc` in pwscf.

ediff:

type: float

argument path:

`simplify_jdata[pwscf]/fp_params/ediff`

`conv_thr` and `ts_vdw_econv_thr` in pwscf.

smearing:

type: str

argument path:

`simplify_jdata[pwscf]/fp_params/smearing`

`smearing` in pwscf.

sigma:

type: float

argument path:

`simplify_jdata[pwscf]/fp_params/sigma`

`degauss` in pwscf.

kspacing:

type: float

argument path:

`simplify_jdata[pwscf]/fp_params/kspacing`

The spacing between kpoints. Helps to determin KPOINTS in pwscf.

user_fp_params:

type: dict, optional

argument path: `simplify_jdata[pwscf]/user_fp_params`

Parameters for pwscf calculation. Find details at https://www.quantum-espresso.org/Doc/INPUT_PW.html. When `user_fp_params` is set, the settings in `fp_params` will be ignored. If one wants to use `user_fp_params`, `kspacing` must be set in `user_fp_params`. `kspacing` is the spacing between kpoints, and helps to determin KPOINTS in pwscf.

When `fp_style` is set to `custom`:

Custom FP code. You need to provide the input and output file format and name. The command argument in the machine file should be the script to run custom FP codes. The extra forward and backward files can be defined in the machine file.

fp_params:

type: dict
argument path: `simplify_jdata[custom]/fp_params`

Parameters for FP calculation.

input_fmt:

type: str
argument path:
`simplify_jdata[custom]/fp_params/input_fmt`

Input dpdata format of the custom FP code. Such format should only need the first argument as the file name.

input_fn:

type: str
argument path:
`simplify_jdata[custom]/fp_params/input_fn`

Input file name of the custom FP code.

output_fmt:

type: str
argument path:
`simplify_jdata[custom]/fp_params/output_fmt`

Output dpdata format of the custom FP code. Such format should only need the first argument as the file name.

output_fn:

type: str
argument path:
`simplify_jdata[custom]/fp_params/output_fn`

Output file name of the custom FP code.

6.3 dpgen simplify machine parameters

Note: One can load, modify, and export the input file by using our effective web-based tool [DP-GUI](#) online or hosted using the *command line interface* `dpgen gui`. All parameters below can be set in DP-GUI. By clicking “SAVE JSON”, one can download the input file.

simplify_mdata:

type: dict
argument path: `simplify_mdata`
machine.json file

api_version:

type: str, optional, default: 1.0
 argument path: simplify_mdata/api_version
 Please set to 1.0

deepmd_version:

type: str, optional, default: 2
 argument path: simplify_mdata/deepmd_version
 DeePMD-kit version, e.g. 2.1.3

train:

type: dict
 argument path: simplify_mdata/train
 Parameters of command, machine, and resources for train

command:

type: str
 argument path: simplify_mdata/train/command
 Command of a program.

machine:

type: dict
 argument path: simplify_mdata/train/machine

batch_type:

type: str
 argument path: simplify_mdata/train/machine/batch_type
 The batch job system type. Option: Bohrium, PBS, OpenAPI, DistributedShell, LSF, SlurmJobArray, Slurm, SGE, Fugaku, Torque, Shell

local_root:

type: str | NoneType
 argument path: simplify_mdata/train/machine/local_root
 The dir where the tasks and relating files locate. Typically the project dir.

remote_root:

type: str | NoneType, optional
 argument path: simplify_mdata/train/machine/remote_root
 The dir where the tasks are executed on the remote machine. Only needed when context is not lazy-local.

clean_asynchronously:

type: bool, optional, default: False
argument path: `simplify_mdata/train/
machine/clean_asynchronously`

Clean the remote directory asynchronously after the job finishes.

Depending on the value of `context_type`, different sub args are accepted.

context_type:

type: str (flag key)
argument path: `simplify_mdata/train/
machine/context_type`
possible choices: `SSHContext`,
`LazyLocalContext`, `HDFSContext`,
`BohriumContext`, `OpenAPIContext`,
`LocalContext`

The connection used to remote machine. Option: `LocalContext`, `SSHContext`, `OpenAPIContext`, `BohriumContext`, `LazyLocalContext`, `HDFSContext`

When `context_type` is set to `SSHContext` (or its aliases `sshcontext`, `SSH`, `ssh`):

remote_profile:

type: dict
argument path: `simplify_mdata/train/
machine[SSHContext]/remote_profile`

The information used to maintain the connection with remote machine.

hostname:

type: str
argument path: `simplify_mdata/
train/machine[SSHContext]/
remote_profile/hostname`
hostname or ip of ssh connection.

username:

type: str
argument path: `simplify_mdata/
train/machine[SSHContext]/
remote_profile/username`
username of target linux system

password:

type: str, optional

argument path: `simplify_mdata/
train/machine[SSHContext]/
remote_profile/password`

(deprecated) password of linux system. Please use [SSH keys](#) instead to improve security.

port:

type: `int`, optional, default: 22
argument path: `simplify_mdata/
train/machine[SSHContext]/
remote_profile/port`

ssh connection port.

key_filename:

type: `str | NoneType`, optional,
default: `None`
argument path: `simplify_mdata/
train/machine[SSHContext]/
remote_profile/key_filename`

key filename used by ssh connection.
If left `None`, find key in `~/.ssh` or use
password for login

passphrase:

type: `str | NoneType`, optional,
default: `None`
argument path: `simplify_mdata/
train/machine[SSHContext]/
remote_profile/passphrase`

passphrase of key used by ssh connection

timeout:

type: `int`, optional, default: 10
argument path: `simplify_mdata/
train/machine[SSHContext]/
remote_profile/timeout`

timeout of ssh connection

totp_secret:

type: `str | NoneType`, optional,
default: `None`
argument path: `simplify_mdata/
train/machine[SSHContext]/
remote_profile/totp_secret`

Time-based one time password secret. It should be a base32-encoded string extracted from the 2D code.

tar_compress:

type: `bool`, optional, default: `True`

argument path: `simplify_mdata/
train/machine[SSHContext]/
remote_profile/tar_compress`

The archive will be compressed in upload and download if it is True. If not, compression will be skipped.

look_for_keys:

type: bool, optional, default: True
argument path: `simplify_mdata/
train/machine[SSHContext]/
remote_profile/look_for_keys`

enable searching for discoverable private key files in `~/.ssh/`

When `context_type` is set to `LazyLocalContext` (or its aliases `lazylocalcontext`, `LazyLocal`, `lazylocal`):

remote_profile:

type: dict, optional
argument path: `simplify_mdata/train/
machine[LazyLocalContext]/
remote_profile`

The information used to maintain the connection with remote machine. This field is empty for this context.

When `context_type` is set to `HDFSContext` (or its aliases `hdfscontext`, `HDFS`, `hdfs`):

remote_profile:

type: dict, optional
argument path: `simplify_mdata/train/
machine[HDFSContext]/remote_profile`

The information used to maintain the connection with remote machine. This field is empty for this context.

When `context_type` is set to `BohriumContext` (or its aliases `bohriumcontext`, `Bohrium`, `bohrium`, `DpCloudServerContext`, `dpcloudservercontext`, `DpCloudServer`, `dpcloudserver`, `LebesgueContext`, `lebesguecontext`, `Lebesgue`, `lebesgue`):

remote_profile:

type: dict
argument path: `simplify_mdata/train/
machine[BohriumContext]/
remote_profile`

The information used to maintain the connection with remote machine.

email:

type: str, optional
 argument path:
 simplify_mdata/train/
 machine[BohriumContext]/
 remote_profile/email

Email

password:

type: str, optional
 argument path:
 simplify_mdata/train/
 machine[BohriumContext]/
 remote_profile/password

Password

program_id:

type: int, alias: *project_id*
 argument path:
 simplify_mdata/train/
 machine[BohriumContext]/
 remote_profile/program_id

Program ID

retry_count:

type: NoneType | int, optional,
 default: 2
 argument path:
 simplify_mdata/train/
 machine[BohriumContext]/
 remote_profile/retry_count

The retry count when a job is terminated

ignore_exit_code:

type: bool, optional, default: True
 argument path:
 simplify_mdata/train/
 machine[BohriumContext]/
 remote_profile/
 ignore_exit_code

The job state will be marked as finished if the exit code is non-zero when set to True. Otherwise, the job state will be designated as terminated.

keep_backup:

type: bool, optional
 argument path:
 simplify_mdata/train/

```
machine[BohriumContext]/
remote_profile/keep_backup

keep download and upload zip
```

input_data:

```
type: dict
argument path:
simplify_mdata/train/
machine[BohriumContext]/
remote_profile/input_data
```

Configuration of job

When `context_type` is set to `OpenAPIContext` (or its aliases `openapicontext`, `OpenAPI`, `openapi`):

remote_profile:

```
type: dict, optional
argument path: simplify_mdata/train/
machine[OpenAPIContext]/
remote_profile
```

The information used to maintain the connection with remote machine. This field is empty for this context.

When `context_type` is set to `LocalContext` (or its aliases `localcontext`, `Local`, `local`):

remote_profile:

```
type: dict, optional
argument path: simplify_mdata/train/
machine[LocalContext]/
remote_profile
```

The information used to maintain the connection with remote machine. This field is empty for this context.

resources:

```
type: dict
argument path: simplify_mdata/train/resources
```

number_node:

```
type: int, optional, default: 1
argument path: simplify_mdata/train/
resources/number_node
```

The number of node need for each *job*

cpu_per_node:

```
type: int, optional, default: 1
argument path: simplify_mdata/train/
resources/cpu_per_node
```

cpu numbers of each node assigned to each job.

gpu_per_node:

type: int, optional, default: 0

argument path: `simplify_mdata/train/resources/gpu_per_node`

gpu numbers of each node assigned to each job.

queue_name:

type: str, optional, default: (empty string)

argument path: `simplify_mdata/train/resources/queue_name`

The queue name of batch job scheduler system.

group_size:

type: int

argument path: `simplify_mdata/train/resources/group_size`

The number of *tasks* in a *job*. 0 means infinity.

custom_flags:

type: `typing.List[str]`, optional

argument path: `simplify_mdata/train/resources/custom_flags`

The extra lines pass to job submitting script header

strategy:

type: dict, optional

argument path: `simplify_mdata/train/resources/strategy`

strategies we use to generation job submitting scripts.

if_cuda_multi_devices:

type: bool, optional, default: False

argument path: `simplify_mdata/train/resources/strategy/if_cuda_multi_devices`

If there are multiple nvidia GPUS on the node, and we want to assign the tasks to different GPUS. If true, `dpdispatcher` will manually export environment variable `CUDA_VISIBLE_DEVICES` to different task. Usually, this option will be used with `Task.task_need_resources` variable simultaneously.

ratio_unfinished:

type: float, optional, default: 0.0
argument path: simplify_mdata/
train/resources/strategy/
ratio_unfinished

The ratio of *tasks* that can be unfinished.

customized_script_header_template_file:

type: str, optional
argument path: simplify_mdata/
train/resources/strategy/
customized_script_header_template_file

The customized template file to generate job submitting script header, which overrides the default file.

para_deg:

type: int, optional, default: 1
argument path: simplify_mdata/train/
resources/para_deg

Decide how many tasks will be run in parallel.

source_list:

type: typing.List[str], optional, default:
[]
argument path: simplify_mdata/train/
resources/source_list

The env file to be sourced before the command execution.

module_purge:

type: bool, optional, default: False
argument path: simplify_mdata/train/
resources/module_purge

Remove all modules on HPC system before module load (module_list)

module_unload_list:

type: typing.List[str], optional, default:
[]
argument path: simplify_mdata/train/
resources/module_unload_list

The modules to be unloaded on HPC system before submitting jobs

module_list:

type: typing.List[str], optional, default:
[]

argument path: `simplify_mdata/train/resources/module_list`

The modules to be loaded on HPC system before submitting jobs

envs:

type: dict, optional, default: {}

argument path: `simplify_mdata/train/resources/envs`

The environment variables to be exported on before submitting jobs

prepend_script:

type: `typing.List[str]`, optional, default: []

argument path: `simplify_mdata/train/resources/prepend_script`

Optional script run before jobs submitted.

append_script:

type: `typing.List[str]`, optional, default: []

argument path: `simplify_mdata/train/resources/append_script`

Optional script run after jobs submitted.

wait_time:

type: `float | int`, optional, default: 0

argument path: `simplify_mdata/train/resources/wait_time`

The waiting time in second after a single *task* submitted

Depending on the value of *batch_type*, different sub args are accepted.

batch_type:

type: str (flag key)

argument path: `simplify_mdata/train/resources/batch_type`

possible choices: *Shell*, *PBS*, *DistributedShell*, *SlurmJobArray*, *Slurm*, *OpenAPI*, *Bohrium*, *SGE*, *Torque*, *Fugaku*, *LSF*

The batch job system type loaded from machine/*batch_type*.

When *batch_type* is set to *Shell* (or its alias *shell*):

kwargs:

type: dict, optional
argument path: `simplify_mdata/train/
resources[Shell]/kwargs`

This field is empty for this batch.

When `batch_type` is set to PBS (or its alias `pbs`):

kwargs:

type: dict, optional
argument path: `simplify_mdata/train/
resources[PBS]/kwargs`

This field is empty for this batch.

When `batch_type` is set to DistributedShell (or its alias `distributedshell`):

kwargs:

type: dict, optional
argument path: `simplify_mdata/train/
resources[DistributedShell]/kwargs`

This field is empty for this batch.

When `batch_type` is set to SlurmJobArray (or its alias `slurmjobarray`):

kwargs:

type: dict, optional
argument path: `simplify_mdata/train/
resources[SlurmJobArray]/kwargs`

Extra arguments.

custom_gpu_line:

type: str | NoneType, optional,
default: None

argument path:
`simplify_mdata/train/
resources[SlurmJobArray]/
kwargs/custom_gpu_line`

Custom GPU configuration, starting
with #SBATCH

slurm_job_size:

type: int, optional, default: 1
argument path:
`simplify_mdata/train/
resources[SlurmJobArray]/
kwargs/slurm_job_size`

Number of tasks in a Slurm job

When `batch_type` is set to Slurm (or its alias `slurm`):

kwargs:

type: dict, optional
 argument path: `simplify_mdata/train/resources[Slurm]/kwargs`

Extra arguments.

custom_gpu_line:

type: str | NoneType, optional,
 default: None
 argument path: `simplify_mdata/train/resources[Slurm]/kwargs/custom_gpu_line`
 Custom GPU configuration, starting with #SBATCH

When `batch_type` is set to OpenAPI (or its alias `openapi`):

kwargs:

type: dict, optional
 argument path: `simplify_mdata/train/resources[OpenAPI]/kwargs`

This field is empty for this batch.

When `batch_type` is set to Bohrium (or its aliases `bohrium`, `Lebesgue`, `lebesgue`, `DpCloudServer`, `dpcloudserver`):

kwargs:

type: dict, optional
 argument path: `simplify_mdata/train/resources[Bohrium]/kwargs`

This field is empty for this batch.

When `batch_type` is set to SGE (or its alias `sge`):

kwargs:

type: dict, optional
 argument path: `simplify_mdata/train/resources[SGE]/kwargs`

This field is empty for this batch.

When `batch_type` is set to Torque (or its alias `torque`):

kwargs:

type: dict, optional
 argument path: `simplify_mdata/train/resources[Torque]/kwargs`

This field is empty for this batch.

When `batch_type` is set to Fugaku (or its alias `fugaku`):

kwargs:

type: dict, optional
argument path: `simplify_mdata/train/resources[Fugaku]/kwargs`

This field is empty for this batch.

When `batch_type` is set to LSF (or its alias `lsf`):

kwargs:

type: dict
argument path: `simplify_mdata/train/resources[LSF]/kwargs`

Extra arguments.

gpu_usage:

type: bool, optional, default: False
argument path: `simplify_mdata/train/resources[LSF]/kwargs/gpu_usage`

Choosing if GPU is used in the calculation step.

gpu_new_syntax:

type: bool, optional, default: False
argument path: `simplify_mdata/train/resources[LSF]/kwargs/gpu_new_syntax`

For LFS \geq 10.1.0.3, new option `-gpu` for `#BSUB` could be used. If False, and old syntax would be used.

gpu_exclusive:

type: bool, optional, default: True
argument path: `simplify_mdata/train/resources[LSF]/kwargs/gpu_exclusive`

Only take effect when new syntax enabled. Control whether submit tasks in exclusive way for GPU.

custom_gpu_line:

type: str | NoneType, optional,
default: None
argument path: `simplify_mdata/train/resources[LSF]/kwargs/custom_gpu_line`

Custom GPU configuration, starting with `#BSUB`

user_forward_files:

type: list, optional
 argument path: `simplify_mdata/train/user_forward_files`
 Files to be forwarded to the remote machine.

user_backward_files:

type: list, optional
 argument path: `simplify_mdata/train/user_backward_files`
 Files to be backwarded from the remote machine.

model_devi:

type: dict
 argument path: `simplify_mdata/model_devi`
 Parameters of command, machine, and resources for model_devi

command:

type: str
 argument path: `simplify_mdata/model_devi/command`
 Command of a program.

machine:

type: dict
 argument path: `simplify_mdata/model_devi/machine`

batch_type:

type: str
 argument path: `simplify_mdata/model_devi/machine/batch_type`
 The batch job system type. Option: Bohrium, PBS, OpenAPI, DistributedShell, LSF, SlurmJobArray, Slurm, SGE, Fugaku, Torque, Shell

local_root:

type: str | NoneType
 argument path: `simplify_mdata/model_devi/machine/local_root`
 The dir where the tasks and relating files locate. Typically the project dir.

remote_root:

type: str | NoneType, optional
 argument path: `simplify_mdata/model_devi/machine/remote_root`
 The dir where the tasks are executed on the remote machine. Only needed when context is not lazy-local.

clean_asynchronously:

type: bool, optional, default: False

argument path:

simplify_mdata/model_devi/machine/
clean_asynchronously

Clean the remote directory asynchronously after the job finishes.

Depending on the value of *context_type*, different sub args are accepted.

context_type:

type: str (flag key)

argument path: simplify_mdata/
model_devi/machine/context_type

possible choices: *SSHContext*,
LazyLocalContext, *HDFSContext*,
BohriumContext, *OpenAPIContext*,
LocalContext

The connection used to remote machine. Option: LocalContext, SSHContext, OpenAPIContext, BohriumContext, LazyLocalContext, HDFSContext

When *context_type* is set to SSHContext (or its aliases *sshcontext*, *SSH*, *ssh*):

remote_profile:

type: dict

argument path:

simplify_mdata/model_devi/
machine[SSHContext]/remote_profile

The information used to maintain the connection with remote machine.

hostname:

type: str

argument path:

simplify_mdata/model_devi/
machine[SSHContext]/
remote_profile/hostname

hostname or ip of ssh connection.

username:

type: str

argument path:

simplify_mdata/model_devi/
machine[SSHContext]/
remote_profile/username

username of target linux system

password:

type: str, optional
 argument path:
 simplify_mdata/model_devi/
 machine[SSHContext]/
 remote_profile/password
 (deprecated) password of linux system. Please use [SSH keys](#) instead to improve security.

port:

type: int, optional, default: 22
 argument path:
 simplify_mdata/model_devi/
 machine[SSHContext]/
 remote_profile/port
 ssh connection port.

key_filename:

type: str | NoneType, optional,
 default: None
 argument path:
 simplify_mdata/model_devi/
 machine[SSHContext]/
 remote_profile/key_filename
 key filename used by ssh connection.
 If left None, find key in ~/.ssh or use
 password for login

passphrase:

type: str | NoneType, optional,
 default: None
 argument path:
 simplify_mdata/model_devi/
 machine[SSHContext]/
 remote_profile/passphrase
 passphrase of key used by ssh connection

timeout:

type: int, optional, default: 10
 argument path:
 simplify_mdata/model_devi/
 machine[SSHContext]/
 remote_profile/timeout
 timeout of ssh connection

totp_secret:

type: str | NoneType, optional,
 default: None

argument path:
simplify_mdata/model_devi/
machine[SSHContext]/
remote_profile/totp_secret

Time-based one time password secret. It should be a base32-encoded string extracted from the 2D code.

tar_compress:

type: bool, optional, default: True
argument path:
simplify_mdata/model_devi/
machine[SSHContext]/
remote_profile/tar_compress

The archive will be compressed in upload and download if it is True. If not, compression will be skipped.

look_for_keys:

type: bool, optional, default: True
argument path:
simplify_mdata/model_devi/
machine[SSHContext]/
remote_profile/look_for_keys

enable searching for discoverable private key files in ~/.ssh/

When `context_type` is set to `LazyLocalContext` (or its aliases `lazylocalcontext`, `LazyLocal`, `lazylocal`):

remote_profile:

type: dict, optional
argument path:
simplify_mdata/model_devi/
machine[LazyLocalContext]/
remote_profile

The information used to maintain the connection with remote machine. This field is empty for this context.

When `context_type` is set to `HDFSContext` (or its aliases `hdfscontext`, `HDFS`, `hdfs`):

remote_profile:

type: dict, optional
argument path:
simplify_mdata/model_devi/
machine[HDFSContext]/remote_profile

The information used to maintain the connection with remote machine. This field is empty for this context.

When `context_type` is set to `BohriumContext` (or its aliases `bohriumcontext`, `Bohrium`, `bohrium`, `DpCloudServerContext`, `dpcloudservercontext`, `DpCloudServer`, `dpcloudserver`, `LebesgueContext`, `lebesguecontext`, `Lebesgue`, `lebesgue`):

remote_profile:

type: dict
 argument path: `simplify_mdata/
 model_devi/machine[BohriumContext]/
 remote_profile`

The information used to maintain the connection with remote machine.

email:

type: str, optional
 argument path:
`simplify_mdata/model_devi/
 machine[BohriumContext]/
 remote_profile/email`

Email

password:

type: str, optional
 argument path:
`simplify_mdata/model_devi/
 machine[BohriumContext]/
 remote_profile/password`

Password

program_id:

type: int, alias: *project_id*
 argument path:
`simplify_mdata/model_devi/
 machine[BohriumContext]/
 remote_profile/program_id`

Program ID

retry_count:

type: `NoneType` | int, optional,
 default: 2
 argument path:
`simplify_mdata/model_devi/
 machine[BohriumContext]/
 remote_profile/retry_count`

The retry count when a job is terminated

ignore_exit_code:

type: bool, optional, default: True

argument path:
simplify_mdata/model_devi/
machine[BohriumContext]/
remote_profile/
ignore_exit_code

The job state will be marked as finished if the exit code is non-zero when set to True. Otherwise,
the job state will be designated as terminated.

keep_backup:

type: bool, optional
argument path:
simplify_mdata/model_devi/
machine[BohriumContext]/
remote_profile/keep_backup

keep download and upload zip

input_data:

type: dict
argument path:
simplify_mdata/model_devi/
machine[BohriumContext]/
remote_profile/input_data

Configuration of job

When `context_type` is set to `OpenAPIContext` (or its aliases `openapicontext`, `OpenAPI`, `openapi`):

remote_profile:

type: dict, optional
argument path: simplify_mdata/
model_devi/machine[OpenAPIContext]/
remote_profile

The information used to maintain the connection with remote machine. This field is empty for this context.

When `context_type` is set to `LocalContext` (or its aliases `localcontext`, `Local`, `local`):

remote_profile:

type: dict, optional
argument path: simplify_mdata/
model_devi/machine[LocalContext]/
remote_profile

The information used to maintain the connection with remote machine. This field is empty for this context.

resources:

type: dict
 argument path:
 simplify_mdata/model_devi/resources

number_node:

type: int, optional, default: 1
 argument path: simplify_mdata/
 model_devi/resources/number_node

The number of node need for each *job*

cpu_per_node:

type: int, optional, default: 1
 argument path: simplify_mdata/
 model_devi/resources/cpu_per_node

cpu numbers of each node assigned to each job.

gpu_per_node:

type: int, optional, default: 0
 argument path: simplify_mdata/
 model_devi/resources/gpu_per_node

gpu numbers of each node assigned to each job.

queue_name:

type: str, optional, default: (empty string)
 argument path: simplify_mdata/
 model_devi/resources/queue_name

The queue name of batch job scheduler system.

group_size:

type: int
 argument path: simplify_mdata/
 model_devi/resources/group_size

The number of *tasks* in a *job*. 0 means infinity.

custom_flags:

type: typing.List[str], optional
 argument path: simplify_mdata/
 model_devi/resources/custom_flags

The extra lines pass to job submitting script header

strategy:

type: dict, optional
 argument path: simplify_mdata/
 model_devi/resources/strategy

strategies we use to generation job submitting scripts.

if_cuda_multi_devices:

type: bool, optional, default: False
argument path:
simplify_mdata/model_devi/
resources/strategy/
if_cuda_multi_devices

If there are multiple nvidia GPUS on the node, and we want to assign the tasks to different GPUS. If true, dpdispatcher will manually export environment variable CUDA_VISIBLE_DEVICES to different task. Usually, this option will be used with Task.task_need_resources variable simultaneously.

ratio_unfinished:

type: float, optional, default: 0.0
argument path: simplify_mdata/
model_devi/resources/
strategy/ratio_unfinished

The ratio of *tasks* that can be unfinished.

customized_script_header_template_file:

type: str, optional
argument path:
simplify_mdata/model_devi/
resources/strategy/
customized_script_header_template_file

The customized template file to generate job submitting script header, which overrides the default file.

para_deg:

type: int, optional, default: 1
argument path: simplify_mdata/
model_devi/resources/para_deg

Decide how many tasks will be run in parallel.

source_list:

type: typing.List[str], optional, default: []
argument path: simplify_mdata/
model_devi/resources/source_list

The env file to be sourced before the command execution.

module_purge:

type: bool, optional, default: False

argument path: simplify_mdata/
model_devi/resources/module_purge

Remove all modules on HPC system before
module load (module_list)

module_unload_list:

type: typing.List[str], optional, default:
[]

argument path:
simplify_mdata/model_devi/
resources/module_unload_list

The modules to be unloaded on HPC system
before submitting jobs

module_list:

type: typing.List[str], optional, default:
[]

argument path: simplify_mdata/
model_devi/resources/module_list

The modules to be loaded on HPC system be-
fore submitting jobs

envs:

type: dict, optional, default: {}

argument path: simplify_mdata/
model_devi/resources/envs

The environment variables to be exported on
before submitting jobs

prepend_script:

type: typing.List[str], optional, default:
[]

argument path: simplify_mdata/
model_devi/resources/prepend_script

Optional script run before jobs submitted.

append_script:

type: typing.List[str], optional, default:
[]

argument path: simplify_mdata/
model_devi/resources/append_script

Optional script run after jobs submitted.

wait_time:

type: float | int, optional, default: 0

argument path: `simplify_mdata/
model_devi/resources/wait_time`

The waiting time in second after a single *task* submitted

Depending on the value of *batch_type*, different sub args are accepted.

batch_type:

type: str (flag key)

argument path: `simplify_mdata/
model_devi/resources/batch_type`

possible choices: *Shell*, *PBS*,
DistributedShell, *SlurmJobArray*,
Slurm, *OpenAPI*, *Bohrrium*, *SGE*, *Torque*,
Fugaku, *LSF*

The batch job system type loaded from machine/*batch_type*.

When *batch_type* is set to *Shell* (or its alias *shell*):

kwargs:

type: dict, optional

argument path: `simplify_mdata/
model_devi/resources[Shell]/kwargs`

This field is empty for this batch.

When *batch_type* is set to *PBS* (or its alias *pbs*):

kwargs:

type: dict, optional

argument path: `simplify_mdata/
model_devi/resources[PBS]/kwargs`

This field is empty for this batch.

When *batch_type* is set to *DistributedShell* (or its alias *distributedshell*):

kwargs:

type: dict, optional

argument path:
`simplify_mdata/model_devi/
resources[DistributedShell]/kwargs`

This field is empty for this batch.

When *batch_type* is set to *SlurmJobArray* (or its alias *slurmjobarray*):

kwargs:

type: dict, optional

argument path:
 simplify_mdata/model_devi/
 resources[SlurmJobArray]/kwargs

Extra arguments.

custom_gpu_line:

type: str | NoneType, optional,
 default: None

argument path:
 simplify_mdata/model_devi/
 resources[SlurmJobArray]/
 kwargs/custom_gpu_line

Custom GPU configuration, starting
 with #SBATCH

slurm_job_size:

type: int, optional, default: 1

argument path:
 simplify_mdata/model_devi/
 resources[SlurmJobArray]/
 kwargs/slurm_job_size

Number of tasks in a Slurm job

When `batch_type` is set to Slurm (or its alias slurm):

kwargs:

type: dict, optional

argument path: simplify_mdata/
 model_devi/resources[Slurm]/kwargs

Extra arguments.

custom_gpu_line:

type: str | NoneType, optional,
 default: None

argument path: simplify_mdata/
 model_devi/resources[Slurm]/
 kwargs/custom_gpu_line

Custom GPU configuration, starting
 with #SBATCH

When `batch_type` is set to OpenAPI (or its alias openapi):

kwargs:

type: dict, optional

argument path:
 simplify_mdata/model_devi/
 resources[OpenAPI]/kwargs

This field is empty for this batch.

When `batch_type` is set to Bohrium (or its aliases bohrium,
 Lebesgue, lebesgue, DpCloudServer, dpcloudserver):

kwargs:

type: dict, optional
argument path:
simplify_mdata/model_devi/
resources[Bohrium]/kwargs

This field is empty for this batch.

When `batch_type` is set to SGE (or its alias `sge`):

kwargs:

type: dict, optional
argument path: simplify_mdata/
model_devi/resources[SGE]/kwargs

This field is empty for this batch.

When `batch_type` is set to Torque (or its alias `torque`):

kwargs:

type: dict, optional
argument path: simplify_mdata/
model_devi/resources[Torque]/kwargs

This field is empty for this batch.

When `batch_type` is set to Fugaku (or its alias `fugaku`):

kwargs:

type: dict, optional
argument path: simplify_mdata/
model_devi/resources[Fugaku]/kwargs

This field is empty for this batch.

When `batch_type` is set to LSF (or its alias `lsf`):

kwargs:

type: dict
argument path: simplify_mdata/
model_devi/resources[LSF]/kwargs

Extra arguments.

gpu_usage:

type: bool, optional, default: False
argument path: simplify_mdata/
model_devi/resources[LSF]/
kwargs/gpu_usage

Choosing if GPU is used in the calculation step.

gpu_new_syntax:

type: bool, optional, default: False

argument path: `simplify_mdata/
model_devi/resources[LSF]/
kwargs/gpu_new_syntax`

For LFS \geq 10.1.0.3, new option `-gpu`
for `#BSUB` could be used. If False,
and old syntax would be used.

gpu_exclusive:

type: bool, optional, default: True
argument path: `simplify_mdata/
model_devi/resources[LSF]/
kwargs/gpu_exclusive`

Only take effect when new syntax en-
abled. Control whether submit tasks
in exclusive way for GPU.

custom_gpu_line:

type: str | NoneType, optional,
default: None
argument path: `simplify_mdata/
model_devi/resources[LSF]/
kwargs/custom_gpu_line`

Custom GPU configuration, starting
with `#BSUB`

user_forward_files:

type: list, optional
argument path:
`simplify_mdata/model_devi/user_forward_files`
Files to be forwarded to the remote machine.

user_backward_files:

type: list, optional
argument path:
`simplify_mdata/model_devi/user_backward_files`
Files to be backwarded from the remote machine.

fp:

type: dict
argument path: `simplify_mdata/fp`
Parameters of command, machine, and resources for fp

command:

type: str
argument path: `simplify_mdata/fp/command`
Command of a program.

machine:

type: dict

argument path: `simplify_mdata/fp/machine`

batch_type:

type: `str`

argument path: `simplify_mdata/fp/machine/batch_type`

The batch job system type. Option: Bohrium, PBS, OpenAPI, DistributedShell, LSF, SlurmJobArray, Slurm, SGE, Fugaku, Torque, Shell

local_root:

type: `str | NoneType`

argument path: `simplify_mdata/fp/machine/local_root`

The dir where the tasks and relating files locate. Typically the project dir.

remote_root:

type: `str | NoneType`, optional

argument path: `simplify_mdata/fp/machine/remote_root`

The dir where the tasks are executed on the remote machine. Only needed when context is not lazy-local.

clean_asynchronously:

type: `bool`, optional, default: `False`

argument path: `simplify_mdata/fp/machine/clean_asynchronously`

Clean the remote directory asynchronously after the job finishes.

Depending on the value of *context_type*, different sub args are accepted.

context_type:

type: `str` (flag key)

argument path: `simplify_mdata/fp/machine/context_type`

possible choices: *SSHContext*, *LazyLocalContext*, *HDFSContext*, *BohriumContext*, *OpenAPIContext*, *LocalContext*

The connection used to remote machine. Option: *LocalContext*, *SSHContext*, *OpenAPIContext*, *BohriumContext*, *LazyLocalContext*, *HDFSContext*

When *context_type* is set to *SSHContext* (or its aliases *sshcontext*, *SSH*, *ssh*):

remote_profile:

type: dict

argument path: simplify_mdata/fp/
machine[SSHContext]/remote_profile

The information used to maintain the connection with remote machine.

hostname:

type: str

argument path: simplify_mdata/
fp/machine[SSHContext]/
remote_profile/hostname

hostname or ip of ssh connection.

username:

type: str

argument path: simplify_mdata/
fp/machine[SSHContext]/
remote_profile/username

username of target linux system

password:

type: str, optional

argument path: simplify_mdata/
fp/machine[SSHContext]/
remote_profile/password

(deprecated) password of linux system. Please use [SSH keys](#) instead to improve security.

port:

type: int, optional, default: 22

argument path: simplify_mdata/
fp/machine[SSHContext]/
remote_profile/port

ssh connection port.

key_filename:

type: str | NoneType, optional,
default: None

argument path: simplify_mdata/
fp/machine[SSHContext]/
remote_profile/key_filename

key filename used by ssh connection.
If left None, find key in ~/.ssh or use
password for login

passphrase:

type: str | NoneType, optional,
default: None

argument path: `simplify_mdata/
fp/machine[SSHContext]/
remote_profile/passphrase`

passphrase of key used by ssh connection

timeout:

type: `int`, optional, default: `10`
argument path: `simplify_mdata/
fp/machine[SSHContext]/
remote_profile/timeout`

timeout of ssh connection

totp_secret:

type: `str | NoneType`, optional,
default: `None`
argument path: `simplify_mdata/
fp/machine[SSHContext]/
remote_profile/totp_secret`

Time-based one time password secret. It should be a base32-encoded string extracted from the 2D code.

tar_compress:

type: `bool`, optional, default: `True`
argument path: `simplify_mdata/
fp/machine[SSHContext]/
remote_profile/tar_compress`

The archive will be compressed in upload and download if it is `True`. If not, compression will be skipped.

look_for_keys:

type: `bool`, optional, default: `True`
argument path: `simplify_mdata/
fp/machine[SSHContext]/
remote_profile/look_for_keys`

enable searching for discoverable private key files in `~/.ssh/`

When `context_type` is set to `LazyLocalContext` (or its aliases `lazylocalcontext`, `LazyLocal`, `lazylocal`):

remote_profile:

type: `dict`, optional
argument path: `simplify_mdata/fp/
machine[LazyLocalContext]/
remote_profile`

The information used to maintain the connection with remote machine. This field is empty for this context.

When `context_type` is set to `HDFSContext` (or its aliases `hdfscontext`, `HDFS`, `hdfs`):

remote_profile:

type: dict, optional
 argument path: `simplify_mdata/fp/machine[HDFSContext]/remote_profile`

The information used to maintain the connection with remote machine. This field is empty for this context.

When `context_type` is set to `BohriumContext` (or its aliases `bohriumcontext`, `Bohrium`, `bohrium`, `DpCloudServerContext`, `dpcloudservercontext`, `DpCloudServer`, `dpcloudserver`, `LebesgueContext`, `lebesguecontext`, `Lebesgue`, `lebesgue`):

remote_profile:

type: dict
 argument path: `simplify_mdata/fp/machine[BohriumContext]/remote_profile`

The information used to maintain the connection with remote machine.

email:

type: str, optional
 argument path: `simplify_mdata/fp/machine[BohriumContext]/remote_profile/email`

Email

password:

type: str, optional
 argument path: `simplify_mdata/fp/machine[BohriumContext]/remote_profile/password`

Password

program_id:

type: int, alias: *project_id*
 argument path: `simplify_mdata/fp/machine[BohriumContext]/remote_profile/program_id`

Program ID

retry_count:

type: `NoneType` | int, optional,
 default: 2
 argument path: `simplify_mdata/fp/machine[BohriumContext]/remote_profile/retry_count`

The retry count when a job is terminated

ignore_exit_code:

type: bool, optional, default: True
argument path: simplify_mdata/
fp/machine[BohriumContext]/
remote_profile/
ignore_exit_code

The job state will be marked as finished if the exit code is non-zero when set to True. Otherwise, the job state will be designated as terminated.

keep_backup:

type: bool, optional
argument path: simplify_mdata/
fp/machine[BohriumContext]/
remote_profile/keep_backup

keep download and upload zip

input_data:

type: dict
argument path: simplify_mdata/
fp/machine[BohriumContext]/
remote_profile/input_data

Configuration of job

When `context_type` is set to `OpenAPIContext` (or its aliases `openapicontext`, `OpenAPI`, `openapi`):

remote_profile:

type: dict, optional
argument path: simplify_mdata/fp/
machine[OpenAPIContext]/
remote_profile

The information used to maintain the connection with remote machine. This field is empty for this context.

When `context_type` is set to `LocalContext` (or its aliases `localcontext`, `Local`, `local`):

remote_profile:

type: dict, optional
argument path: simplify_mdata/fp/
machine[LocalContext]/
remote_profile

The information used to maintain the connection with remote machine. This field is empty for this context.

resources:

type: dict

argument path: `simplify_mdata/fp/resources`

number_node:

type: int, optional, default: 1

argument path: `simplify_mdata/fp/resources/number_node`

The number of node need for each *job*

cpu_per_node:

type: int, optional, default: 1

argument path: `simplify_mdata/fp/resources/cpu_per_node`

cpu numbers of each node assigned to each job.

gpu_per_node:

type: int, optional, default: 0

argument path: `simplify_mdata/fp/resources/gpu_per_node`

gpu numbers of each node assigned to each job.

queue_name:

type: str, optional, default: (empty string)

argument path: `simplify_mdata/fp/resources/queue_name`

The queue name of batch job scheduler system.

group_size:

type: int

argument path: `simplify_mdata/fp/resources/group_size`

The number of *tasks* in a *job*. 0 means infinity.

custom_flags:

type: `typing.List[str]`, optional

argument path: `simplify_mdata/fp/resources/custom_flags`

The extra lines pass to job submitting script header

strategy:

type: dict, optional

argument path: `simplify_mdata/fp/resources/strategy`

strategies we use to generation job submitting scripts.

if_cuda_multi_devices:

type: bool, optional, default: False
argument path: simplify_mdata/
fp/resources/strategy/
if_cuda_multi_devices

If there are multiple nvidia GPUS on the node, and we want to assign the tasks to different GPUS. If true, dpdispatcher will manually export environment variable CUDA_VISIBLE_DEVICES to different task. Usually, this option will be used with Task.task_need_resources variable simultaneously.

ratio_unfinished:

type: float, optional, default: 0.0
argument path:
simplify_mdata/fp/resources/
strategy/ratio_unfinished

The ratio of *tasks* that can be unfinished.

customized_script_header_template_file:

type: str, optional
argument path: simplify_mdata/
fp/resources/strategy/
customized_script_header_template_file

The customized template file to generate job submitting script header, which overrides the default file.

para_deg:

type: int, optional, default: 1
argument path: simplify_mdata/fp/
resources/para_deg

Decide how many tasks will be run in parallel.

source_list:

type: typing.List[str], optional, default: []
argument path: simplify_mdata/fp/
resources/source_list

The env file to be sourced before the command execution.

module_purge:

type: bool, optional, default: False
argument path: simplify_mdata/fp/
resources/module_purge

Remove all modules on HPC system before module load (module_list)

module_unload_list:

type: typing.List[str], optional, default: []

argument path: simplify_mdata/fp/resources/module_unload_list

The modules to be unloaded on HPC system before submitting jobs

module_list:

type: typing.List[str], optional, default: []

argument path: simplify_mdata/fp/resources/module_list

The modules to be loaded on HPC system before submitting jobs

envs:

type: dict, optional, default: {}

argument path: simplify_mdata/fp/resources/envs

The environment variables to be exported on before submitting jobs

prepend_script:

type: typing.List[str], optional, default: []

argument path: simplify_mdata/fp/resources/prepend_script

Optional script run before jobs submitted.

append_script:

type: typing.List[str], optional, default: []

argument path: simplify_mdata/fp/resources/append_script

Optional script run after jobs submitted.

wait_time:

type: float | int, optional, default: 0

argument path: simplify_mdata/fp/resources/wait_time

The waiting time in second after a single *task* submitted

Depending on the value of *batch_type*, different sub args are accepted.

batch_type:

type: str (flag key)
argument path: `simplify_mdata/fp/resources/batch_type`
possible choices: *Shell*, *PBS*,
DistributedShell, *SlurmJobArray*,
Slurm, *OpenAPI*, *Bohrium*, *SGE*, *Torque*,
Fugaku, *LSF*
The batch job system type loaded from machine/`batch_type`.

When `batch_type` is set to *Shell* (or its alias *shell*):

kwargs:

type: dict, optional
argument path: `simplify_mdata/fp/resources[Shell]/kwargs`
This field is empty for this batch.

When `batch_type` is set to *PBS* (or its alias *pbs*):

kwargs:

type: dict, optional
argument path: `simplify_mdata/fp/resources[PBS]/kwargs`
This field is empty for this batch.

When `batch_type` is set to *DistributedShell* (or its alias *distributedshell*):

kwargs:

type: dict, optional
argument path: `simplify_mdata/fp/resources[DistributedShell]/kwargs`
This field is empty for this batch.

When `batch_type` is set to *SlurmJobArray* (or its alias *slurmjobarray*):

kwargs:

type: dict, optional
argument path: `simplify_mdata/fp/resources[SlurmJobArray]/kwargs`
Extra arguments.

custom_gpu_line:

type: str | NoneType, optional,
default: None
argument path: `simplify_mdata/fp/resources[SlurmJobArray]/kwargs/custom_gpu_line`

Custom GPU configuration, starting with #SBATCH

slurm_job_size:

type: int, optional, default: 1
 argument path: simplify_mdata/
 fp/resources[SlurmJobArray]/
 kwargs/slurm_job_size

Number of tasks in a Slurm job

When `batch_type` is set to Slurm (or its alias `slurm`):

kwargs:

type: dict, optional
 argument path: simplify_mdata/fp/
 resources[Slurm]/kwargs

Extra arguments.

custom_gpu_line:

type: str | NoneType, optional,
 default: None
 argument path: simplify_mdata/
 fp/resources[Slurm]/kwargs/
 custom_gpu_line

Custom GPU configuration, starting with #SBATCH

When `batch_type` is set to OpenAPI (or its alias `openapi`):

kwargs:

type: dict, optional
 argument path: simplify_mdata/fp/
 resources[OpenAPI]/kwargs

This field is empty for this batch.

When `batch_type` is set to Bohrium (or its aliases `bohrium`, `Lebesgue`, `lebesgue`, `DpCloudServer`, `dpcloudserver`):

kwargs:

type: dict, optional
 argument path: simplify_mdata/fp/
 resources[Bohrium]/kwargs

This field is empty for this batch.

When `batch_type` is set to SGE (or its alias `sge`):

kwargs:

type: dict, optional
 argument path: simplify_mdata/fp/
 resources[SGE]/kwargs

This field is empty for this batch.

When `batch_type` is set to Torque (or its alias `torque`):

kwargs:

type: dict, optional
argument path: `simplify_mdata/fp/
resources[Torque]/kwargs`

This field is empty for this batch.

When `batch_type` is set to Fugaku (or its alias `fugaku`):

kwargs:

type: dict, optional
argument path: `simplify_mdata/fp/
resources[Fugaku]/kwargs`

This field is empty for this batch.

When `batch_type` is set to LSF (or its alias `lsf`):

kwargs:

type: dict
argument path: `simplify_mdata/fp/
resources[LSF]/kwargs`

Extra arguments.

gpu_usage:

type: bool, optional, default: False
argument path: `simplify_mdata/
fp/resources[LSF]/kwargs/
gpu_usage`

Choosing if GPU is used in the calculation step.

gpu_new_syntax:

type: bool, optional, default: False
argument path: `simplify_mdata/
fp/resources[LSF]/kwargs/
gpu_new_syntax`

For LFS >= 10.1.0.3, new option `-gpu` for `#BSUB` could be used. If False, and old syntax would be used.

gpu_exclusive:

type: bool, optional, default: True
argument path: `simplify_mdata/
fp/resources[LSF]/kwargs/
gpu_exclusive`

Only take effect when new syntax enabled. Control whether submit tasks in exclusive way for GPU.

custom_gpu_line:

type: str | NoneType, optional,

default: None

argument path: simplify_mdata/

fp/resources[LSF]/kwargs/

custom_gpu_line

Custom GPU configuration, starting
with #BSUB

user_forward_files:

type: list, optional

argument path:

simplify_mdata/fp/user_forward_files

Files to be forwarded to the remote machine.

user_backward_files:

type: list, optional

argument path:

simplify_mdata/fp/user_backward_files

Files to be backwarded from the remote machine.

AUTO TEST

7.1 Autotest Overview: Autotest for Deep Generator

Suppose that we have a potential (can be DFT, DP, MEAM...), `autotest` helps us automatically calculate M properties on N configurations. The folder where the `autotest` runs is called the working directory of `autotest`. Different potentials should be tested in different working directories.

A property is tested in three steps: `make`, `run` and `post`. `make` prepares all computational tasks that are needed to calculate the property. For example to calculate EOS, `make` prepares a series of tasks, each of which has a scaled configuration with certain volume, and all necessary input files necessary for starting a VASP, ABACUS, or LAMMPS calculations. `run` sends all the computational tasks to remote computational resources defined in a machine configuration file like `machine.json`, and automatically collects the results when remote calculations finish. `post` calculates the desired property from the collected results.

7.1.1 Relaxation

The relaxation of a structure should be carried out before calculating all other properties:

```
dpngen autotest make relax.json
dpngen autotest run relax.json machine.json
dpngen autotest post relax.json
```

If, for some reasons, the main program terminated at stage `run`, one can easily restart with the same command. `relax.json` is the parameter file. An example for `deepmd` relaxation is given as:

```
{
  "structures": ["confs/mp-*"],
  "interaction": {
    "type": "deepmd",
    "model": "frozen_model.pb",
    "type_map": {"Al": 0, "Mg": 1}
  },
  "relaxation": {}
}
```

where the key `structures` provides the structures to relax. `interaction` is provided with `deepmd`, and other options are `vasp`, `abacus`, `meam`...

7.1.2 Task type

There are now six task types implemented in the package: `vasp`, `abacus`, `deepmd`, `meam`, `eam_fs`, and `eam_alloy`. An `inter.json` file in json format containing the interaction parameters will be written in the directory of each task after make. We give input examples of the `interaction` part for each type below:

VASP:

The default of `potcar_prefix` is `""`.

```
"interaction": {
  "type": "vasp",
  "incar": "vasp_input/INCAR",
  "potcar_prefix": "vasp_input",
  "potcars": {"Al": "POTCAR.al", "Mg": "POTCAR.mg"}
}
```

ABACUS:

The default of `potcar_prefix` is `""`. The path of `potcars/orb_files/deepks_desc` is `potcar_prefix + potcars/orb_files/deepks_desc/deepks_model`.

```
"interaction": {
  "type": "abacus",
  "incar": "abacus_input/INPUT",
  "potcar_prefix": "abacus_input",
  "potcars": {"Al": "pseudo_potential.al", "Mg": "pseudo_potential.
↪mg"},
  "orb_files": {"Al": "numerical_orb.al", "Mg": "numerical_orb.mg"},
  "atom_masses": {"Al": 26.9815, "Mg": 24.305},
  "deepks_desc": "jle.orb",
  "deepks_model": "model.ptg"
}
```

deepmd:

Only 1 model can be used in autotest in one working directory.

```
"interaction": {
  "type": "deepmd",
  "model": "frozen_model.pb",
  "type_map": {"Al": 0, "Mg": 1}
}
```

meam:

Please make sure the [USER-MEAMC](#) package has already been installed in LAMMPS.

```
"interaction": {
  "type": "meam",
  "model": ["meam.lib", "AlMg.meam"],
  "type_map": {"Al": 1, "Mg": 2}
}
```

eam_fs & eam_alloy:

Please make sure the [MANYBODY](#) package has already been installed in LAMMPS

```

"interaction": {
  "type":          "eam_fs (eam_alloy)",
  "model":         "AlMg.eam.fs (AlMg.eam.alloy)",
  "type_map":      {"Al": 1, "Mg": 2}
}

```

7.1.3 Property type

Now the supported property types are `eos`, `elastic`, `vacancy`, `interstitial`, `surface`, and `gamma`. Before property tests, `relaxation` should be done first or the relaxation results should be present in the corresponding directory `confs/mp-*/relaxation/relax_task`. A file named `task.json` in json format containing the property parameter will be written in the directory of each task after make step. Multiple property tests can be performed simultaneously.

7.2 Make run and post

There are three operations in auto test package, namely `make`, `run`, and `post`. Here we take `eos` property as an example for property type.

7.2.1 Make

The INCAR, POSCAR, POTCAR input files for VASP or `in.lammps`, `conf.lmp`, and the interatomic potential files for LAMMPS will be generated in the directory `confs/mp-*/relaxation/relax_task` for relaxation or `confs/mp-*/eos_00/task.[0-9]*[0-9]` for EOS. The `machine.json` file is not needed for make. Example:

```
dpngen autotest make relaxation.json
```

7.2.2 Run

The jobs would be dispatched according to the parameter in `machine.json` file and the calculation results would be sent back. Example:

```
dpngen autotest run relaxation.json machine.json
```

7.2.3 Post

The post process of calculation results would be performed. `result.json` in json format will be generated in `confs/mp-*/relaxation/relax_task` for relaxation and `result.json` in json format and `result.out` in txt format in `confs/mp-*/eos_00` for EOS. The `machine.json` file is also not needed for post. Example:

```
dpngen autotest post relaxation.json
```

7.3 Relaxation

7.3.1 Relaxation get started and input examples

The relaxation of a structure should be carried out before calculating all other properties.

First, we need input parameter file and we name it `relax.json` here. All the relaxation calculations should be taken either by VASP, ABACUS, or LAMMPS. Here are two input examples for VASP and LAMMPS respectively.

An example of the input file for relaxation by VASP:

```
{
  "structures":          ["confs/std-*"],
  "interaction": {
    "type":              "vasp",
    "incar":              "vasp_input/INCAR",
    "potcar_prefix":     "vasp_input",
    "potcars":           {"Al": "POTCAR.al"}
  },
  "relaxation": {
    "cal_type":           "relaxation",
    "cal_setting": {
      "relax_pos":        true,
      "relax_shape":      true,
      "relax_vol":        true,
      "ediff":            1e-6,
      "ediffg":           -0.01,
      "encut":            650,
      "kspacing":         0.1,
      "kgamma":           false}
    }
}
```

Key words	data ture	struc- ture	example		description
structures	List of String		["confs/std-*"]		path of different structures
interaction	Dict		See above		description of the task type and atomic interaction
type	String		"vasp"		task type
incar	String		"vasp_input/INCAR"		path for INCAR file in vasp
pot- car_prefix	String		"vasp_input"		prefix of path for POTCAR file in vasp, default = ""
potcars	Dict		{ "Al": CAR.al" }	"POT-	key is element type and value is potcar name
relaxation	Dict		See above		calculation type and setting for relaxation
cal_type	String		"relaxation" "static"	or	calculation type
cal_setting	Dict		See above		calculation setting
relax_pos	Boolean		true		relax atomic position or not, default = true for relaxation
relax_shape	Boolean		true		relax box shape or not, default = true for relaxation
relax_vol	Boolean		true		relax box volume or not, default = true for relaxation
ediff	Float		1e-6		set EDIFF parameter in INCAR files
ediffg	Float		-0.01		set EDIFFG parameter in INCAR files
encut	Int		650		set encut parameter in INCAR files
kspacing	Float		0.1		set KSPACING parameter in INCAR files
kgamma	Boolean		false		set KGAMMA parameter in INCAR files

An example of the input file for relaxation by LAMMPS:

```
{
  "structures":      ["confs/std-*"],
  "interaction": {
    "type":          "deepmd",
    "model":          "frozen_model.pb",
    "in_lammps":     "lammps_input/in.lammps",
    "type_map":      {"Al": 0}
  },
  "relaxation": {
    "cal_setting": {"etol": 0,
                  "ftol": 1e-10,
                  "maxiter": 5000,
                  "maximal": 500000}
  }
}
```

Other key words different from vasp:

Key words	data structure	example	description
model	String or List of String	“frozen_model.pb”	model file for atomic interaction
in_lammps	String	“lammps_input/in.lan	input file for lammps commands
type_map	Dict	{“Al”: 0}	key is element type and value is type number. DP starts from 0, others starts from 1
etol	Float	0	stopping tolerance for energy
ftol	Float	1e-10	stopping tolerance for force
maxiter	Int	5000	max iterations of minimizer
maxeval	Int	500000	max number of force/energy evaluations

For LAMMPS relaxation and all the property calculations, **package will help to generate in.lammps file for user automatically** according to the property type. We can also make the final changes in the `minimize` setting (`minimize etol ftol maxiter maxeval`) in `in.lammps`. In addition, users can apply the input file for lammps commands in the interaction part. For further information of the LAMMPS relaxation, we refer users to [minimize command](#).

7.3.2 Relaxation make

The list of the directories storing structures are `["confs/std-*"]` in the previous example. For single element system, if POSCAR doesn't exist in the directories: `std-fcc`, `std-hcp`, `std-dhcp`, `std-bcc`, `std-diamond`, and `std-sc`, the package will automatically generate the standard crystal structures **fcc**, **hcp**, **dhcp**, **bcc**, **diamond**, and **sc** in the corresponding directories, respectively. In other conditions and for multi-component system (more than 1), if POSCAR doesn't exist, the package will terminate and print the error **“no configuration for autotest”**.

VASP relaxation

Take the input example of Al in the previous section, when we do `make` as follows:

```
dpngen autotest make relaxation.json
```

the following files would be generated:

```
tree confs/std-fcc/relaxation/
```

```
confs/std-fcc/relaxation/
|-- INCAR
|-- POTCAR
|-- relax_task
|   |-- INCAR -> ../INCAR
|   |-- inter.json
|   |-- KPOINTS
|   |-- POSCAR -> ../../POSCAR
|   |-- POTCAR -> ../POTCAR
|-- task.json
```

`inter.json` records the information in the interaction dictionary and `task.json` records the information in the relaxation dictionary.

LAMMPS relaxation

```
dpngen autotest make relaxation.json
tree confs/std-fcc/
```

the output would be:

```
confs/std-fcc/
|-- POSCAR
|-- relaxation
|   |-- frozen_model.pb -> ../../frozen_model.pb
|   |-- in.lammps
|   |-- relax_task
|       |-- conf.lmp
|       |-- frozen_model.pb -> ../frozen_model.pb
|       |-- in.lammps -> ../in.lammps
|       |-- inter.json
|       |-- POSCAR -> ../../POSCAR
|       |-- task.json
```

the `conf.lmp` is the input configuration and `in.lammps` is the input command file for lammps.

in.lammps: the package would generate the file `confs/mp-*/relaxation/in.lammps` as follows and we refer the user to the further information of `fix box/relax` function in lammps:

```
clear
units                metal
dimension            3
boundary             p p p
atom_style           atomic
box                  tilt large
read_data            conf.lmp
mass                 1 26.982
neigh_modify         every 1 delay 0 check no
pair_style deepmd    frozen_model.pb
pair_coeff
compute             mype all pe
thermo              100
thermo_style         custom step pe pxx pyy pzz pxy pxz pyz lx ly lz vol c_mype
dump                1 all custom 100 dump.relax id type xs ys zs fx fy fz
min_style            cg
fix                  1 all box/relax iso 0.0
minimize             0 1.000000e-10 5000 500000
fix                  1 all box/relax aniso 0.0
minimize             0 1.000000e-10 5000 500000
variable             N equal count(all)
variable             V equal vol
variable             E equal "c_mype"
variable             tmp1x equal lx
variable             tmp1y equal ly
variable             Pxx equal pxx
variable             Pyy equal pyy
variable             Pzz equal pzz
variable             Pxy equal pxy
```

(continues on next page)

(continued from previous page)

```

variable      Pxz equal pxz
variable      Pyz equal pyz
variable      Epa equal ${E}/${N}
variable      Vpa equal ${V}/${N}
variable      AA equal (${tmplx}*${tmply})
print "All done"
print "Total number of atoms = ${N}"
print "Final energy per atoms = ${Epa}"
print "Final volume per atoms = ${Vpa}"
print "Final Base area = ${AA}"
print "Final Stress (xx yy zz xy xz yz) = ${Pxx} ${Pyy} ${Pzz} ${Pxy} ${Pxz} ${Pyz}"

```

If user provides lammps input command file `in.lammps`, the `thermo_style` and `dump` commands should be the same as the above file.

interatomic potential model: the `frozen_model.pb` in `confs/mp-*/relaxation` would link to the `frozen_model.pb` file given in the input.

7.3.3 Relaxation run

The work path of each task should be in the form like `confs/mp-*/relaxation` and all task is in the form like `confs/mp-*/relaxation/relax_task`.

The `machine.json` file should be applied in this process and the machine parameters (eg. GPU or CPU) are determined according to the task type (VASP or LAMMPS). Then in each work path, the corresponding tasks would be submitted and the results would be sent back through [make_dispatcher](#).

Take `deepmd` run for example:

```

nohup dpngen autotest run relaxation.json machine-ali.json > run.result 2>&1 &
tree confs/std-fcc/relaxation/

```

the output would be:

```

confs/std-fcc/relaxation/
|-- frozen_model.pb -> ../../../../frozen_model.pb
|-- in.lammps
|-- jr.json
|-- relax_task
|   |-- conf.lmp
|   |-- dump.relax
|   |-- frozen_model.pb -> ../frozen_model.pb
|   |-- in.lammps -> ../in.lammps
|   |-- inter.json
|   |-- log.lammps
|   |-- outlog
|   |-- POSCAR -> ../../POSCAR
|-- task.json

```

`dump.relax` is the file storing configurations and `log.lammps` is the output file for lammps.

7.3.4 Relaxation post

Take deepmd post for example:

```
dpngen autotest post relaxation.json
tree confs/std-fcc/relaxation/
```

the output will be:

```
confs/std-fcc/relaxation/
|-- frozen_model.pb -> ../../../../frozen_model.pb
|-- in.lammps
|-- jr.json
|-- relax_task
|   |-- conf.lmp
|   |-- CONTCAR
|   |-- dump.relax
|   |-- frozen_model.pb -> ../frozen_model.pb
|   |-- in.lammps -> ../in.lammps
|   |-- inter.json
|   |-- log.lammps
|   |-- outlog
|   |-- POSCAR -> ../../POSCAR
|   |-- result.json
|-- task.json
```

result.json stores the box cell, coordinates, energy, force, virial,... information of each frame in the relaxation trajectory and CONTCAR is the final equilibrium configuration.

result.json:

```
{
  "@module": "dpdata.system",
  "@class": "LabeledSystem",
  "data": {
    "atom_nums": [
      1
    ],
    "atom_names": [
      "Al"
    ],
    "atom_types": {
      "@module": "numpy",
      "@class": "array",
      "dtype": "int64",
      "data": [
        0
      ]
    },
    "orig": {
      "@module": "numpy",
      "@class": "array",
      "dtype": "int64",
      "data": [
```

(continues on next page)

(continued from previous page)

```

        0,
        0,
        0
    ]
},
"cells": {
    "@module": "numpy",
    "@class": "array",
    "dtype": "float64",
    "data": [
        [
            [
                2.8637824638,
                0.0,
                0.0
            ],
            [
                1.4318912319,
                2.4801083646,
                0.0
            ],
            [
                1.4318912319,
                0.8267027882,
                2.3382685902
            ]
        ],
        [
            [
                2.8549207998018438,
                0.0,
                0.0
            ],
            [
                1.4274603999009239,
                2.472433938457684,
                0.0
            ],
            [
                1.4274603999009212,
                0.8241446461525599,
                2.331033071844216
            ]
        ],
        [
            [
                2.854920788303194,
                0.0,
                0.0
            ],
            [
                1.427460394144466,

```

(continues on next page)

(continued from previous page)

```

                2.472433928487206,
                0.0
            ],
            [
                1.427460394154763,
                0.8241446428350139,
                2.331033062460779
            ]
        ]
    ],
    "coords": {
        "@module": "numpy",
        "@class": "array",
        "dtype": "float64",
        "data": [
            [
                [
                    0.0,
                    0.0,
                    0.0
                ]
            ],
            [
                [
                    5.709841595683707e-25,
                    -4.3367974740910857e-19,
                    0.0
                ]
            ],
            [
                [
                    -8.673606219968035e-19,
                    8.673619637565944e-19,
                    8.673610853102186e-19
                ]
            ]
        ]
    },
    "energies": {
        "@module": "numpy",
        "@class": "array",
        "dtype": "float64",
        "data": [
            -3.745029,
            -3.7453815,
            -3.7453815
        ]
    },
    "forces": {
        "@module": "numpy",
        "@class": "array",

```

(continues on next page)

(continued from previous page)

```

    "dtype": "float64",
    "data": [
        [
            [
                0.0,
                -6.93889e-18,
                -3.46945e-18
            ]
        ],
        [
            [
                1.38778e-17,
                6.93889e-18,
                -1.73472e-17
            ]
        ],
        [
            [
                1.38778e-17,
                1.73472e-17,
                -4.51028e-17
            ]
        ]
    ],
    "virials": {
        "@module": "numpy",
        "@class": "array",
        "dtype": "float64",
        "data": [
            [
                [
                    -0.07534992071654338,
                    1.2156615579052586e-17,
                    1.3904892126132796e-17
                ],
                [
                    1.2156615579052586e-17,
                    -0.07534992071654338,
                    4.61571024026576e-12
                ],
                [
                    1.3904892126132796e-17,
                    4.61571024026576e-12,
                    -0.07534992071654338
                ]
            ],
            [
                [
                    -9.978994290457664e-08,
                    -3.396452753975288e-15,
                    8.785831629151552e-16
                ]
            ]
        ]
    }

```

(continues on next page)

(continued from previous page)

```

    ],
    [
        -3.396452753975288e-15,
        -9.991375413666671e-08,
        5.4790751628409565e-12
    ],
    [
        8.785831629151552e-16,
        5.4790751628409565e-12,
        -9.973497959053003e-08
    ]
],
[
    [
        1.506940521266962e-11,
        1.1152016233536118e-11,
        -8.231900529157644e-12
    ],
    [
        1.1152016233536118e-11,
        -6.517665029355618e-11,
        -6.33706710415926e-12
    ],
    [
        -8.231900529157644e-12,
        -6.33706710415926e-12,
        5.0011471096530724e-11
    ]
]
]
},
"stress": {
    "@module": "numpy",
    "@class": "array",
    "dtype": "float64",
    "data": [
        [
            -7.2692250000000005,
            1.1727839e-15,
            1.3414452e-15
        ],
        [
            1.1727839e-15,
            -7.2692250000000005,
            4.4529093000000003e-10
        ],
        [
            1.3414452e-15,
            4.4529093000000003e-10,
            -7.2692250000000005
        ]
    ]
}

```

(continues on next page)

(continued from previous page)

```

    ],
    [
      [
        -9.71695e-06,
        -3.3072633e-13,
        8.5551193e-14
      ],
      [
        -3.3072633e-13,
        -9.7290060000000001e-06,
        5.3351969e-10
      ],
      [
        8.5551193e-14,
        5.3351969e-10,
        -9.711598e-06
      ]
    ],
    [
      [
        1.4673689e-09,
        1.0859169e-09,
        -8.0157343e-10
      ],
      [
        1.0859169e-09,
        -6.3465139e-09,
        -6.1706584e-10
      ],
      [
        -8.0157343e-10,
        -6.1706584e-10,
        4.8698191e-09
      ]
    ]
  ]
}

```

7.4 Property

7.4.1 Property get started and input examples

Here we take deepmd for example and the input file for other task types is similar.

```

{
  "structures": ["confs/std-*"],
  "interaction": {
    "type": "deepmd",

```

(continues on next page)

(continued from previous page)

```

    "model":      "frozen_model.pb",
    "type_map":   {"Al": 0}
  },
  "properties": [
    {
      "type":      "eos",
      "vol_start": 0.9,
      "vol_end":   1.1,
      "vol_step":  0.01
    },
    {
      "type":      "elastic",
      "norm_deform": 1e-2,
      "shear_deform": 1e-2
    },
    {
      "type":      "vacancy",
      "supercell":  [3, 3, 3],
      "start_confs_path": "../vasp/confs"
    },
    {
      "type":      "interstitial",
      "supercell":  [3, 3, 3],
      "insert_ele": ["Al"],
      "conf_filters": {"min_dist": 1.5},
      "cal_setting": {"input_prop": "lammps_input/lammps_high"}
    },
    {
      "type":      "surface",
      "min_slab_size": 10,
      "min_vacuum_size": 11,
      "max_miller":   2,
      "cal_type":     "static"
    },
    {
      "type": "gamma",
      "lattice_type": "fcc",
      "miller_index": [1, 1, 1],
      "displace_direction": [1, 1, 0],
      "supercell_size": [1, 1, 10],
      "min_vacuum_size": 10,
      "add_fix": ["true", "true", "false"],
      "n_steps": 20
    }
  ]
}

```

Universal key words for properties

Key words	data structure	example	description
type	String	“eos”	property type
skip	Boolean	true	whether to skip current property or not
start_conf_path	String	“../vasp/confs”	start from the equilibrium configuration in other path only for the current property type
cal_setting[“input_prop”]	String	“lammps_input/lamr	input commands file
cal_setting[“overwrite_i”]	Dict		overwrite the interaction in the interaction part only for the current property type

other parameters in **cal_setting** and **cal_type** in **relaxation** also apply in **property**.

Key words for **EOS**

Key words	data structure	example	description
vol_start	Float	0.9	the starting volume related to the equilibrium structure
vol_end	Float	1.1	the biggest volume related to the equilibrium structure
vol_step	Float	0.01	the volume increment related to the equilibrium structure
vol_abs	Boolean	false	whether to treat vol_start, vol_end and vol_step as absolute volume or not (as relative volume), default = false

Key words for **Elastic**

Key words	data structure	example	description
norm_deform	Float	1e-2	deformation in xx, yy, zz, default = 1e-2
shear_deform	Float	1e-2	deformation in other directions, default = 1e-2

Key words for **Vacancy**

Key words	data structure	example	description
supercell	List of Int	[3,3,3]	the supercell to be constructed, default = [1,1,1]

Key words for **Interstitial**

Key words	data structure	example	description
in-sert_ele	List of String	[“Al”]	the element to be inserted
supercell	List of Int	[3,3,3]	the supercell to be constructed, default = [1,1,1]
conf_filters	Dict	“min_dist”: 1.5	filter out the undesirable configuration
bcc_self	Boolean	false	whether to do the self-interstitial calculations for bcc structures, default = false

Key words for **Surface**

Key words	data structure	example	description
min_slab_size	Int	10	minimum size of slab thickness
min_vacuum_size	Int	11	minimum size of vacuum width
pert_xz	Float	0.01	perturbation through xz direction used to compute surface energy, default = 0.01
max_miller	Int	2	the maximum miller index, default = 2

Key words for **Gamma**

Key words	data structure	example	description
lattice_type	String	“fcc”	“bcc” or “fcc” at this stage
miller_index	List of Int	[1,1,1]	slip plane for gamma-line calculation
displace_direction	List of Int	[1,1,0]	slip direction for gamma-line calculation
supercell_size	List of Int	[1,1,10]	the supercell to be constructed, default = [1,1,5]
min_vacuum_size	Int or Float	10	minimum size of vacuum width, default = 20
add_fix	List of String	['true','true','false']	whether to fix atoms in the direction, default = ['true','true','false'] (standard method)
n_steps	Int	20	Number of points for gamma-line calculation, default = 10

7.4.2 Property make

```
dpngen autotest make property.json
```

EOS output:

```
confs/std-fcc/eos_00/
|-- frozen_model.pb -> ../../../../frozen_model.pb
|-- task.000000
|   |-- conf.lmp
|   |-- eos.json
|   |-- frozen_model.pb -> ../frozen_model.pb
|   |-- in.lammps
|   |-- inter.json
|   |-- POSCAR
|   |-- POSCAR.orig -> ../../relaxation/relax_task/CONTCAR
|   |-- task.json
|-- task.000001
|   |-- conf.lmp
|   |-- eos.json
|   |-- frozen_model.pb -> ../frozen_model.pb
|   |-- in.lammps
|   |-- inter.json
|   |-- POSCAR
|   |-- POSCAR.orig -> ../../relaxation/relax_task/CONTCAR
```

(continues on next page)

(continued from previous page)

```
|  -- task.json
...
|-- task.000019
|  -- conf.lmp
|  -- eos.json
|  -- frozen_model.pb -> ../frozen_model.pb
|  -- in.lammps
|  -- inter.json
|  -- POSCAR
|  -- POSCAR.orig -> ../../relaxation/relax_task/CONTCAR
|-- task.json
```

eos.json records the volume and scale of the corresponding task.

Elastic output:

```
confs/std-fcc/elastic_00/
|-- equi.stress.json
|-- frozen_model.pb -> ../../../frozen_model.pb
|-- in.lammps
|-- POSCAR -> ../relaxation/relax_task/CONTCAR
|-- task.000000
|  -- conf.lmp
|  -- frozen_model.pb -> ../frozen_model.pb
|  -- in.lammps -> ../in.lammps
|  -- inter.json
|  -- POSCAR
|  -- strain.json
|-- task.json
|-- task.000001
|  -- conf.lmp
|  -- frozen_model.pb -> ../frozen_model.pb
|  -- in.lammps -> ../in.lammps
|  -- inter.json
|  -- POSCAR
|  -- strain.json
|-- task.json
...
|-- task.000023
|  -- conf.lmp
|  -- frozen_model.pb -> ../frozen_model.pb
|  -- in.lammps -> ../in.lammps
|  -- inter.json
|  -- POSCAR
|  -- strain.json
|-- task.json
```

equi.stress.json records the stress information of the equilibrium task and strain.json records the deformation information of the corresponding task.

Vacancy output:

```
confs/std-fcc/vacancy_00/
```

(continues on next page)

(continued from previous page)

```

|-- frozen_model.pb -> ../../../../frozen_model.pb
|-- in.lammps
|-- POSCAR -> ../relaxation/relax_task/CONTCAR
|-- task.000000
|   |-- conf.lmp
|   |-- frozen_model.pb -> ../frozen_model.pb
|   |-- in.lammps -> ../in.lammps
|   |-- inter.json
|   |-- POSCAR
|   |-- supercell.json
|-- task.json

```

supercell.json records the supercell size information of the corresponding task.

Interstitial output:

```

confs/std-fcc/interstitial_00/
|-- element.out
|-- frozen_model.pb -> ../../../../frozen_model.pb
|-- in.lammps
|-- POSCAR -> ../relaxation/relax_task/CONTCAR
|-- task.000000
|   |-- conf.lmp
|   |-- frozen_model.pb -> ../frozen_model.pb
|   |-- in.lammps -> ../in.lammps
|   |-- inter.json
|   |-- POSCAR
|   |-- supercell.json
|-- task.json
|-- task.000001
|   |-- conf.lmp
|   |-- frozen_model.pb -> ../frozen_model.pb
|   |-- in.lammps -> ../in.lammps
|   |-- inter.json
|   |-- POSCAR
|   |-- supercell.json
|-- task.json

```

element.out records the inserted element type of each task and supercell.json records the supercell size information of the corresponding task.

Surface output:

```

confs/std-fcc/surface_00/
|-- frozen_model.pb -> ../../../../frozen_model.pb
|-- in.lammps
|-- POSCAR -> ../relaxation/relax_task/CONTCAR
|-- task.000000
|   |-- conf.lmp
|   |-- frozen_model.pb -> ../frozen_model.pb
|   |-- in.lammps -> ../in.lammps
|   |-- inter.json
|   |-- miller.json

```

(continues on next page)

(continued from previous page)

```

| | -- POSCAR
| | -- POSCAR.tmp
| | -- task.json
|-- task.000001
| | -- conf.lmp
| | -- frozen_model.pb -> ../frozen_model.pb
| | -- in.lammps -> ../in.lammps
| | -- inter.json
| | -- miller.json
| | -- POSCAR
| | -- POSCAR.tmp
| | -- task.json
...
|-- task.000008
| | -- conf.lmp
| | -- frozen_model.pb -> ../frozen_model.pb
| | -- in.lammps -> ../in.lammps
| | -- inter.json
| | -- miller.json
| | -- POSCAR
| | -- POSCAR.tmp
| | -- task.json

```

miller.json records the miller index of the corresponding task.

7.4.3 Property run

```
nohup dpngen autotest run property.json machine-ali.json > run.result 2>&1 &
```

the result file log.lammps, dump.relax, and outlog would be sent back.

7.4.4 Property-post

Use command

```
dpngen autotest post property.json
```

to post results as result.json and result.out in each property's path.

7.4.5 Properties

EOS get started and input examples

Equation of State (EOS) here calculates the energies of the most stable structures as a function of volume. Users can refer to Figure 4 of the [dpngen CPC paper](#) for more information of EOS.

An example of the input file for EOS by VASP:

```
{
  "structures":      ["confs/mp-*", "confs/std-*", "confs/test-*"],
  "interaction": {
    "type":          "vasp",
    "incar":         "vasp_input/INCAR",
    "potcar_prefix": "vasp_input",
    "potcars":       {"Al": "POTCAR.al", "Mg": "POTCAR.mg"}
  },
  "properties": [
    {
      "type":         "eos",
      "vol_start":    0.9,
      "vol_end":      1.1,
      "vol_step":     0.01
    }
  ]
}
```

vol_start is the starting volume relative to the equilibrium structure, vol_step is the volume increment step relative to the equilibrium structure, and the biggest relative volume is smaller than vol_end.

EOS make

Step 1. Before make in EOS, the equilibrium configuration CONTCAR must be present in confs/mp-*/relaxation.

Step 2. For the input example in the previous section, when we do make, 40 tasks would be generated as confs/mp-*/eos_00/task.000000, confs/mp-*/eos_00/task.000001, ... , confs/mp-*/eos_00/task.000039. The suffix 00 is used for possible refine later.

Step 3. If the task directory, for example confs/mp-*/eos_00/task.000000 is not empty, the old input files in it including INCAR, POSCAR, POTCAR, conf.lmp, in.lammps would be deleted.

Step 4. In each task directory, POSCAR.orig would link to confs/mp-*/relaxation/CONTCAR. Then the scale parameter can be calculated as:

```
scale = (vol_current / vol_equi) ** (1. / 3.)
```

vol_current is the corresponding volume per atom of the current task and vol_equi is the volume per atom of the equilibrium configuration. Then the poscar_scale function in dpgen.auto_test.lib.vasp module would help to generate POSCAR file with vol_current in confs/mp-*/eos_00/task.[0-9]*[0-9].

Step 5. According to the task type, the input file including INCAR, POTCAR or conf.lmp, in.lammps would be written in every confs/mp-*/eos_00/task.[0-9]*[0-9].

EOS run

The work path of each task should be in the form like `confs/mp-*/eos_00` and all task is in the form like `confs/mp-*/eos_00/task.[0-9]*[0-9]`.

When we dispatch tasks, we would go through every individual work path in the list `confs/mp-*/eos_00`, and then submit `task.[0-9]*[0-9]` in each work path.

EOS post

The post processing of EOS would go to every directory in `confs/mp-*/eos_00` and do the post processing. Let's suppose we are now in `confs/mp-100/eos_00` and there are `task.000000`, `task.000001`, ..., `task.000039` in this directory. By reading `inter.json` file in every task directory, the task type can be determined and the energy and force information of every task can further be obtained. By appending the dict of energy and force into a list, an example of the list with 1 atom is given as:

```
[
  {"energy": E1, "force": [fx1, fy1, fz1]},
  {"energy": E2, "force": [fx2, fy2, fz2]},
  ...
  {"energy": E40, "force": [fx40, fy40, fz40]}
]
```

Then the volume can be calculated from the task id and the corresponding energy can be obtained from the list above. Finally, there would be `result.json` in json format and `result.out` in txt format in `confs/mp-100/eos_00` containing the EOS results.

An example of `result.json` is give as:

```
{
  "14.808453313267595": -3.7194474,
  "14.972991683415014": -3.7242038,
  ...
  "17.934682346068534": -3.7087655
}
```

An example of `result.out` is given below:

```
onf_dir: /root/auto_test_example/deepmd/confs/std-fcc/eos_00
VpA(A^3)  EpA(eV)
14.808    -3.7194
14.973    -3.7242
...
17.935    -3.7088
```


Elastic get started and input examples

Here we calculate the mechanical properties which include elastic constants (C11 to C66), bulk modulus B_v , shear modulus G_v , Youngs modulus E_v , and Poission ratio U_v of a certain crystal structure.

An example of the input file for Elastic by deepmd:

```
{
  "structures":      ["confs/mp-*", "confs/std-*", "confs/test-*"],
  "interaction": {
    "type": "deepmd",
    "model": "frozen_model.pb",
    "type_map":      {"Al": 0, "Mg": 1}
  },
  "properties": [
    {
      "type": "elastic",
      "norm_deform": 1e-2,
      "shear_deform": 1e-2
    }
  ]
}
```

Here the default values of `norm_deform` and `shear_deform` are **1e-2** and **1e-2**, respectively. A list of `norm_strains` and `shear_strains` would be generated as below:

```
[-norm_def, -0.5 * norm_def, 0.5 * norm_def, norm_def]
[-shear_def, -0.5 * shear_def, 0.5 * shear_def, shear_def]
```

Elastic make

Step 1. The `DeformedStructureSet` module in `pymatgen.analysis.elasticity.strain` is used to generate a set of independently deformed structures. `equi.stress.out` file is written to record the equilibrium stress in the Elastic directory. For the example in the previous section, `equi.stress.out` should be in `confs/mp-*/elastic_00`.

Step 2. If there are `init_from_suffix` and `output_suffix` parameter in the `properties` part, the *refine process* follows. Else, the deformed structure (POSCAR) and strain information (`strain.out`) are written in the task directory, for example, in `confs/mp-*/elastic_00/task.0000000`.

Step 3. When doing elastic by VASP, `ISIF=2`. When doing by LAMMPS, the following `in.lammps` would be written.

```
units          metal
dimension      3
boundary       p      p      p
atom_style     atomic
box            tilt large
read_data      conf.lmp
mass           1 1
mass           2 1
neigh_modify   every 1 delay 0 check no
pair_style     deepmd frozen_model.pb
```

(continues on next page)

(continued from previous page)

```

pair_coeff
compute      mype all pe
thermo        100
thermo_style  custom step pe pxx pyy pzz pxy pxz pyz lx ly lz vol c_mype
dump          1 all custom 100 dump.relax id type xs ys zs fx fy fz
min_style     cg
minimize      0 1.0000000e-10 5000 500000
variable      N equal count(all)
variable      V equal vol
variable      E equal "c_mype"
variable      Pxx equal pxx
variable      Pyy equal pyy
variable      Pzz equal pzz
variable      Pxy equal pxy
variable      Pxz equal pxz
variable      Pyz equal pyz
variable      Epa equal ${E}/${N}
variable      Vpa equal ${V}/${N}
print "All done"
print "Total number of atoms = ${N}"
print "Final energy per atoms = ${Epa}"
print "Final volume per atoms = ${Vpa}"
print "Final Stress (xx yy zz xy xz yz) = ${Pxx} ${Pyy} ${Pzz} ${Pxy} ${Pxz} ${Pyz}"

```

Elastic run

Very similar to the run operation of EOS except for in different directories. Now the work path of each task should be in the form like `confs/mp-*/elastic_00` and all task is in the form like `confs/mp-*/elastic_00/task.[0-9]*[0-9]`.

Elastic post

The ElasticTensor module in `pymatgen.analysis.elasticity.elastic` is used to get the elastic tensor, Bv, and Gv. The mechanical properties of a crystal structure would be written in `result.json` in json format and `result.out` in txt format. The example of the output file is give below.

result.json

```

{
  "elastic_tensor": [
    134.90955999999997,
    54.329958699999985,
    51.802386099999985,
    3.5745279599999993,
    -1.38863259999999648e-05,
    -1.96382339999999486e-05,
    54.558402999999999,
    134.59654699999996,
    51.7972336,

```

(continues on next page)

(continued from previous page)

```

-3.53972684,
1.8395687999999963e-05,
8.7567993999999951e-05,
51.913248599999999,
51.913292199999994,
137.01763799999998,
-5.0903393999999969e-05,
6.992516299999996e-05,
3.7364786999999946e-05,
3.8780564440000007,
-3.770445632,
-1.2766205999999956,
35.413431999999999,
2.2479590800000023e-05,
1.3837692000000017e-06,
-4.959999999495933e-06,
2.5800000003918792e-06,
1.4800000030874965e-06,
2.9000000008417968e-06,
35.375960199999994,
3.8608356,
0.0,
0.0,
0.0,
0.0,
4.02554856,
38.375018399999995
],
"BV": 80.315363022222,
"GV": 38.40582656,
"EV": 99.37716395728943,
"uV": 0.2937771799031088
}

```

The order of `elastic_tensor` is C11, C12, ..., C16, C21, C22, ..., C26, ..., C66 and the unit of Bv, Gv, Ev, and uv is GPa.

result.out

```

/root/auto_test_example/deepmd/confs/std-fcc/elastic_00
134.91  54.33  51.80  3.57  -0.00  -0.00
54.56  134.60  51.80  -3.54  0.00  0.00
51.91  51.91  137.02  -0.00  0.00  0.00
3.88   -3.77  -1.28  35.41  0.00  0.00
-0.00  0.00  0.00  0.00  35.38  3.86
0.00  0.00  0.00  0.00  4.03  38.38
# Bulk   Modulus BV = 80.32 GPa
# Shear  Modulus GV = 38.41 GPa
# Youngs Modulus EV = 99.38 GPa
# Poission Ratio uV = 0.29

```

Vacancy get started and input examples

Vacancy calculates the energy difference when removing an atom from the crystal structure. We only need to give the information of `supercell` to help calculate the vacancy energy and the default value of `supercell` is `[1, 1, 1]`.

An example of the input file for Vacancy by deepmd:

```
{
  "structures":      "confs/mp-*",
  "interaction": {
    "type":          "deepmd",
    "model":          "frozen_model.pb",
    "type_map":       {"Al": 0, "Mg": 1}
  },
  "properties": [
    {
      "type":         "vacancy",
      "supercell":     [1, 1, 1]
    }
  ]
}
```

Vacancy make

Step 1. The `VacancyGenerator` module in `pymatgen.analysis.defects.generators` is used to generate a set of structures with vacancy.

Step 2. If there are `init_from_suffix` and `output_suffix` parameter in the `properties` part, the *refine process* follows. If `reproduce` is evoked, the *reproduce process* follows. Otherwise, the vacancy structure (POSCAR) and supercell information (`supercell.out`) are written in the task directory, for example, in `confs/mp-*/vacancy_00/task.000000` with the check and possible removing of the old input files like before.

Step 3. When doing vacancy by VASP, `ISIF = 3`. When doing vacancy by LAMMPS, the same `in.lammps` as that in *EOS* (`change_box is True`) would be generated with `scale` set to one.

Vacancy run

Very similar to the run operation of EOS except for in different directories. Now the work path of each task should be in the form like `confs/mp-*/vacancy_00` and all task is in the form like `confs/mp-*/vacancy_00/task.[0-9]*[0-9]`.

Vacancy post

For Vacancy, we need to calculate the energy difference between a crystal structure with and without a vacancy. The examples of the output files `result.json` in json format and `result.out` in txt format are given below.

result.json

```
{
  "[3, 3, 3]-task.000000": [
    0.7352769999999964,
    -96.644642,
    -97.379919
  ]
}
```

result.out

```
/root/auto_test_example/deepmd/confs/std-fcc/vacancy_00
Structure:      Vac_E(eV)  E(eV)  equi_E(eV)
[3, 3, 3]-task.000000:  0.735  -96.645  -97.380
```

Interstitial get started and input examples

Interstitial calculates the energy difference when adding an atom into the crystal structure. We need to give the information of supercell (default value is [1, 1, 1]) and insert_ele list for the element types of the atoms added in.

An example of the input file for Interstitial by deepmd:

```
{
  "structures":      "confs/mp-*",
  "interaction": {
    "type":          "deepmd",
    "model":          "frozen_model.pb",
    "type_map":       {"Al": 0, "Mg": 1}
  },
  "properties": [
    {
      "type":         "interstitial",
      "supercell":    [3, 3, 3],
      "insert_ele":   ["Al"],
      "conf_filters": {"min_dist": 1.5},
      "cal_setting":  {"input_prop": "lammmps_input/lammmps_high"}
    }
  ]
}
```

We add a `conf_filters` parameter in `properties` part and this parameter can help to eliminate undesirable structure which can render rather difficult convergence in calculations. In the example above, “**min_dist**”: 1.5 means if the smallest atomic distance in the structure is less than 1.5 angstrom, the configuration would be eliminated and not used in calculations.

Interstitial make

Step 1. For each element in `insert_ele` list, `InterstitialGenerator` module in `pymatgen.analysis.defects.generators` would help to generate interstitial structure. The structure would be appended into a list if it can meet the requirements in `conf_filters`.

Step 2. If `refine` is True, we do *refine process*. If `reprod-opt` is True (the default is **False**), we do *reproduce process*. Else, the vacancy structure (POSCAR) and supercell information (`supercell.out`) are written in the task directory, for example, in `confs/mp-*/interstitial_00/task.0000000` with the check and possible removing of the old input files like before.

Step 3. In interstitial by VASP, `ISIF = 3`. In interstitial by LAMMPS, the same `in.lammps` as that in *EOS (change_box is True)* would be generated with `scale` set to one.

Interstitial run

Very similar to the run operation of EOS except for in different directories. Now the work path of each task should be in the form like `confs/mp-*/interstitial_00` and all task is in the form like `confs/mp-*/interstitial_00/task.[0-9]*[0-9]`.

Interstitial post

For Interstitial, we need to calculate the energy difference between a crystal structure with and without atom added in. The examples of the output files `result.json` in json format and `result.out` in txt format are given below.

result.json

```
{
  "Al-[3, 3, 3]-task.0000000": [
    4.0229520000000004,
    -100.84773,
    -104.870682
  ],
  "Al-[3, 3, 3]-task.0000001": [
    2.78295200000000088,
    -102.08773,
    -104.870682
  ]
}
```

result.out

```
/root/auto_test_example/deepmd/confs/std-fcc/interstitial_00
Insert_ele-Struct: Inter_E(eV)  E(eV)  equi_E(eV)
Al-[3, 3, 3]-task.0000000:   4.023  -100.848 -104.871
Al-[3, 3, 3]-task.0000001:   2.783  -102.088 -104.871
```

Surface get started and input examples

Surface calculates the surface energy. We need to give the information of `min_slab_size`, `min_vacuum_size`, `max_miller` (default value is 2), and `pert_xz` which means perturbations in xz and will help work around vasp bug.

An example of the input file for Surface by deepmd:

```
{
  "structures":      "confs/mp-*",
  "interaction": {
    "type":          "deepmd",
    "model":          "frozen_model.pb",
    "type_map":      {"Al": 0, "Mg": 1}
  },
  "properties": [
    {
      "type":          "surface",
      "min_slab_size": 10,
      "min_vacuum_size": 11,
      "max_miller":    2,
      "cal_type":      "static"
    }
  ]
}
```

Surface make

Step 1. Based on the equilibrium configuration, `generate_all_slabs` module in `pymatgen.core.surface` would help to generate surface structure list with using `max_miller`, `min_slab_size`, and `min_vacuum_size` parameters.

Step 2. If `refine` is True, we do *refine process*. If `reprod-opt` is True (the default is False), we do *reproduce process*. Otherwise, the surface structure (POSCAR) with perturbations in xz and miller index information (`miller.out`) are written in the task directory, for example, in `confs/mp-*/interstitial_00/task.000000` with the check and possible removing of the old input files like before.

Surface run

Very similar to the run operation of EOS except for in different directories. Now the work path of each task should be in the form like `confs/mp-*/surface_00` and all task is in the form like `confs/mp-*/surface_00/task.[0-9]*[0-9]`.

Surface post

For Surface, we need to calculate the energy difference between a crystal structure with and without a surface with a certain miller index divided by the surface area.

The examples of the output files `result.json` in json format and `result.out` in txt format are given below.

result.json

```
{
  "[1, 1, 1]-task.000000": [
    0.8051037974207992,
    -3.6035018,
    -3.7453815
  ],
  "[2, 2, 1]-task.000001": [
    0.9913881928811771,
    -3.5781115999999997,
    -3.7453815
  ],
  "[1, 1, 0]-task.000002": [
    0.9457333586026173,
    -3.5529366000000002,
    -3.7453815
  ],
  "[2, 2, -1]-task.000003": [
    0.9868013100872397,
    -3.5590607142857142,
    -3.7453815
  ],
  "[2, 1, 1]-task.000004": [
    1.0138239046484236,
    -3.563035875,
    -3.7453815
  ],
  "[2, 1, -1]-task.000005": [
    1.0661817319108005,
    -3.5432459166666668,
    -3.7453815
  ],
  "[2, 1, -2]-task.000006": [
    1.034003253044026,
    -3.550884125,
    -3.7453815
  ],
  "[2, 0, -1]-task.000007": [
```

(continues on next page)

(continued from previous page)

```

    0.9569958287615818,
    -3.5685403333333334,
    -3.7453815
  ],
  "[2, -1, -1]-task.000008": [
    0.9432935501134583,
    -3.5774615714285716,
    -3.7453815
  ]
}

```

result.out

```

/root/auto_test_example/deepmd/confs/std-fcc/surface_00
Miller_Indices:      Surf_E(J/m^2)  EpA(eV)  equi_EpA(eV)
[1, 1, 1]-task.000000:      0.805      -3.604   -3.745
[2, 2, 1]-task.000001:      0.991      -3.578   -3.745
[1, 1, 0]-task.000002:      0.946      -3.553   -3.745
[2, 2, -1]-task.000003:     0.987      -3.559   -3.745
[2, 1, 1]-task.000004:      1.014      -3.563   -3.745
[2, 1, -1]-task.000005:     1.066      -3.543   -3.745
[2, 1, -2]-task.000006:     1.034      -3.551   -3.745
[2, 0, -1]-task.000007:     0.957      -3.569   -3.745
[2, -1, -1]-task.000008:     0.943      -3.577   -3.745

```

7.5 Refine

7.5.1 Refine get started and input examples

Sometimes we want to refine the calculation of a property from previous results. For example, when higher convergence criteria EDIFF and EDIFFG are necessary in VASP, the new VASP calculation is desired to start from the previous output configuration, rather than starting from scratch.

An example of the input file `refine.json` is given below:

```

{
  "structures":      ["confs/std-*"],
  "interaction": {
    "type":          "deepmd",
    "model":          "frozen_model.pb",
    "type_map":      {"Al": 0}
  },
  "properties": [
    {
      "type":          "vacancy",
      "init_from_suffix": "00",
      "output_suffix":  "01",
      "cal_setting":    {"input_prop": "lammps_input/lammps_high"}
    }
  ]
}

```

(continues on next page)

(continued from previous page)

```

    }
  ]
}

```

In this example, `refine` would output the results to `vacancy_01` based on the previous results in `vacancy_00` by using a different input commands file for `lammps`.

7.5.2 Refine make

```

dpgen autotest make refine.json
tree confs/std-fcc/vacancy_01/

```

the output will be:

```

confs/std-fcc/vacancy_01/
|-- frozen_model.pb -> ../../../../frozen_model.pb
|-- in.lammps
|-- task.000000
|   |-- conf.lmp
|   |-- frozen_model.pb -> ../frozen_model.pb
|   |-- in.lammps -> ../in.lammps
|   |-- inter.json
|   |-- POSCAR -> ../../vacancy_00/task.000000/CONTCAR
|   |-- supercell.json -> ../../vacancy_00/task.000000/supercell.json
|-- task.json

```

an new directory `vacancy_01` would be established and the starting configuration links to previous results.

7.5.3 Refine run

```

nohup dpgen autotest run refine.json machine-ali.json > run.result 2>&1 &

```

the run process of `refine` is similar to before.

7.5.4 Refine post

```

dpgen autotest post refine.json

```

the post process of `refine` is similar to the corresponding property.

7.6 Reproduce

7.6.1 Reproduce get started and input examples

Sometimes we want to reproduce the initial results with the same configurations for cross validation. This version of autotest package can accomplish this successfully in all property types except for Elastic. An input example for using deepmd to reproduce the VASP Interstitial results is given below:

```
{
  "structures":      ["confs/std-*"],
  "interaction": {
    "type":          "deepmd",
    "model":          "frozen_model.pb",
    "type_map":       {"Al": 0}
  },
  "properties": [
    {
      "type":          "interstitial",
      "reproduce":       true,
      "init_from_suffix": "00",
      "init_data_path": "../vasp/confs",
      "reprod_last_frame": false
    }
  ]
}
```

reproduce denotes whether to do reproduce or not and the default value is False.

init_data_path is the path of VASP or LAMMPS initial data to be reproduced. init_from_suffix is the suffix of the initial data and the default value is "00". In this case, the VASP Interstitial results are stored in ../vasp/confs/std-*/interstitial_00 and the reproduced Interstitial results would be in deepmd/confs/std-*/interstitial_reprod.

reprod_last_frame denotes if only the last frame is used in reproduce. The default value is True for eos and surface, but is False for vacancy and interstitial.

7.6.2 Reproduce make

```
dpngen autotest make reproduce.json
tree confs/std-fcc/interstitial_reprod/
```

the output will be:

```
confs/std-fcc/interstitial_reprod/
|-- frozen_model.pb -> ../../../../frozen_model.pb
|-- in.lammps
|-- task.0000000
|  |-- conf.lmp
|  |-- frozen_model.pb -> ../frozen_model.pb
|  |-- in.lammps -> ../in.lammps
|  |-- inter.json
|  |-- POSCAR
```

(continues on next page)

(continued from previous page)

```

|  -- task.json
|-- task.000001
|  -- conf.lmp
|  -- frozen_model.pb -> ../frozen_model.pb
|  -- in.lammps -> ../in.lammps
|  -- inter.json
|  -- POSCAR
|  -- task.json
...
|-- task.000038
|  -- conf.lmp
|  -- frozen_model.pb -> ../frozen_model.pb
|  -- in.lammps -> ../in.lammps
|  -- inter.json
|  -- POSCAR
|-- task.json

```

every single frame in the initial data is split into each task and the following `in.lammps` would help to do the static calculation:

```

clear
units          metal
dimension      3
boundary       p p p
atom_style     atomic
box            tilt large
read_data      conf.lmp
mass           1 26.982
neigh_modify   every 1 delay 0 check no
pair_style     deepmd frozen_model.pb
pair_coeff
compute        mype all pe
thermo         100
thermo_style   custom step pe pxx pyy pzz pxy pxz pyz lx ly lz vol c_mype
dump           1 all custom 100 dump.relax id type xs ys zs fx fy fz
run            0
variable       N equal count(all)
variable       V equal vol
variable       E equal "c_mype"
variable       tmp1x equal lx
variable       tmp1y equal ly
variable       Pxx equal pxx
variable       Pyy equal pyy
variable       Pzz equal pzz
variable       Pxy equal pxy
variable       Pxz equal pxz
variable       Pyz equal pyz
variable       Epa equal ${E}/${N}
variable       Vpa equal ${V}/${N}
variable       AA equal (${tmp1x}*${tmp1y})
print "All done"
print "Total number of atoms = ${N}"

```

(continues on next page)

(continued from previous page)

```

print "Final energy per atoms = ${Epa}"
print "Final volume per atoms = ${Vpa}"
print "Final Base area = ${AA}"
print "Final Stress (xx yy zz xy xz yz) = ${Pxx} ${Pyy} ${Pzz} ${Pxy} ${Pxz} ${Pyz}"

```

7.6.3 Reproduce run

```
nohup dpngen autotest run reproduce.json machine-ali.json > run.result 2>&1 &
```

the run process of reproduce is similar to before.

7.6.4 Reproduce post

```
dpngen autotest post reproduce.json
```

the output will be:

result.out:

```

/root/auto_test_example/deepmd/confs/std-fcc/interstitial_reprod
Reproduce: Initial_path Init_E(eV/atom)  Reprod_E(eV/atom)  Difference(eV/atom)
.../vasp/confs/std-fcc/interstitial_00/task.000000 -3.020 -3.240 -0.220
.../vasp/confs/std-fcc/interstitial_00/task.000000 -3.539 -3.541 -0.002
.../vasp/confs/std-fcc/interstitial_00/task.000000 -3.582 -3.582 -0.001
.../vasp/confs/std-fcc/interstitial_00/task.000000 -3.582 -3.581 0.001
.../vasp/confs/std-fcc/interstitial_00/task.000000 -3.594 -3.593 0.001
.../vasp/confs/std-fcc/interstitial_00/task.000000 -3.594 -3.594 0.001
.../vasp/confs/std-fcc/interstitial_00/task.000000 -3.598 -3.597 0.001
.../vasp/confs/std-fcc/interstitial_00/task.000000 -3.600 -3.600 0.001
.../vasp/confs/std-fcc/interstitial_00/task.000000 -3.600 -3.600 0.001
.../vasp/confs/std-fcc/interstitial_00/task.000000 -3.601 -3.600 0.001
.../vasp/confs/std-fcc/interstitial_00/task.000000 -3.602 -3.601 0.001
.../vasp/confs/std-fcc/interstitial_00/task.000000 -3.603 -3.602 0.001
.../vasp/confs/std-fcc/interstitial_00/task.000000 -3.603 -3.602 0.001
.../vasp/confs/std-fcc/interstitial_00/task.000000 -3.603 -3.602 0.001
.../vasp/confs/std-fcc/interstitial_00/task.000000 -3.603 -3.602 0.001
.../vasp/confs/std-fcc/interstitial_00/task.000000 -3.603 -3.602 0.001
.../vasp/confs/std-fcc/interstitial_00/task.000000 -3.603 -3.602 0.001
.../vasp/confs/std-fcc/interstitial_00/task.000001 -3.345 -3.372 -0.027
.../vasp/confs/std-fcc/interstitial_00/task.000001 -3.546 -3.556 -0.009
.../vasp/confs/std-fcc/interstitial_00/task.000001 -3.587 -3.593 -0.007
.../vasp/confs/std-fcc/interstitial_00/task.000001 -3.593 -3.599 -0.006
.../vasp/confs/std-fcc/interstitial_00/task.000001 -3.600 -3.606 -0.006
.../vasp/confs/std-fcc/interstitial_00/task.000001 -3.600 -3.606 -0.006
.../vasp/confs/std-fcc/interstitial_00/task.000001 -3.624 -3.631 -0.006
.../vasp/confs/std-fcc/interstitial_00/task.000001 -3.634 -3.640 -0.007
.../vasp/confs/std-fcc/interstitial_00/task.000001 -3.637 -3.644 -0.007
.../vasp/confs/std-fcc/interstitial_00/task.000001 -3.637 -3.644 -0.007
.../vasp/confs/std-fcc/interstitial_00/task.000001 -3.638 -3.645 -0.007

```

(continues on next page)

(continued from previous page)

```

.../vasp/confs/std-fcc/interstitial_00/task.000001 -3.638 -3.645 -0.007
.../vasp/confs/std-fcc/interstitial_00/task.000001 -3.639 -3.646 -0.007
.../vasp/confs/std-fcc/interstitial_00/task.000001 -3.639 -3.646 -0.007
.../vasp/confs/std-fcc/interstitial_00/task.000001 -3.639 -3.646 -0.007
.../vasp/confs/std-fcc/interstitial_00/task.000001 -3.639 -3.646 -0.007
.../vasp/confs/std-fcc/interstitial_00/task.000001 -3.639 -3.646 -0.007
.../vasp/confs/std-fcc/interstitial_00/task.000001 -3.639 -3.646 -0.007
.../vasp/confs/std-fcc/interstitial_00/task.000001 -3.639 -3.646 -0.007
.../vasp/confs/std-fcc/interstitial_00/task.000001 -3.639 -3.646 -0.007
.../vasp/confs/std-fcc/interstitial_00/task.000001 -3.639 -3.646 -0.007

```

the comparison of the initial and reproduced results as well as the absolute path of the initial data is recorded.

result.json:

```

{
  "/root/auto_test_example/vasp/confs/std-fcc/interstitial_00/task.000000": {
    "nframes": 18,
    "error": 0.0009738182472213228
  },
  "/root/auto_test_example/vasp/confs/std-fcc/interstitial_00/task.000001": {
    "nframes": 21,
    "error": 0.0006417039154057605
  }
}

```

the error analysis corresponding to the initial data is recorded and the error of the first frame is disregarded when all the frames are considered in reproduce.

USER GUIDE

This part aims to show you how to get the community's help. Some frequently asked questions are listed in troubleshooting, and the explanation of errors that often occur is listed in common errors. If other unexpected problems occur, you're welcome to contact us for help.

8.1 Discussions

Welcome everyone to participate in the discussion about DP-GEN in the [discussion](#) module. You can ask for help, share an idea or anything to discuss here. Note: before you raise a question, please check TUTORIAL/FAQs and search history discussions to find solutions.

8.2 Issue

If you want to make a bug report or a request for new features, you can make an issue in the issue module.

Here are the types you can choose. A proper type can help developer figure out what you need. Also, you can assign yourself to solve the issue. Your contribution is welcome!

Note: before you raise a question, please check TUTORIAL/FAQs and search history issues to find solutions.

8.3 Tutorials

Tutorials can be found [here](#).

8.4 Example for parameters

If you have no idea how to prepare a PARAM for your task, you can find examples of PARAM for different tasks in [examples](#).

For example, if you want to set specific template for LAMMPS, you can find an example [here](#)

If you want to learn more about Machine parameters, please check [docs for dpdispatcher](#)

8.5 Pull requests - How to contribute

8.6 Troubleshooting

1. The most common problem is whether two settings correspond with each other, including:
 - The order of elements in `type_map` and `mass_map` and **`fp_pp_files`**.
 - Size of `init_data_sys` and `init_batch_size`.
 - Size of `sys_configs` and `sys_batch_size`.
 - Size of `sel_a` and actual types of atoms in your system.
 - Index of `sys_configs` and `sys_idx`.
2. Please verify the directories of `sys_configs`. If there isn't any POSCAR for `01.model_devi` in one iteration, it may happen that you write the false path of `sys_configs`. Note that `init_data_sys` is a list, while `sys_configs` should be a two-dimensional list. The first dimension corresponds to `sys_idx`, and the second level are some poscars under each group. Refer to the [sample file](#).
3. Correct format of JSON file.
4. The frames of one system should be larger than `batch_size` and `numb_test` in `default_training_param`. It happens that one iteration adds only a few structures and causes error in next iteration's training. In this condition, you may let `fp_task_min` be larger than `numb_test`.
5. If you found the dpgen with the same version on two machines behaves differently, you may have modified the code in one of them.

8.7 Common Errors

(Errors are sorted alphabetically)

8.7.1 Command not found: xxx.

There is no such software in the environment, or it is unavailable. It may be because 1. It is not installed; 2. The Conda environment is not activated; 3. You have chosen the wrong image in machine.json.

8.7.2 `dargs.dargs.ArgumentKeyError: [at location xxx] undefined key xxx is not allowed in strict mode.`

Strict format check has been applied since version 0.10.7. To avoid misleading users, some older-version keys that are already ignored or absorbed into default settings are not allowed to be present. And the expected structure of the dictionary in the param.json also differs from those before version 0.10.7. This error will occur when format check finds older-fashion keys in the json file. Please try deleting or annotating these keys, or correspondingly modulate the json file. Example files in the newest format could be found in [examples](#).

8.7.3 dargs.dargs.ArgumentTypeError: [at root location] key xxx gets wrong value type, requires but gets

Please check your parameters with [DPGEN's Document](#). Maybe you have superfluous parentheses in your parameter file.

8.7.4 FileNotFoundError: [Errno 2] No such file or directory: './.../01.model_devi/graph.xxx.pb'

If you find this error occurs, please check your initial data. Your model will not be generated if the initial data is incorrect.

8.7.5 json.decoder.JSONDecodeError

Your .json file is incorrect. It may be a mistake in syntax or a missing comma.

8.7.6 OSError: [Error cannot find valid a data system] Please check your setting for data systems

Check if the path to the dataset in the parameter file is set correctly. Note that `init_data_sys` is a list, while `sys_configs` should be a two-dimensional list. The first dimension corresponds to `sys_idx`, and the second level are some poscars under each group. Refer to the [sample file](#).

8.7.7 RuntimeError: job:xxxxxxx failed 3 times

```
RuntimeError: job:xxxxxxx failed 3 times

.....

RuntimeError: Meet errors will handle unexpected submission state.
Debug information: remote_root==xxxxxxx
Debug information: submission_hash==xxxxxxx
Please check the dirs and scripts in remote_root. The job information mentioned above,
↪may help.
```

If a user finds an error like this, he or she is advised to check the files on the remote server. It shows that your job has failed 3 times, but has not shown the reason.

To find the reason, you can check the log on the remote root. For example, you can check `train.log`, which is generated by DeepMD-kit. It can tell you more details. If it doesn't help, you can manually run the `.sub` script, whose path is shown in Debug information: `remote_root==xxxxxxx`

Some common reasons are as follows:

1. Two or more jobs are submitted manually or automatically at the same time, and their hash value collide. This bug will be fixed in `dpdispatcher`.
2. You may have something wrong in your input files, which causes the process to fail.

8.7.8 RuntimeError: find too many unsuccessfully terminated jobs.

The ratio of failed jobs is larger than `ratio_failure`. You can set a high value for `ratio_failure` or check if there is something wrong with your input files.

8.7.9 ValueError: Cannot load file containing picked data when `allow_picked=False`

Please ensure that you write the correct path of the dataset with no excess files.

8.7.10 `warnings.warn("Some Gromacs commands were NOT found; "`

You can ignore this warning if you don't need Gromacs. It just show that Gromacs is not installed in you environment.

CONTRIBUTING GUIDE

9.1 Contributing Guide

The way to make contributions is through making pull requests(PR for short). After your PR is merged, the changes you make can be applied by other users.

Firstly, fork in DP-GEN repository. Then you can clone the repository, build a new branch, make changes and then make a pull request.

9.1.1 How to contribute to DP-GEN

Welcome to the repository of [DP-GEN](#)

DP-GEN adopts the same convention as other software in DeepModeling Community.

You can first refer to DeePMD-kit's [Contributing guide](#) and [Developer guide](#).

You can also read relative chapters on [Github Docs](#).

If you have no idea how to fix your problem or where to find the relative source code, please check [Code Structure](#) of the DP-GEN repository on this website.

Use command line

You can use git with the command line, or open the repository on Github Desktop. Here is a video as a demo of making changes to DP-GEN and publishing it with command line.

If you have never used Github before, remember to generate your ssh key and configure the public key in Github Settings. If you can't configure your username and password, please use token. The explanation from Github see [Github Blog: token authentication requirements for git operations](#). A discussion on [StackOverflow](#) can solve this problem.

Use Github Desktop

Also, you can use Github Desktop to make PR. The following shows the steps to clone the repository and add your doc to tutorials. If it is your first time using Github, Open with Github Desktop is recommended. Github Desktop is a software, which can make your operations on branches visually.

After you clone it to your PC, you can open it with Github Desktop.

Firstly, create your new branch based on devel branch.

Secondly, add your doc to the certain directory in your local repository, and add its name into index.

Here is an [example](#). Remember to add the filename of your doc into index!

Thirdly, select the changes that you want to push, and commit to it. Press “Publish branch” to push your origin repository to the remote branch.

Finally, you can check it on github and make a pull request. Press “Compare & pull request” to make a PR.

(Note: please commit pr to the devel branch)

9.1.2 How to contribute to DP-GEN tutorials and documents

Welcome to [the documents of DP-GEN](#)

- If you want to add the documentation of a toy model, simply put your file in the directory doc/toymodels/ and push;
- If you want to add a new directory for a new category of instructions, make a new directory and add it in doc/index.rst.

Also welcome to [Tutorials repository](#) You can find the structure of tutorials and preparations before writing a document in [Writing Tips](#).

The latest page of DP-GEN Docs

Examples of contributions

- [Example 1](#)
- [Example 2](#) (a simple one for beginner)

1. Push your doc

2. Add the directory in index.rst

3. Build and check it

As mentioned in “How to build the website to check if the modification works”.

4. Make pull request to dpngen

9.1.3 Find how a parameter is used in the code

It is strongly recommended that you use the `find in files` function of Visual Studio software, Search function of Visual Studio Code, or similar functions of other software. Type in the name of the parameter you are looking for, and you will see where it is read in and used in the procedure. Of course, you can also search for the relevant code according to the above guide.

9.1.4 Want to modify a function?

If you have special requirements, you can make personalized modifications in the code corresponding to the function. If you think your modification can benefit the public, and it does not conflict with the current DP-GEN function; or if you fix a bug, please make a pull request to contribute the optimization to the DP-GEN repository.

9.1.5 DP-GEN dependencies

`dpdispatcher` and `dpdata` are dependencies of DP-GEN. `dpdispatcher` is related to task submission, monitoring and recovery, and `dpdata` is related to data processing. If you encounter an error and want to find the reason, please judge whether the problem comes from DP-GEN, `dpdispatcher` or `dpdata` according to the last line of `Traceback`.

9.1.6 About the update of the parameter file

You may have noticed that there are `arginfo.py` files in many folders. This is a file used to generate parameter documentation. If you add or modify a parameter in DP-GEN and intend to export it to the main repository, please sync your changes in `arginfo`.

9.1.7 Tips

1. Please try to submit a PR after finishing all the changes
2. Please briefly describe what you do with `git commit -m "<conclude-the-change-you-make>"`! “No description provided.” will make the maintainer feel confused.
3. It is not recommended to make changes directly in the `devel` branch. It is recommended to pull a branch from `devel`: `git checkout -b <new-branch-name>`
4. When switching branches, remember to check if you want to bring the changes to the next branch!
5. Please fix the errors reported by the unit test. You can firstly test on your local machine before pushing commits. Hint: The way to test the code is to go from the main directory to the tests directory, and use the command `python3 -m unittest`. You can watch the demo video for review. Sometimes you may fail unit tests due to your local circumstance. You can check whether the error reported is related to the part you modified to eliminate this problem. After submitting, as long as there is a green check mark after the PR title on the webpage, it means that the test has been passed.
6. Pay attention to whether there are comments under your PR. If there is a change request, you need to check and modify the code. If there are conflicts, you need to solve them manually.

After successfully making a PR, developers will check it and give comments. It will be merged after everything done. Then CONGRATULATIONS! You become a first-time contributor to DP-GEN!

How to get help from the community

10.1 dpgen package

`dpgen.info()`

10.1.1 Subpackages

`dpgen.auto_test` package

Subpackages

`dpgen.auto_test.lib` package

Submodules

`dpgen.auto_test.lib.abacus` module

`dpgen.auto_test.lib.abacus.check_finished(fname)`

`dpgen.auto_test.lib.abacus.check_stru_fixed(struf, fixed)`

`dpgen.auto_test.lib.abacus.final_stru(abacus_path)`

`dpgen.auto_test.lib.abacus.make_kspacing_kpt(struf, kspacing)`

`dpgen.auto_test.lib.abacus.modify_stru_path(strucf, tpath)`

`dpgen.auto_test.lib.abacus.poscar2stru(poscar, inter_param, stru='STRU')`

- `poscar`: POSCAR for input
- `inter_param`: dictionary of 'interaction' from `param.json`
some key words for ABACUS are:
 - `atom_masses`: a dictionary of atoms' masses
 - `orb_files`: a dictionary of orbital files
 - `deepks_desc`: a string of deepks descriptor file
- `stru`: output filename, usually is 'STRU'.

`dpgen.auto_test.lib.abacus.stru2Structure(struf)`

```
dpngen.auto_test.lib.abacus.stru_fix_atom(struf, fix_atom=[True, True, True])
... ATOMIC_POSITIONS Cartesian #Cartesian(Unit is LATTICE_CONSTANT) Si #Name of element 0.0
#Magnetic for this element. 2 #Number of atoms 0.00 0.00 0.00 0 0 0 #x,y,z, move_x, move_y, move_z 0.25
0.25 0.25 0 0 0.

dpngen.auto_test.lib.abacus.stru_scale(stru_in, stru_out, scale)

dpngen.auto_test.lib.abacus.write_input(inputf, inputdict)

dpngen.auto_test.lib.abacus.write_kpt(kptf, kptlist)
```

dpngen.auto_test.lib.crys module

```
dpngen.auto_test.lib.crys.bcc(ele_name='ele', a=3.2144871302356037)

dpngen.auto_test.lib.crys.dhcp(ele_name='ele', a=2.863782463805517, c=9.353074360871936)

dpngen.auto_test.lib.crys.diamond(ele_name='ele', a=2.551340126037118)

dpngen.auto_test.lib.crys.fcc(ele_name='ele', a=4.05)

dpngen.auto_test.lib.crys.fcc1(ele_name='ele', a=4.05)

dpngen.auto_test.lib.crys.hcp(ele_name='ele', a=2.863782463805517, c=4.676537180435968)

dpngen.auto_test.lib.crys.sc(ele_name='ele', a=2.551340126037118)
```

dpngen.auto_test.lib.lammps module

```
dpngen.auto_test.lib.lammps.apply_type_map(conf_file, deepmd_type_map, ptypes)
    Apply type map. conf_file: conf file converted from POSCAR deepmd_type_map: deepmd atom type map
    ptypes: atom types defined in POSCAR.

dpngen.auto_test.lib.lammps.check_finished(fname)

dpngen.auto_test.lib.lammps.check_finished_new(fname, keyword)

dpngen.auto_test.lib.lammps.cvt_lammps_conf(fin, fout, type_map, ofmt='lammps/data')
    Format convert from fin to fout, specify the output format by ofmt Imcomplete situation.

dpngen.auto_test.lib.lammps.element_list(type_map)

dpngen.auto_test.lib.lammps.get_base_area(log)
    Get base area.

dpngen.auto_test.lib.lammps.get_nev(log)
    Get natoms, energy_per_atom and volume_per_atom from lammps log.

dpngen.auto_test.lib.lammps.get_stress(log)
    Get stress from lammps log.

dpngen.auto_test.lib.lammps.inter_deepmd(param)

dpngen.auto_test.lib.lammps.inter_eam_alloy(param)

dpngen.auto_test.lib.lammps.inter_eam_fs(param)
```



```
dpngen.auto_test.lib.lammps.inter_meam(param)
```

```
dpngen.auto_test.lib.lammps.make_lammps_elastic(conf, type_map, interaction, param, etol=0,
                                                ftol=1e-10, maxiter=5000, maxeval=500000)
```

```
dpngen.auto_test.lib.lammps.make_lammps_equi(conf, type_map, interaction, param, etol=0, ftol=1e-10,
                                              maxiter=5000, maxeval=500000, change_box=True)
```

```
dpngen.auto_test.lib.lammps.make_lammps_eval(conf, type_map, interaction, param)
```

```
dpngen.auto_test.lib.lammps.make_lammps_phonon(conf, masses, interaction, param, etol=0, ftol=1e-10,
                                                maxiter=5000, maxeval=500000)
```

Make lammps input for elastic calculation.

```
dpngen.auto_test.lib.lammps.make_lammps_press_relax(conf, type_map, scale2equi, interaction, param,
                                                    B0=70, bp=0, etol=0, ftol=1e-10, maxiter=5000,
                                                    maxeval=500000)
```

```
dpngen.auto_test.lib.lammps.poscar_from_last_dump(dump, poscar_out, deepmd_type_map)
```

Get poscar from the last frame of a lammps MD traj (dump format).

dpngen.auto_test.lib.lmp module

```
dpngen.auto_test.lib.lmp.box2lmpbox(orig, box)
```

```
dpngen.auto_test.lib.lmp.from_system_data(system)
```

```
dpngen.auto_test.lib.lmp.get_atoms(lines)
```

```
dpngen.auto_test.lib.lmp.get_atype(lines)
```

```
dpngen.auto_test.lib.lmp.get_lmpbox(lines)
```

```
dpngen.auto_test.lib.lmp.get_natoms(lines)
```

```
dpngen.auto_test.lib.lmp.get_natoms_vec(lines)
```

```
dpngen.auto_test.lib.lmp.get_natomtypes(lines)
```

```
dpngen.auto_test.lib.lmp.get_posi(lines)
```

```
dpngen.auto_test.lib.lmp.lmpbox2box(lohi, tilt)
```

```
dpngen.auto_test.lib.lmp.system_data(lines)
```

```
dpngen.auto_test.lib.lmp.to_system_data(lines)
```

dpngen.auto_test.lib.mfp_eosfit module

```
dpngen.auto_test.lib.mfp_eosfit.BM4(vol, pars)
```

Birch-Murnaghan 4 pars equation from PRB 70, 224107, 3-order.

```
dpngen.auto_test.lib.mfp_eosfit.BM5(vol, pars)
```

Birch-Murnaghan 5 pars equation from PRB 70, 224107, 4-Order.

`dpngen.auto_test.lib.mfp_eosfit.LOG4(vol, pars)`

Natrual strain (Poirier-Tarantola)EOS with 4 paramters Seems only work in near-equilibrium range.

`dpngen.auto_test.lib.mfp_eosfit.LOG5(vol, parameters)`

Natrual strain (Poirier-Tarantola)EOS with 5 paramters.

`dpngen.auto_test.lib.mfp_eosfit.Li4p(V, parameters)`

Li JH, APL, 87, 194111 (2005).

`dpngen.auto_test.lib.mfp_eosfit.SJX_5p(vol, par)`

SJX_5p's five parameters EOS, Physica B: Condens Mater, 2011, 406: 1276-1282.

`dpngen.auto_test.lib.mfp_eosfit.SJX_v2(vol, par)`

Sun Jiuxun, et al. J phys Chem Solids, 2005, 66: 773-782. They said it is satisfied for the limiting condition at high pressure.

`dpngen.auto_test.lib.mfp_eosfit.TEOS(v, par)`

Holland, et al, Journal of Metamorphic Geology, 2011, 29(3): 333-383 Modified Tait equation of Huang & Chow.

`dpngen.auto_test.lib.mfp_eosfit.birch(v, parameters)`

From Intermetallic compounds: Principles and Practice, Vol. I: Principles Chapter 9 pages 195-210 by M. Mehl. B. Klein, D. Papaconstantopoulos paper downloaded from Web.

case where n=0

`dpngen.auto_test.lib.mfp_eosfit.calc_props_BM4(pars)`

`dpngen.auto_test.lib.mfp_eosfit.calc_props_LOG4(pars)`

`dpngen.auto_test.lib.mfp_eosfit.calc_props_SJX_5p(par)`

`dpngen.auto_test.lib.mfp_eosfit.calc_props_mBM4(pars)`

`dpngen.auto_test.lib.mfp_eosfit.calc_props_mBM4poly(pars)`

`dpngen.auto_test.lib.mfp_eosfit.calc_props_mBM5poly(pars)`

`dpngen.auto_test.lib.mfp_eosfit.calc_props_morse(pars)`

`dpngen.auto_test.lib.mfp_eosfit.calc_props_morse_6p(par)`

`dpngen.auto_test.lib.mfp_eosfit.calc_props_vinet(pars)`

`dpngen.auto_test.lib.mfp_eosfit.calc_v0_mBM4poly(x, pars)`

`dpngen.auto_test.lib.mfp_eosfit.calc_v0_mBM5poly(x, pars)`

`dpngen.auto_test.lib.mfp_eosfit.ext_splint(xp, yp, order=3, method='unispl')`

`dpngen.auto_test.lib.mfp_eosfit.ext_vec(func, fin, p0, fs, fe, vols=None, vole=None, ndata=101, refit=0, show_fig=False)`

Extrapolate the data points for E-V based on the fitted parameters in small or very large volume range.

`dpngen.auto_test.lib.mfp_eosfit.ext_velp(fin, fstart, fend, vols, vole, ndata, order=3, method='unispl', fout='ext_velp.dat', show_fig=False)`

Extrapolate the lattice parameters based on input data.

`dpngen.auto_test.lib.mfp_eosfit.get_eos_list()`

```

dpgen.auto_test.lib.mfp_eosfit.get_eos_list_3p()
dpgen.auto_test.lib.mfp_eosfit.get_eos_list_4p()
dpgen.auto_test.lib.mfp_eosfit.get_eos_list_5p()
dpgen.auto_test.lib.mfp_eosfit.get_eos_list_6p()
dpgen.auto_test.lib.mfp_eosfit.init_guess(fin)
dpgen.auto_test.lib.mfp_eosfit.lsqfit_eos(func, fin, par, fstart, fend, show_fig=False, fout='EoSfit.out',
                                          refit=-1)

dpgen.auto_test.lib.mfp_eosfit.mBM4(vol, pars)
    Birch-Murnaghan 4 pars equation from PRB 70, 224107, 3-order BM.
dpgen.auto_test.lib.mfp_eosfit.mBM4poly(vol, parameters)
    Modified BM5 EOS, Shang SL comput mater sci, 2010: 1040-1048, original expressions.
dpgen.auto_test.lib.mfp_eosfit.mBM5(vol, pars)
    Modified BM5 EOS, Shang SL comput mater sci, 2010: 1040-1048.
dpgen.auto_test.lib.mfp_eosfit.mBM5poly(vol, pars)
    Modified BM5 EOS, Shang SL comput mater sci, 2010: 1040-1048, original expressions.
dpgen.auto_test.lib.mfp_eosfit.mie(v, p)
    Mie model for song's FVT.
dpgen.auto_test.lib.mfp_eosfit.mie_simple(v, p)
    Mie_simple model for song's FVT.
dpgen.auto_test.lib.mfp_eosfit.morse(v, pars)
    Reproduce from ShunliShang's matlab script.
dpgen.auto_test.lib.mfp_eosfit.morse_3p(volume, p)
    morse_AB EOS formula from Song's FVT sources A= 0.5*B.
dpgen.auto_test.lib.mfp_eosfit.morse_6p(vol, par)
    Generalized Morse EOS proposed by Qin, see: Qin et al. Phys Rev B, 2008, 78, 214108. Qin et al. Phys Rev B,
    2008, 77, 220103(R).
dpgen.auto_test.lib.mfp_eosfit.morse_AB(volume, p)
    morse_AB EOS formula from Song's FVT sources.
dpgen.auto_test.lib.mfp_eosfit.murnaghan(vol, pars)
    Four-parameters murnaghan EOS. From PRB 28,5480 (1983).
dpgen.auto_test.lib.mfp_eosfit.parse_argument()
dpgen.auto_test.lib.mfp_eosfit.rBM4(vol, pars)
    Implementions as Alberto Otero-de-la-Roza, i.e. rBM4 is used here Comput Physics Comm, 2011, 182: 1708-
    1720.
dpgen.auto_test.lib.mfp_eosfit.rBM4_pv(vol, pars)
    Implementions as Alberto Otero-de-la-Roza, i.e. rBM4 is used here Comput Physics Comm, 2011, 182: 1708-
    1720 Fit for V-P relations.

```

`dpgen.auto_test.lib.mfp_eosfit.rBM5(vol, pars)`

Implementions as Alberto Otero-de-la-Roza, i.e. rBM5 is used here Comput Physics Comm, 2011, 182: 1708-1720.

`dpgen.auto_test.lib.mfp_eosfit.rBM5_pv(vol, pars)`

Implementions as Alberto Otero-de-la-Roza, i.e. rBM5 is used here Comput Physics Comm, 2011, 182: 1708-1720 Fit for V-P relations.

`dpgen.auto_test.lib.mfp_eosfit.rPT4(vol, pars)`

Natrual strain EOS with 4 paramters Seems only work in near-equilibrium range. Implementions as Alberto Otero-de-la-Roza, i.e. rPT4 is used here Comput Physics Comm, 2011, 182: 1708-1720, in their article, labeled as PT3 (3-order), however, we mention it as rPT4 for 4-parameters EOS.

`dpgen.auto_test.lib.mfp_eosfit.rPT4_pv(vol, pars)`

Natrual strain (Poirier-Tarantola)EOS with 4 paramters Seems only work in near-equilibrium range. Implementions as Alberto Otero-de-la-Roza, i.e. rPT4 is used here Comput Physics Comm, 2011, 182: 1708-1720, in their article, labeled as PT3 (3-order), however, we mention it as rPT4 for 4-parameters EOS.

`dpgen.auto_test.lib.mfp_eosfit.rPT5(vol, pars)`

Natrual strain EOS with 4 paramters Seems only work in near-equilibrium range. Implementions as Alberto Otero-de-la-Roza, i.e. rPT5 is used here Comput Physics Comm, 2011, 182: 1708-1720, in their article, labeled as PT3 (3-order), however, we mention it as rPT5 for 4-parameters EOS.

`dpgen.auto_test.lib.mfp_eosfit.rPT5_pv(vol, pars)`

Natrual strain (Poirier-Tarantola)EOS with 5 paramters Implementions as Alberto Otero-de-la-Roza, i.e. rPT5 is used here Comput Physics Comm, 2011, 182: 1708-1720, in their article, labeled as PT3 (3-order), however, we mention it as rPT5 for 4-parameters EOS.

`dpgen.auto_test.lib.mfp_eosfit.read_ve(fin)`

`dpgen.auto_test.lib.mfp_eosfit.read_velp(fin, fstart, fend)`

`dpgen.auto_test.lib.mfp_eosfit.read_vlp(fin, fstart, fend)`

`dpgen.auto_test.lib.mfp_eosfit.repro_ve(func, vol_i, p)`

`dpgen.auto_test.lib.mfp_eosfit.repro_vp(func, vol_i, pars)`

`dpgen.auto_test.lib.mfp_eosfit.res_BM4(pars, y, x)`

`dpgen.auto_test.lib.mfp_eosfit.res_BM5(pars, y, x)`

`dpgen.auto_test.lib.mfp_eosfit.res_LOG4(pars, y, x)`

`dpgen.auto_test.lib.mfp_eosfit.res_LOG5(pars, y, x)`

`dpgen.auto_test.lib.mfp_eosfit.res_Li4p(p, y, x)`

`dpgen.auto_test.lib.mfp_eosfit.res_SJX_5p(p, e, v)`

`dpgen.auto_test.lib.mfp_eosfit.res_SJX_v2(p, e, v)`

`dpgen.auto_test.lib.mfp_eosfit.res_TEOS(p, e, v)`

`dpgen.auto_test.lib.mfp_eosfit.res_birch(pars, y, x)`

`dpgen.auto_test.lib.mfp_eosfit.res_mBM4(pars, y, x)`

`dpgen.auto_test.lib.mfp_eosfit.res_mBM4poly(pars, y, x)`

```

dpgen.auto_test.lib.mfp_eosfit.res_mBM5(pars, y, x)
dpgen.auto_test.lib.mfp_eosfit.res_mBM5poly(pars, y, x)
dpgen.auto_test.lib.mfp_eosfit.res_mie(p, e, v)
dpgen.auto_test.lib.mfp_eosfit.res_mie_simple(p, e, v)
dpgen.auto_test.lib.mfp_eosfit.res_morse(p, en, volume)
dpgen.auto_test.lib.mfp_eosfit.res_morse_3p(p, en, volume)
dpgen.auto_test.lib.mfp_eosfit.res_morse_6p(p, en, volume)
dpgen.auto_test.lib.mfp_eosfit.res_morse_AB(p, en, volume)
dpgen.auto_test.lib.mfp_eosfit.res_murnaghan(pars, y, x)
dpgen.auto_test.lib.mfp_eosfit.res_rBM4(pars, y, x)
dpgen.auto_test.lib.mfp_eosfit.res_rBM4_pv(pars, y, x)
dpgen.auto_test.lib.mfp_eosfit.res_rBM5(pars, y, x)
dpgen.auto_test.lib.mfp_eosfit.res_rBM5_pv(pars, y, x)
dpgen.auto_test.lib.mfp_eosfit.res_rPT4(pars, y, x)
dpgen.auto_test.lib.mfp_eosfit.res_rPT4_pv(pars, y, x)
dpgen.auto_test.lib.mfp_eosfit.res_rPT5(pars, y, x)
dpgen.auto_test.lib.mfp_eosfit.res_rPT5_pv(pars, y, x)
dpgen.auto_test.lib.mfp_eosfit.res_universal(pars, y, x)
dpgen.auto_test.lib.mfp_eosfit.res_vinet(pars, y, x)
dpgen.auto_test.lib.mfp_eosfit.res_vinet_pv(pars, y, x)
dpgen.auto_test.lib.mfp_eosfit.universal(vol, parameters)
    Universal equation of state(Vinet P et al., J. Phys.: Condens. Matter 1, p1941 (1989)).
dpgen.auto_test.lib.mfp_eosfit.vinet(vol, pars)
    Vinet equation from PRB 70, 224107 Following, Shang Shunli et al., comput mater sci, 2010: 1040-1048,
    original expressions.
dpgen.auto_test.lib.mfp_eosfit.vinet_pv(vol, pars)

```

dpgen.auto_test.lib.pwscf module

```

dpgen.auto_test.lib.pwscf.make_pwscf_input(sys_data, fp_pp_files, fp_params)

```

dpngen.auto_test.lib.siesta module

`dpngen.auto_test.lib.siesta.make_siesta_input(sys_data, fp_pp_files, fp_params)`

dpngen.auto_test.lib.util module

`dpngen.auto_test.lib.util.collect_task(all_task, task_type)`

`dpngen.auto_test.lib.util.get_machine_info(mdata, task_type)`

`dpngen.auto_test.lib.util.insert_data(task, task_type, username, file_name)`

`dpngen.auto_test.lib.util.make_work_path(jdata, task, reprod_opt, static, user)`

`dpngen.auto_test.lib.util.voigt_to_stress(inpt)`

dpngen.auto_test.lib.utils module

`dpngen.auto_test.lib.utils.cmd_append_log(cmd, log_file)`

`dpngen.auto_test.lib.utils.copy_file_list(file_list, from_path, to_path)`

`dpngen.auto_test.lib.utils.create_path(path)`

`dpngen.auto_test.lib.utils.log_iter(task, ii, jj)`

`dpngen.auto_test.lib.utils.log_task(message)`

`dpngen.auto_test.lib.utils.make_iter_name(iter_index)`

`dpngen.auto_test.lib.utils.record_iter(record, confs, ii, jj)`

`dpngen.auto_test.lib.utils.repeat_to_length(string_to_expand, length)`

`dpngen.auto_test.lib.utils.replace(file_name, pattern, subst)`

dpngen.auto_test.lib.vasp module

exception `dpngen.auto_test.lib.vasp.OutcarItemError`

Bases: `Exception`

`dpngen.auto_test.lib.vasp.check_finished(fname)`

`dpngen.auto_test.lib.vasp.get_boxes(fname)`

`dpngen.auto_test.lib.vasp.get_energies(fname)`

`dpngen.auto_test.lib.vasp.get_nev(fname)`

`dpngen.auto_test.lib.vasp.get_poscar_natoms(fname)`

`dpngen.auto_test.lib.vasp.get_poscar_types(fname)`

`dpngen.auto_test.lib.vasp.get_stress(fname)`

```

dpngen.auto_test.lib.vasp.make_kspacing_kpoints(poscar, kspacing, kgamma)
dpngen.auto_test.lib.vasp.make_vasp_kpoints(kpoints, kgamma=False)
dpngen.auto_test.lib.vasp.make_vasp_kpoints_from_incar(work_dir, jdata)
dpngen.auto_test.lib.vasp.make_vasp_phonon_incar(ecut, ediff, npar, kpar, kspacing=0.5, kgamma=True,
ismear=1, sigma=0.2)
dpngen.auto_test.lib.vasp.make_vasp_relax_incar(ecut, ediff, relax_ion, relax_shape, relax_volume,
npar, kpar, kspacing=0.5, kgamma=True, ismear=1,
sigma=0.22)
dpngen.auto_test.lib.vasp.make_vasp_static_incar(ecut, ediff, npar, kpar, kspacing=0.5, kgamma=True,
ismear=1, sigma=0.2)
dpngen.auto_test.lib.vasp.perturb_xz(poscar_in, poscar_out, pert=0.01)
dpngen.auto_test.lib.vasp.poscar_natoms(poscar_in)
dpngen.auto_test.lib.vasp.poscar_scale(poscar_in, poscar_out, scale)
dpngen.auto_test.lib.vasp.poscar_vol(poscar_in)
dpngen.auto_test.lib.vasp.reciprocal_box(box)
dpngen.auto_test.lib.vasp.regulate_poscar(poscar_in, poscar_out)
dpngen.auto_test.lib.vasp.sort_poscar(poscar_in, poscar_out, new_names)

```

Submodules

dpngen.auto_test.ABACUS module

```
class dpngen.auto_test.ABACUS.ABACUS(inter_parameter, path_to_poscar)
```

Bases: [Task](#)

Methods

<i>backward_files</i> ([property_type])	Return backward files.
<i>compute</i> (output_dir)	Compute output of the task.
<i>forward_common_files</i> ([property_type])	Return forward common files.
<i>forward_files</i> ([property_type])	Return forward files.
<i>make_input_file</i> (output_dir, task_type, ...)	Prepare input files for a computational task For example, the VASP prepares INCAR.
<i>make_potential_files</i> (output_dir)	Prepare potential files for a computational task.

modify_input

```
backward_files(property_type='relaxation')
```

Return backward files.

compute(*output_dir*)

Compute output of the task. IMPORTANT: The output configuration should be converted and stored in a CONTCAR file.

Parameters**output_dir**

[str] The directory storing the input and output files.

Returns**result_dict: dict**

A dict that storing the result. For example: { "energy": xxx, "force": [xxx] }

Notes

The following files are generated: CONTCAR: output file

The output configuration is converted to CONTCAR and stored in the *output_dir*

forward_common_files(*property_type='relaxation'*)

Return forward common files.

forward_files(*property_type='relaxation'*)

Return forward files.

make_input_file(*output_dir, task_type, task_param*)

Prepare input files for a computational task For example, the VASP prepares INCAR. LAMMPS (including DeePMD, MEAM...) prepares in.lammps.

Parameters**output_dir**

[str] The directory storing the input files.

task_type

[str] Can be - "relaxation": structure relaxation - "static": static computation calculates the energy, force... of a strcture

task_param

[dict] The parameters of the task. For example the VASP interaction can be provided with { "ediff": 1e-6, "ediffg": 1e-5 }

make_potential_files(*output_dir*)

Prepare potential files for a computational task. For example, the VASP prepares POTCAR. DeePMD prepares frozen model(s). IMPORTANT: Interaction should be stored in *output_dir/inter.json*.

Parameters**output_dir**

[str] The directory storing the potential files.

Notes

The following files are generated:

inter.json: output file

The task information is stored in *output_dir/inter.json*

modify_input(*incard, x, y*)

dpngen.auto_test.EOS module

class dpngen.auto_test.EOS.**EOS**(parameter, inter_param=None)

Bases: *Property*

Methods

<code>compute(output_file, print_file, path_to_work)</code>	Postprocess the finished tasks to compute the property.
<code>make_confs(path_to_work, path_to_equi[, refine])</code>	Make configurations needed to compute the property.
<code>post_process(task_list)</code>	post_process the KPOINTS file in elastic.
<code>task_param()</code>	Return the parameter of each computational task, for example, {'ediffg': 1e-4}.
<code>task_type()</code>	Return the type of each computational task, for example, 'relaxation', 'static'....

make_confs(path_to_work, path_to_equi, refine=False)

Make configurations needed to compute the property. The tasks directory will be named as path_to_work/task.xxxxxx IMPORTANT: handel the case when the directory exists.

Parameters

path_to_work

[str] The path where the tasks for the property are located

path_to_equi

[str] -refine == False: The path to the directory that equilibrated the configuration.
-refine == True: The path to the directory that has property confs.

refine

[str] To refine existing property confs or generate property confs from a equilibrated conf

Returns

task_list: list of str

The list of task directories.

post_process(task_list)

post_process the KPOINTS file in elastic.

task_param()

Return the parameter of each computational task, for example, {'ediffg': 1e-4}.

task_type()

Return the type of each computational task, for example, 'relaxation', 'static'....

dpngen.auto_test.Elastic module

class dpngen.auto_test.Elastic.**Elastic**(parameter, inter_param=None)

Bases: *Property*

Methods

<code>compute(output_file, print_file, path_to_work)</code>	Postprocess the finished tasks to compute the property.
<code>make_confs(path_to_work, path_to_equi[, refine])</code>	Make configurations needed to compute the property.
<code>post_process(task_list)</code>	post_process the KPOINTS file in elastic.
<code>task_param()</code>	Return the parameter of each computational task, for example, {'ediffg': 1e-4}.
<code>task_type()</code>	Return the type of each computational task, for example, 'relaxation', 'static'....

make_confs(*path_to_work*, *path_to_equi*, *refine=False*)

Make configurations needed to compute the property. The tasks directory will be named as *path_to_work/task.xxxxxx* IMPORTANT: handel the case when the directory exists.

Parameters

path_to_work

[str] The path where the tasks for the property are located

path_to_equi

[str] -refine == False: The path to the directory that equilibrated the configuration.
-refine == True: The path to the directory that has property confs.

refine

[str] To refine existing property confs or generate property confs from a equilibrated conf

Returns

task_list: list of str

The list of task directories.

post_process(*task_list*)

post_process the KPOINTS file in elastic.

task_param()

Return the parameter of each computational task, for example, {'ediffg': 1e-4}.

task_type()

Return the type of each computational task, for example, 'relaxation', 'static'....

dpgen.auto_test.Gamma module

class dpgen.auto_test.Gamma.**Gamma**(*parameter*, *inter_param=None*)

Bases: *Property*

Calculation of common gamma lines for bcc and fcc.

Methods

<code>compute(output_file, print_file, path_to_work)</code>	Postprocess the finished tasks to compute the property.
<code>make_confs(path_to_work, path_to_equi[, refine])</code>	Make configurations needed to compute the property.
<code>post_process(task_list)</code>	post_process the KPOINTS file in elastic.
<code>task_param()</code>	Return the parameter of each computational task, for example, {'ediffg': 1e-4}.
<code>task_type()</code>	Return the type of each computational task, for example, 'relaxation', 'static'....

centralize_slab return_direction

static **centralize_slab**(*slab*) → None

make_confs(*path_to_work*, *path_to_equi*, *refine=False*)

Make configurations needed to compute the property. The tasks directory will be named as *path_to_work/task.xxxxxx* IMPORTANT: handel the case when the directory exists.

Parameters

path_to_work

[str] The path where the tasks for the property are located

path_to_equi

[str] -refine == False: The path to the directory that equilibrated the configuration.
-refine == True: The path to the directory that has property confs.

refine

[str] To refine existing property confs or generate property confs from a equilibrated conf

Returns

task_list: list of str

The list of task directories.

post_process(*task_list*)

post_process the KPOINTS file in elastic.

return_direction()

task_param()

Return the parameter of each computational task, for example, {'ediffg': 1e-4}.

task_type()

Return the type of each computational task, for example, 'relaxation', 'static'....

dpngen.auto_test.Interstitial module

```
class dpngen.auto_test.Interstitial.Interstitial(parameter, inter_param=None)
```

Bases: *Property*

Methods

<code>compute(output_file, print_file, path_to_work)</code>	Postprocess the finished tasks to compute the property.
<code>make_confs(path_to_work, path_to_equi[, refine])</code>	Make configurations needed to compute the property.
<code>post_process(task_list)</code>	post_process the KPOINTS file in elastic.
<code>task_param()</code>	Return the parameter of each computational task, for example, {'ediffg': 1e-4}.
<code>task_type()</code>	Return the type of each computational task, for example, 'relaxation', 'static'....

```
make_confs(path_to_work, path_to_equi, refine=False)
```

Make configurations needed to compute the property. The tasks directory will be named as path_to_work/task.xxxxxx IMPORTANT: handel the case when the directory exists.

Parameters**path_to_work**

[str] The path where the tasks for the property are located

path_to_equi

[str] -refine == False: The path to the directory that equilibrated the configuration.
-refine == True: The path to the directory that has property confs.

refine

[str] To refine existing property confs or generate property confs from a equilibrated conf

Returns**task_list: list of str**

The list of task directories.

```
post_process(task_list)
```

post_process the KPOINTS file in elastic.

```
task_param()
```

Return the parameter of each computational task, for example, {'ediffg': 1e-4}.

```
task_type()
```

Return the type of each computational task, for example, 'relaxation', 'static'....

dpngen.auto_test.Lammps module

```
class dpngen.auto_test.Lammps.Lammps(inter_parameter, path_to_poscar)
```

Bases: *Task*

Methods

<code>backward_files([property_type])</code>	Return backward files.
<code>compute(output_dir)</code>	Compute output of the task.
<code>forward_common_files([property_type])</code>	Return forward common files.
<code>forward_files([property_type])</code>	Return forward files.
<code>make_input_file(output_dir, task_type, ...)</code>	Prepare input files for a computational task For example, the VASP prepares INCAR.
<code>make_potential_files(output_dir)</code>	Prepare potential files for a computational task.

set_inter_type_func set_model_param
--

backward_files(*property_type*='relaxation')

Return backward files.

compute(*output_dir*)

Compute output of the task. IMPORTANT: The output configuration should be converted and stored in a CONTCAR file.

Parameters

output_dir

[str] The directory storing the input and output files.

Returns

result_dict: dict

A dict that storing the result. For example: { "energy": xxx, "force": [xxx] }

Notes

The following files are generated: CONTCAR: output file

The output configuration is converted to CONTCAR and stored in the *output_dir*

forward_common_files(*property_type*='relaxation')

Return forward common files.

forward_files(*property_type*='relaxation')

Return forward files.

make_input_file(*output_dir, task_type, task_param*)

Prepare input files for a computational task For example, the VASP prepares INCAR. LAMMPS (including DeePMD, MEAM...) prepares in.lammps.

Parameters

output_dir

[str] The directory storing the input files.

task_type

[str] Can be - "relaxation": structure relaxation - "static": static computation calculates the energy, force... of a structure

task_param

[dict] The parameters of the task. For example the VASP interaction can be provided with { "ediff": 1e-6, "ediffg": 1e-5 }

make_potential_files(*output_dir*)

Prepare potential files for a computational task. For example, the VASP prepares POTCAR. DeePMD prepares frozen model(s). IMPORTANT: Interaction should be stored in *output_dir/inter.json*.

Parameters**output_dir**

[str] The directory storing the potential files.

Notes

The following files are generated:

inter.json: output file

The task information is stored in *output_dir/inter.json*

set_inter_type_func()**set_model_param()****dpngen.auto_test.Property module****class dpngen.auto_test.Property.Property**(*parameter*)

Bases: ABC

Attributes**task_param**

Return the parameter of each computational task, for example, {'ediffg': 1e-4}.

task_type

Return the type of each computational task, for example, 'relaxation', 'static'....

Methods

<i>compute</i> (<i>output_file</i> , <i>print_file</i> , <i>path_to_work</i>)	Postprocess the finished tasks to compute the property.
<i>make_confs</i> (<i>path_to_work</i> , <i>path_to_equi</i> [, <i>refine</i>])	Make configurations needed to compute the property.
<i>post_process</i> (<i>task_list</i>)	post_process the KPOINTS file in elastic.

compute(*output_file*, *print_file*, *path_to_work*)

Postprocess the finished tasks to compute the property. Output the result to a json database.

Parameters**output_file:**

The file to output the property in json format

print_file:

The file to output the property in txt format

path_to_work:

The working directory where the computational tasks locate.

abstract make_confs(*path_to_work*, *path_to_equi*, *refine=False*)

Make configurations needed to compute the property. The tasks directory will be named as *path_to_work/task.xxxxxx* IMPORTANT: handel the case when the directory exists.

Parameters**path_to_work**

[str] The path where the tasks for the property are located

path_to_equi

[str] -refine == False: The path to the directory that equilibrated the configuration.
 -refine == True: The path to the directory that has property confs.

refine

[str] To refine existing property confs or generate property confs from a equilibrated conf

Returns**task_list: list of str**

The list of task directories.

abstract post_process(task_list)

post_process the KPOINTS file in elastic.

abstract property task_param

Return the parameter of each computational task, for example, {'ediffg': 1e-4}.

abstract property task_type

Return the type of each computational task, for example, 'relaxation', 'static'....

dpngen.auto_test.Surface module**class dpngen.auto_test.Surface.Surface(parameter, inter_param=None)**

Bases: *Property*

Methods

<code>compute(output_file, print_file, path_to_work)</code>	Postprocess the finished tasks to compute the property.
<code>make_confs(path_to_work, path_to_equi[, refine])</code>	Make configurations needed to compute the property.
<code>post_process(task_list)</code>	post_process the KPOINTS file in elastic.
<code>task_param()</code>	Return the parameter of each computational task, for example, {'ediffg': 1e-4}.
<code>task_type()</code>	Return the type of each computational task, for example, 'relaxation', 'static'....

make_confs(path_to_work, path_to_equi, refine=False)

Make configurations needed to compute the property. The tasks directory will be named as path_to_work/task.xxxxxx IMPORTANT: handel the case when the directory exists.

Parameters**path_to_work**

[str] The path where the tasks for the property are located

path_to_equi

[str] -refine == False: The path to the directory that equilibrated the configuration.
 -refine == True: The path to the directory that has property confs.

refine

[str] To refine existing property confs or generate property confs from a equilibrated conf

Returns**task_list: list of str**

The list of task directories.

post_process(*task_list*)

post_process the KPOINTS file in elastic.

task_param()

Return the parameter of each computational task, for example, {'ediffg': 1e-4}.

task_type()

Return the type of each computational task, for example, 'relaxation', 'static'...

dpgen.auto_test.Task module

class dpgen.auto_test.Task.**Task**(*inter_parameter*, *path_to_poscar*)

Bases: ABC

Attributes

backward_files

Return backward files.

forward_common_files

Return forward common files.

forward_files

Return forward files.

Methods

<i>compute</i> (<i>output_dir</i>)	Compute output of the task.
<i>make_input_file</i> (<i>output_dir</i> , <i>task_type</i> , ...)	Prepare input files for a computational task For example, the VASP prepares INCAR.
<i>make_potential_files</i> (<i>output_dir</i>)	Prepare potential files for a computational task.

abstract property backward_files

Return backward files.

abstract compute(*output_dir*)

Compute output of the task. IMPORTANT: The output configuration should be converted and stored in a CONTCAR file.

Parameters

output_dir

[str] The directory storing the input and output files.

Returns

result_dict: dict

A dict that storing the result. For example: { "energy": xxx, "force": [xxx] }

Notes

The following files are generated: CONTCAR: output file

The output configuration is converted to CONTCAR and stored in the *output_dir*

abstract property forward_common_files

Return forward common files.

abstract property forward_files

Return forward files.

abstract make_input_file(*output_dir*, *task_type*, *task_param*)

Prepare input files for a computational task For example, the VASP prepares INCAR. LAMMPS (including DeePMD, MEAM...) prepares in.lammps.

Parameters

output_dir

[str] The directory storing the input files.

task_type

[str] Can be - “relaxation”: structure relaxation - “static”: static computation calculates the energy, force... of a strcture

task_param

[dict] The parameters of the task. For example the VASP interaction can be provided with { “ediff”: 1e-6, “ediffg”: 1e-5 }

abstract make_potential_files(*output_dir*)

Prepare potential files for a computational task. For example, the VASP prepares POTCAR. DeePMD prepares frozen model(s). IMPORTANT: Interaction should be stored in *output_dir/inter.json*.

Parameters

output_dir

[str] The directory storing the potential files.

Notes

The following files are generated:

inter.json: output file

The task information is stored in *output_dir/inter.json*

dpngen.auto_test.VASP module

class dpngen.auto_test.VASP.VASP(*inter_parameter*, *path_to_poscar*)

Bases: *Task*

Methods

<i>backward_files</i> ([<i>property_type</i>])	Return backward files.
<i>compute</i> (<i>output_dir</i>)	Compute output of the task.
<i>forward_common_files</i> ([<i>property_type</i>])	Return forward common files.
<i>forward_files</i> ([<i>property_type</i>])	Return forward files.
<i>make_input_file</i> (<i>output_dir</i> , <i>task_type</i> , ...)	Prepare input files for a computational task For example, the VASP prepares INCAR.
<i>make_potential_files</i> (<i>output_dir</i>)	Prepare potential files for a computational task.

backward_files(*property_type='relaxation'*)

Return backward files.

compute(*output_dir*)

Compute output of the task. IMPORTANT: The output configuration should be converted and stored in a CONTCAR file.

Parameters

output_dir

[str] The directory storing the input and output files.

Returns

result_dict: dict

A dict that storing the result. For example: { "energy": xxx, "force": [xxx] }

Notes

The following files are generated: CONTCAR: output file

The output configuration is converted to CONTCAR and stored in the *output_dir*

forward_common_files(*property_type='relaxation'*)

Return forward common files.

forward_files(*property_type='relaxation'*)

Return forward files.

make_input_file(*output_dir, task_type, task_param*)

Prepare input files for a computational task For example, the VASP prepares INCAR. LAMMPS (including DeePMD, MEAM...) prepares in.lammps.

Parameters

output_dir

[str] The directory storing the input files.

task_type

[str] Can be - "relaxation": structure relaxation - "static": static computation calculates the energy, force... of a strcture

task_param

[dict] The parameters of the task. For example the VASP interaction can be provided with { "ediff": 1e-6, "ediffg": 1e-5 }

make_potential_files(*output_dir*)

Prepare potential files for a computational task. For example, the VASP prepares POTCAR. DeePMD prepares frozen model(s). IMPORTANT: Interaction should be stored in *output_dir/inter.json*.

Parameters

output_dir

[str] The directory storing the potential files.

Notes

The following files are generated:

inter.json: output file

The task information is stored in *output_dir/inter.json*

dpngen.auto_test.Vacancy module

class dpngen.auto_test.Vacancy.Vacancy(*parameter*, *inter_param*=None)

Bases: *Property*

Methods

<code>compute(output_file, print_file, path_to_work)</code>	Postprocess the finished tasks to compute the property.
<code>make_confs(path_to_work, path_to_equi[, refine])</code>	Make configurations needed to compute the property.
<code>post_process(task_list)</code>	post_process the KPOINTS file in elastic.
<code>task_param()</code>	Return the parameter of each computational task, for example, {'ediffg': 1e-4}.
<code>task_type()</code>	Return the type of each computational task, for example, 'relaxation', 'static'....

make_confs(*path_to_work*, *path_to_equi*, *refine*=False)

Make configurations needed to compute the property. The tasks directory will be named as *path_to_work/task.xxxxxx* IMPORTANT: handel the case when the directory exists.

Parameters

path_to_work

[str] The path where the tasks for the property are located

path_to_equi

[str] -refine == False: The path to the directory that equilibrated the configuration.
-refine == True: The path to the directory that has property confs.

refine

[str] To refine existing property confs or generate property confs from a equilibrated conf

Returns

task_list: list of str

The list of task directories.

post_process(*task_list*)

post_process the KPOINTS file in elastic.

task_param()

Return the parameter of each computational task, for example, {'ediffg': 1e-4}.

task_type()

Return the type of each computational task, for example, 'relaxation', 'static'....

dpngen.auto_test.calculator module

dpngen.auto_test.calculator.**make_calculator**(*inter_parameter*, *path_to_poscar*)

Make an instance of Task.

dpgen.auto_test.common_equi module

`dpgen.auto_test.common_equi.make_equi`(*confs*, *inter_param*, *relax_param*)

`dpgen.auto_test.common_equi.post_equi`(*confs*, *inter_param*)

`dpgen.auto_test.common_equi.run_equi`(*confs*, *inter_param*, *mdata*)

dpgen.auto_test.common_prop module

`dpgen.auto_test.common_prop.make_property`(*confs*, *inter_param*, *property_list*)

`dpgen.auto_test.common_prop.make_property_instance`(*parameters*, *inter_param*)

Make an instance of Property.

`dpgen.auto_test.common_prop.post_property`(*confs*, *inter_param*, *property_list*)

`dpgen.auto_test.common_prop.run_property`(*confs*, *inter_param*, *property_list*, *mdata*)

`dpgen.auto_test.common_prop.worker`(*work_path*, *all_task*, *forward_common_files*, *forward_files*,
backward_files, *mdata*, *inter_type*)

dpgen.auto_test.gen_confs module

`dpgen.auto_test.gen_confs.gen_alloy`(*eles*, *key*)

`dpgen.auto_test.gen_confs.gen_ele_std`(*ele_name*, *ctype*)

`dpgen.auto_test.gen_confs.gen_element`(*ele_name*, *key*)

`dpgen.auto_test.gen_confs.gen_element_std`(*ele_name*)

`dpgen.auto_test.gen_confs.make_path_mp`(*ii*)

`dpgen.auto_test.gen_confs.test_fit`(*struct*, *data*)

dpgen.auto_test.mpdb module

`dpgen.auto_test.mpdb.check_apikey`()

`dpgen.auto_test.mpdb.get_structure`(*mp_id*)

dpgen.auto_test.refine module

`dpgen.auto_test.refine.make_refine`(*init_from_suffix*, *output_suffix*, *path_to_work*)

dpngen.auto_test.reproduce module

`dpngen.auto_test.reproduce.make_repro`(*inter_param*, *init_data_path*, *init_from_suffix*, *path_to_work*,
reprod_last_frame=True)

`dpngen.auto_test.reproduce.post_repro`(*init_data_path*, *init_from_suffix*, *all_tasks*, *ptr_data*,
reprod_last_frame=True)

dpngen.auto_test.run module

`dpngen.auto_test.run.gen_test`(*args*)

`dpngen.auto_test.run.run_task`(*step*, *param_file*, *machine_file=None*)

dpngen.collect package

Submodules

dpngen.collect.collect module

`dpngen.collect.collect.collect_data`(*target_folder*, *param_file*, *output*, *verbose=True*, *shuffle=True*,
merge=True)

`dpngen.collect.collect.gen_collect`(*args*)

dpngen.data package

Subpackages

dpngen.data.tools package

Submodules

dpngen.data.tools.bcc module

`dpngen.data.tools.bcc.gen_box`()

`dpngen.data.tools.bcc.numb_atoms`()

`dpngen.data.tools.bcc.poscar_unit`(*latt*)

dpgen.data.tools.cessp2force_lin module

```
dpgen.data.tools.cessp2force_lin.Parser()
dpgen.data.tools.cessp2force_lin.get_outcar_files(directory, recursive)
dpgen.data.tools.cessp2force_lin.process_outcar_file_v5_dev(outcars, data, numbers, types,
                                                            max_types, elements=None,
                                                            windex=None, fout='potfit.configs')
dpgen.data.tools.cessp2force_lin.scan_outcar_file(file_handle)
dpgen.data.tools.cessp2force_lin.uniq(seq)
```

dpgen.data.tools.create_random_disturb module

```
dpgen.data.tools.create_random_disturb.RandomDisturbParser()
dpgen.data.tools.create_random_disturb.create_disturbs_abacus_dev(fin, nfile, dmax=1.0,
                                                                etmax=0.1, ofmt='abacus',
                                                                dstyle='uniform',
                                                                write_d=False, diag=0)
dpgen.data.tools.create_random_disturb.create_disturbs_ase(fin, nfile, dmax=1.0, ofmt='lmp',
                                                            dstyle='uniform', write_d=False)
dpgen.data.tools.create_random_disturb.create_disturbs_ase_dev(fin, nfile, dmax=1.0, etmax=0.1,
                                                                ofmt='lmp', dstyle='uniform',
                                                                write_d=False, diag=0)
dpgen.data.tools.create_random_disturb.create_disturbs_atomsk(fin, nfile, dmax=1.0, ofmt='lmp')
dpgen.data.tools.create_random_disturb.create_random_alloys(fin, alloy_dist, ifmt='vasp',
                                                            ofmt='vasp')
```

In fact, atomsk also gives us the convinient tool to do this.

```
dpgen.data.tools.create_random_disturb.gen_random_disturb(dmax, a, b, dstyle='uniform')
dpgen.data.tools.create_random_disturb.gen_random_emat(etmax, diag=0)
dpgen.data.tools.create_random_disturb.random_range(a, b, ndata=1)
```

dpgen.data.tools.diamond module

```
dpgen.data.tools.diamond.gen_box()
dpgen.data.tools.diamond.numb_atoms()
dpgen.data.tools.diamond.poscar_unit(latt)
```

dpngen.data.tools.fcc module

```
dpngen.data.tools.fcc.gen_box()
dpngen.data.tools.fcc.numb_atoms()
dpngen.data.tools.fcc.poscar_unit(latt)
```

dpngen.data.tools.hcp module

```
dpngen.data.tools.hcp.gen_box()
dpngen.data.tools.hcp.numb_atoms()
dpngen.data.tools.hcp.poscar_unit(latt)
```

dpngen.data.tools.io_lammps module

ASE Atoms convert to LAMMPS configuration Some functions are adapted from ASE lammpsrun.py.

```
dpngen.data.tools.io_lammps.ase2lammpsdata(atoms, typeids=None, fout='out.lmp')
dpngen.data.tools.io_lammps.car2dir(v, Ainv)
    Cartesian to direct coordinates.
dpngen.data.tools.io_lammps.convert_cell(ase_cell)
    Convert a parallel piped (forming right hand basis) to lower triangular matrix LAMMPS can accept. This function
    transposes cell matrix so the bases are column vectors.
dpngen.data.tools.io_lammps.convert_forces(forces0, cell0, cell_new)
dpngen.data.tools.io_lammps.convert_positions(pos0, cell0, cell_new, direct=False)
dpngen.data.tools.io_lammps.convert_stress(s6_0, cell0, cell_new)
dpngen.data.tools.io_lammps.dir2car(v, A)
    Direct to cartesian coordinates.
dpngen.data.tools.io_lammps.get_atoms_ntypes(atoms)
dpngen.data.tools.io_lammps.get_typeid(typeids, csymbol)
dpngen.data.tools.io_lammps.is_upper_triangular(mat)
    Test if 3x3 matrix is upper triangular LAMMPS has a rule for cell matrix definition.
dpngen.data.tools.io_lammps.set_atoms_typeids(atoms)
dpngen.data.tools.io_lammps.set_atoms_typeids_with_atomic_numbers(atoms)
dpngen.data.tools.io_lammps.stress6_to_stress9(s6)
dpngen.data.tools.io_lammps.stress9_to_stress6(s9)
```

dpgen.data.tools.ovito_file_convert module

dpgen.data.tools.poscar_copy module

dpgen.data.tools.sc module

`dpgen.data.tools.sc.gen_box()`

`dpgen.data.tools.sc.numb_atoms()`

`dpgen.data.tools.sc.poscar_unit(latt)`

Submodules

dpgen.data.arginfo module

`dpgen.data.arginfo.init_bulk_abacus_args()` → list[Argument]

`dpgen.data.arginfo.init_bulk_jdata_arginfo()` → Argument

Generate arginfo for dpgen init_bulk jdata.

Returns

Argument

dpgen init_bulk jdata arginfo

`dpgen.data.arginfo.init_bulk_mdata_arginfo()` → Argument

Generate arginfo for dpgen init_bulk mdata.

Returns

Argument

arginfo

`dpgen.data.arginfo.init_bulk_variant_type_args()` → list[Variant]

`dpgen.data.arginfo.init_bulk_vasp_args()` → list[Argument]

`dpgen.data.arginfo.init_reaction_jdata_arginfo()` → Argument

Generate arginfo for dpgen init_reaction jdata.

Returns

Argument

dpgen init_reaction jdata arginfo

`dpgen.data.arginfo.init_reaction_mdata_arginfo()` → Argument

Generate arginfo for dpgen init_reaction mdata.

Returns

Argument

arginfo

`dpgen.data.arginfo.init_surf_jdata_arginfo()` → Argument

Generate arginfo for dpgen init_surf jdata.

Returns

Argument

dpgen init_surf jdata arginfo

`dpgen.data.arginfo.init_surf_mdata_arginfo()` → Argument

Generate arginfo for dpgen init_surf mdata.

Returns

Argument

arginfo

dpgen.data.gen module

`dpgen.data.gen.class_cell_type(jdata)`

`dpgen.data.gen.coll_abacus_md(jdata)`

`dpgen.data.gen.coll_vasp_md(jdata)`

`dpgen.data.gen.create_path(path, back=False)`

`dpgen.data.gen.gen_init_bulk(args)`

`dpgen.data.gen.make_abacus_md(jdata, mdata)`

`dpgen.data.gen.make_abacus_relax(jdata, mdata)`

`dpgen.data.gen.make_combines(dim, natoms)`

`dpgen.data.gen.make_scale(jdata)`

`dpgen.data.gen.make_scale_ABACUS(jdata)`

`dpgen.data.gen.make_super_cell(jdata)`

`dpgen.data.gen.make_super_cell_ABACUS(jdata, stru_data)`

`dpgen.data.gen.make_super_cell_STRU(jdata)`

`dpgen.data.gen.make_super_cell_poscar(jdata)`

`dpgen.data.gen.make_unit_cell(jdata)`

`dpgen.data.gen.make_unit_cell_ABACUS(jdata)`

`dpgen.data.gen.make_vasp_md(jdata, mdata)`

`dpgen.data.gen.make_vasp_relax(jdata, mdata)`

`dpgen.data.gen.out_dir_name(jdata)`

`dpgen.data.gen.pert_scaled(jdata)`

`dpgen.data.gen.place_element(jdata)`

`dpgen.data.gen.place_element_ABACUS(jdata, supercell_stru)`

`dpgen.data.gen.poscar_ele(poscar_in, poscar_out, eles, natoms)`

`dpgen.data.gen.poscar_natoms(lines)`

`dpgen.data.gen.poscar_scale(poscar_in, poscar_out, scale)`

```
dpngen.data.gen.poscar_scale_abacus(poscar_in, poscar_out, scale, jdata)
dpngen.data.gen.poscar_scale_cartesian(str_in, scale)
dpngen.data.gen.poscar_scale_direct(str_in, scale)
dpngen.data.gen.poscar_shuffle(poscar_in, poscar_out)
dpngen.data.gen.replace(file_name, pattern, subst)
dpngen.data.gen.run_abacus_md(jdata, mdata)
dpngen.data.gen.run_abacus_relax(jdata, mdata)
dpngen.data.gen.run_vasp_md(jdata, mdata)
dpngen.data.gen.run_vasp_relax(jdata, mdata)
dpngen.data.gen.shuffle_stru_data(supercell_stru)
dpngen.data.gen.stru_ele(supercell_stru, stru_out, eles, natoms, jdata, path_work)
```

dpngen.data.reaction module

input: trajectory 00: ReaxFF MD (lammmps) 01: build dataset (mddatasetbuilder) 02: fp (gaussian) 03: convert to deepmd data output: data.

```
dpngen.data.reaction.convert_data(jdata)
dpngen.data.reaction.gen_init_reaction(args)
dpngen.data.reaction.link_fp_input()
dpngen.data.reaction.link_reaxff(jdata)
dpngen.data.reaction.link_trj(jdata)
    Link lammppstrj.
dpngen.data.reaction.make_lmp(jdata)
dpngen.data.reaction.run_build_dataset(jdata, mdata, log_file='build_log')
dpngen.data.reaction.run_fp(jdata, mdata, log_file='output', forward_common_files=[])
dpngen.data.reaction.run_reaxff(jdata, mdata, log_file='reaxff_log')
```

dpngen.data.surf module

```
dpngen.data.surf.class_cell_type(jdata)
dpngen.data.surf.create_path(path)
dpngen.data.surf.gen_init_surf(args)
dpngen.data.surf.make_combines(dim, natoms)
dpngen.data.surf.make_scale(jdata)
```

```

dpgen.data.surf.make_super_cell_pymatgen(jdata)
dpgen.data.surf.make_unit_cell(jdata)
dpgen.data.surf.make_vasp_relax(jdata)
dpgen.data.surf.out_dir_name(jdata)
dpgen.data.surf.pert_scaled(jdata)
dpgen.data.surf.place_element(jdata)
dpgen.data.surf.poscar_ele(poscar_in, poscar_out, eles, natoms)
dpgen.data.surf.poscar_elong(poscar_in, poscar_out, elong, shift_center=True)
dpgen.data.surf.poscar_natoms(poscar_in)
dpgen.data.surf.poscar_scale(poscar_in, poscar_out, scale)
dpgen.data.surf.poscar_scale_cartesian(str_in, scale)
dpgen.data.surf.poscar_scale_direct(str_in, scale)
dpgen.data.surf.poscar_shuffle(poscar_in, poscar_out)
dpgen.data.surf.replace(file_name, pattern, subst)
dpgen.data.surf.run_vasp_relax(jdata, mdata)

```

dpgen.database package

```

class dpgen.database.DPPotcar(symbols=None, functional='PBE', pp_file=None, pp_lists=None)
    Bases: MSONable

```

Methods

<code>as_dict()</code>	A JSON serializable dict representation of an object.
<code>from_dict(d)</code>	
param d	
Dict representation.	
<code>to_json()</code>	Returns a json string representation of the MSONable object.
<code>unsafe_hash()</code>	Returns an hash of the current object.
<code>validate_monty_v1(_MSONable__input_value)</code>	Pydantic validator with correct signature for pydantic v1.x
<code>validate_monty_v2(_MSONable__input_value, _)</code>	Pydantic validator with correct signature for pydantic v2.x

```

from_file
write_file

```

as_dict()

A JSON serializable dict representation of an object.

classmethod from_dict(*d*)

Parameters

d – Dict representation.

Returns

MSONable class.

classmethod from_file(*filename*)

write_file(*filename*)

class dpgen.database.**Entry**(*composition, calculator, inputs, data, entry_id=None, attribute=None, tag=None*)

Bases: MSONable

An lightweight Entry object containing key computed data for storing purpose.

Parameters

composition

[Composition] Composition of the entry. For flexibility, this can take the form of all the typical input taken by a Composition, including a {symbol: amt} dict, a string formula, and others.

inputs

[dict] An dict of parameters associated with the entry. Defaults to None.

data

[dict] An dict of any additional data associated with the entry. Defaults to None.

entry_id

[obj] An optional id to uniquely identify the entry.

attribute

Optional attribute of the entry. This can be used to specify that the entry is a newly found compound, or to specify a particular label for the entry, or else ... Used for further analysis and plotting purposes. An attribute can be anything but must be MSONable.

Attributes

number_element

Methods

as_dict()	A JSON serializable dict representation of an object.
from_dict(<i>d</i>)	
param d	
Dict representation.	
to_json()	Returns a json string representation of the MSONable object.
unsafe_hash()	Returns an hash of the current object.
validate_monty_v1(_MSONable__input_value)	Pydantic validator with correct signature for pydantic v1.x
validate_monty_v2(_MSONable__input_value, _)	Pydantic validator with correct signature for pydantic v2.x

as_dict()

A JSON serializable dict representation of an object.

classmethod `from_dict(d)`

Parameters

d – Dict representation.

Returns

MSONable class.

property `number_element`

class `dpngen.database.VaspInput`(*incar, poscar, potcar, kpoints=None, optional_files=None, **kwargs*)

Bases: dict, MSONable

Class to contain a set of vasp input objects corresponding to a run.

Args:

`incar`: Incar object. `kpoints`: Kpoints object. `poscar`: Poscar object. `potcar`: Potcar object. `optional_files`: Other input files supplied as a dict of {
 `filename`: object}. The object should follow standard pymatgen conventions in implementing a `as_dict()` and `from_dict` method.

Methods

<code>as_dict()</code>	A JSON serializable dict representation of an object.
<code>clear()</code>	
<code>copy()</code>	
<code>from_dict(d)</code>	param d Dict representation.
<code>from_directory(input_dir[, optional_files])</code>	Read in a set of VASP input from a directory.
<code>fromkeys(iterable[, value])</code>	Create a new dictionary with keys from iterable and values set to value.
<code>get(key[, default])</code>	Return the value for key if key is in the dictionary, else default.
<code>items()</code>	
<code>keys()</code>	
<code>pop(key[, default])</code>	If the key is not found, return the default if given; otherwise, raise a KeyError.
<code>popitem(/)</code>	Remove and return a (key, value) pair as a 2-tuple.
<code>setdefault(key[, default])</code>	Insert key with a value of default if key is not in the dictionary.
<code>to_json()</code>	Returns a json string representation of the MSONable object.
<code>unsafe_hash()</code>	Returns an hash of the current object.
<code>update([E,]**F)</code>	If E is present and has a .keys() method, then does: for k in E: D[k] = E[k] If E is present and lacks a .keys() method, then does: for k, v in E: D[k] = v In either case, this is followed by: for k in F: D[k] = F[k]
<code>validate_monty_v1(_MSONable__input_value)</code>	Pydantic validator with correct signature for pydantic v1.x
<code>validate_monty_v2(_MSONable__input_value, _)</code>	Pydantic validator with correct signature for pydantic v2.x
<code>values()</code>	
<code>write_input([output_dir, ...])</code>	Write VASP input to a directory.

as_dict()

A JSON serializable dict representation of an object.

classmethod from_dict(d)**Parameters**

d – Dict representation.

Returns

MSONable class.

static from_directory(input_dir, optional_files=None)

Read in a set of VASP input from a directory. Note that only the standard INCAR, POSCAR, POTCAR and KPOINTS files are read unless optional_filenames is specified.

Parameters

input_dir

[str] Directory to read VASP input from.

optional_files

[dict] Optional files to read in as well as a dict of {filename: Object type}. Object type must have a static method from_file.

write_input(*output_dir='.', make_dir_if_not_present=True*)

Write VASP input to a directory.

Parameters**output_dir**

[str] Directory to write to. Defaults to current directory (“.”).

make_dir_if_not_present

[bool] Create the directory if not present. Defaults to True.

Submodules**dpngen.database.entry module****class** dpngen.database.entry.**Entry**(*composition, calculator, inputs, data, entry_id=None, attribute=None, tag=None*)

Bases: MSONable

An lightweight Entry object containing key computed data for storing purpose.

Parameters**composition**

[Composition] Composition of the entry. For flexibility, this can take the form of all the typical input taken by a Composition, including a {symbol: amt} dict, a string formula, and others.

inputs

[dict] An dict of parameters associated with the entry. Defaults to None.

data

[dict] An dict of any additional data associated with the entry. Defaults to None.

entry_id

[obj] An optional id to uniquely identify the entry.

attribute

Optional attribute of the entry. This can be used to specify that the entry is a newly found compound, or to specify a particular label for the entry, or else ... Used for further analysis and plotting purposes. An attribute can be anything but must be MSONable.

Attributes**number_element**

Methods

<code>as_dict()</code>	A JSON serializable dict representation of an object.
<code>from_dict(d)</code>	<p>param d Dict representation.</p>
<code>to_json()</code>	Returns a json string representation of the MSONable object.
<code>unsafe_hash()</code>	Returns an hash of the current object.
<code>validate_monty_v1(_MSONable__input_value)</code>	Pydantic validator with correct signature for pydantic v1.x
<code>validate_monty_v2(_MSONable__input_value, _)</code>	Pydantic validator with correct signature for pydantic v2.x

`as_dict()`

A JSON serializable dict representation of an object.

classmethod `from_dict(d)`

Parameters

d – Dict representation.

Returns

MSONable class.

property `number_element`

`dpngen.database.run` module

`dpngen.database.run.db_run(args)`

`dpngen.database.run.parsing_gaussian(path, output='dpngen_db.json')`

`dpngen.database.run.parsing_pwscf(path, output='dpngen_db.json')`

`dpngen.database.run.parsing_vasp(path, config_info_dict, skip_init, output='dpngen_db.json', id_prefix=None)`

`dpngen.database.vasp` module

`class dpngen.database.vasp.DPPotcar(symbols=None, functional='PBE', pp_file=None, pp_lists=None)`

Bases: MSONable

Methods

<code>as_dict()</code>	A JSON serializable dict representation of an object.
<code>from_dict(d)</code>	<p>param d Dict representation.</p>
<code>to_json()</code>	Returns a json string representation of the MSONable object.
<code>unsafe_hash()</code>	Returns an hash of the current object.
<code>validate_monty_v1(_MSONable__input_value)</code>	Pydantic validator with correct signature for pydantic v1.x
<code>validate_monty_v2(_MSONable__input_value, _)</code>	Pydantic validator with correct signature for pydantic v2.x

from_file
write_file

as_dict()

A JSON serializable dict representation of an object.

classmethod from_dict(d)

Parameters

d – Dict representation.

Returns

MSONable class.

classmethod from_file(filename)

write_file(filename)

class dpngen.database.vasp.**VaspInput**(*incar, poscar, potcar, kpoints=None, optional_files=None, **kwargs*)

Bases: dict, MSONable

Class to contain a set of vasp input objects corresponding to a run.

Args:

incar: Incar object. *kpoints*: Kpoints object. *poscar*: Poscar object. *potcar*: Potcar object. *optional_files*: Other input files supplied as a dict of {
 filename: object}. The object should follow standard pymatgen conventions in implementing a `as_dict()` and `from_dict` method.

Methods

<code>as_dict()</code>	A JSON serializable dict representation of an object.
<code>clear()</code>	
<code>copy()</code>	
<code>from_dict(d)</code>	param d Dict representation.
<code>from_directory(input_dir[, optional_files])</code>	Read in a set of VASP input from a directory.
<code>fromkeys(iterable[, value])</code>	Create a new dictionary with keys from iterable and values set to value.
<code>get(key[, default])</code>	Return the value for key if key is in the dictionary, else default.
<code>items()</code>	
<code>keys()</code>	
<code>pop(key[, default])</code>	If the key is not found, return the default if given; otherwise, raise a KeyError.
<code>popitem(/)</code>	Remove and return a (key, value) pair as a 2-tuple.
<code>setdefault(key[, default])</code>	Insert key with a value of default if key is not in the dictionary.
<code>to_json()</code>	Returns a json string representation of the MSONable object.
<code>unsafe_hash()</code>	Returns an hash of the current object.
<code>update([E,]**F)</code>	If E is present and has a .keys() method, then does: for k in E: D[k] = E[k] If E is present and lacks a .keys() method, then does: for k, v in E: D[k] = v In either case, this is followed by: for k in F: D[k] = F[k]
<code>validate_monty_v1(_MSONable__input_value)</code>	Pydantic validator with correct signature for pydantic v1.x
<code>validate_monty_v2(_MSONable__input_value, _)</code>	Pydantic validator with correct signature for pydantic v2.x
<code>values()</code>	
<code>write_input([output_dir, ...])</code>	Write VASP input to a directory.

as_dict()

A JSON serializable dict representation of an object.

classmethod from_dict(d)**Parameters**

d – Dict representation.

Returns

MSONable class.

static from_directory(input_dir, optional_files=None)

Read in a set of VASP input from a directory. Note that only the standard INCAR, POSCAR, POTCAR and KPOINTS files are read unless optional_filenames is specified.

Parameters

input_dir

[str] Directory to read VASP input from.

optional_files

[dict] Optional files to read in as well as a dict of {filename: Object type}. Object type must have a static method from_file.

write_input(*output_dir='.', make_dir_if_not_present=True*)

Write VASP input to a directory.

Parameters**output_dir**

[str] Directory to write to. Defaults to current directory (“.”).

make_dir_if_not_present

[bool] Create the directory if not present. Defaults to True.

dpgen.dispatcher package**Submodules****dpgen.dispatcher.Dispatcher module**

dpgen.dispatcher.Dispatcher.make_submission(*mdata_machine, mdata_resources, commands, work_path, run_tasks, group_size, forward_common_files, forward_files, backward_files, outlog, errlog*)

dpgen.dispatcher.Dispatcher.make_submission_compat(*machine: dict, resources: dict, commands: list[str], work_path: str, run_tasks: list[str], group_size: int, forward_common_files: list[str], forward_files: list[str], backward_files: list[str], outlog: str = 'log', errlog: str = 'err', api_version: str = '1.0'*) → None

Make submission with compatibility of both dispatcher API v0 and v1.

If *api_version* is less than 1.0, raise RuntimeError. If *api_version* is large than 1.0, use *make_submission*.

Parameters**machine**

[dict] machine dict

resources

[dict] resource dict

commands

[list[str]] list of commands

work_path

[str] working directory

run_tasks

[list[str]] list of paths to running tasks

group_size

[int] group size

forward_common_files

[list[str]] forwarded common files shared for all tasks

forward_files

[list[str]] forwarded files for each task

backward_files

[list[str]] backwarded files for each task

outlog
[str, default=log] path to log from stdout
errlog
[str, default=err] path to log from stderr
api_version
[str, default=1.0] API version. 1.0 is required

`dpngen.dispatcher.Dispatcher.mdata_arginfo()` → `list[Argument]`

This method generates arginfo for a single mdata.

A submission requires the following keys: command, machine, and resources.

Returns
`list[Argument]`
arginfo

dpngen.generator package

Subpackages

dpngen.generator.lib package

Submodules

dpngen.generator.lib.abacus_scf module

`dpngen.generator.lib.abacus_scf.get_abacus_STRU(STRU, INPUT=None, n_ele=None)`
`dpngen.generator.lib.abacus_scf.get_abacus_input_parameters(INPUT)`
`dpngen.generator.lib.abacus_scf.get_additional_from_STRU(geometry_inlines, nele)`
`dpngen.generator.lib.abacus_scf.get_mass_from_STRU(geometry_inlines, atom_names)`
`dpngen.generator.lib.abacus_scf.get_natoms_from_stru(geometry_inlines)`
`dpngen.generator.lib.abacus_scf.make_abacus_scf_input(fp_params, extra_file_path="")`
`dpngen.generator.lib.abacus_scf.make_abacus_scf_kpt(fp_params)`
`dpngen.generator.lib.abacus_scf.make_abacus_scf_stru(sys_data, fp_pp_files, fp_orb_files=None,
fp_dpks_descriptor=None, fp_params=None,
type_map=None, pporb="")`
`dpngen.generator.lib.abacus_scf.make_kspacing_kpoints_stru(stru, kspacing)`
`dpngen.generator.lib.abacus_scf.make_supercell_abacus(from_struct, super_cell)`

dpgen.generator.lib.calypso_check_outcar module**dpgen.generator.lib.calypso_run_model_devi module****dpgen.generator.lib.calypso_run_opt module****dpgen.generator.lib.cp2k module**

`dpgen.generator.lib.cp2k.iterdict(d, out_list, flag=None, indent=0)`

Doc

a recursive expansion of dictionary into cp2k input

K

current key

V

current value

D

current dictionary under expansion

Flag

used to record dictionary state. if flag is None,
it means we are in top level dict. flag is a string. :indent: intent for current section.

`dpgen.generator.lib.cp2k.make_cp2k_input(sys_data, fp_params)`

`dpgen.generator.lib.cp2k.make_cp2k_input_from_external(sys_data, exinput_path)`

`dpgen.generator.lib.cp2k.make_cp2k_xyz(sys_data)`

`dpgen.generator.lib.cp2k.update_dict(old_d, update_d)`

A method to recursive update dict :old_d: old dictionary :update_d: some update value written in dictionary form.

dpgen.generator.lib.cvasp module

`dpgen.generator.lib.cvasp.runvasp(cmd, opt=False, max_errors=3, backup=False, auto_gamma=False, auto_npar=False, ediffg=-0.05)`

Cmd example: `cmd=['mpirun', '-np', '32', '-machinefile', 'hosts', 'vasp_std']`.

dpgen.generator.lib.ele_temp module

`class dpgen.generator.lib.ele_temp.NBandsEsti(test_list)`

Bases: object

Methods

predict
save

predict(*target_dir*, *tolerance*=0.5)

save(*fname*)

dpngen.generator.lib.gaussian module

dpngen.generator.lib.gaussian.**detect_multiplicity**(*symbols*)

dpngen.generator.lib.gaussian.**make_gaussian_input**(*sys_data*, *fp_params*)

dpngen.generator.lib.gaussian.**take_cluster**(*old_conf_name*, *type_map*, *idx*, *jdata*)

dpngen.generator.lib.lammps module

dpngen.generator.lib.lammps.**get_all_dumped_forces**(*file_name*)

dpngen.generator.lib.lammps.**get_dumped_forces**(*file_name*)

dpngen.generator.lib.lammps.**make_lammps_input**(*ensemble*, *conf_file*, *graphs*, *nsteps*, *dt*, *neidelay*, *trj_freq*,
mass_map, *temp*, *jdata*, *tau_t*=0.1, *pres*=None,
tau_p=0.5, *pka_e*=None, *ele_temp_f*=None,
ele_temp_a=None, *max_seed*=1000000, *nopbc*=False,
deepmd_version='0.1', *nbeads*=None)

dpngen.generator.lib.make_calypso module

dpngen.generator.lib.make_calypso.**make_calypso_input**(*nameofatoms*, *numberofatoms*, *numberofformula*,
volume, *distanceofion*, *psoratio*, *popsiz*,
maxstep, *icode*, *split*, *vsc*, *maxnumatom*,
ctrlrange, *pstress*, *fmax*)

dpngen.generator.lib.make_calypso.**write_model_devi_out**(*devi*, *fname*)

dpngen.generator.lib.parse_calypso module

dpngen.generator.lib.pwmat module

dpngen.generator.lib.pwmat.**input_upper**(*dinput*)

dpngen.generator.lib.pwmat.**make_pwmat_input_dict**(*node1*, *node2*, *atom_config*, *ecut*, *e_error*, *rho_error*,
icmix=None, *smearing*=None, *sigma*=None,
kspacing=0.5, *flag_symm*=None)

`dpgen.generator.lib.pwmat.make_pwmat_input_user_dict(fp_params)`

`dpgen.generator.lib.pwmat.write_input_dict(input_dict)`

dpgen.generator.lib.pwscf module

`dpgen.generator.lib.pwscf.cvt_1frame(fin, fout)`

`dpgen.generator.lib.pwscf.get_atom_types(lines)`

`dpgen.generator.lib.pwscf.get_block(lines, keyword, skip=0)`

`dpgen.generator.lib.pwscf.get_cell(lines)`

`dpgen.generator.lib.pwscf.get_coords(lines)`

`dpgen.generator.lib.pwscf.get_energy(lines)`

`dpgen.generator.lib.pwscf.get_force(lines)`

`dpgen.generator.lib.pwscf.get_natoms(lines)`

`dpgen.generator.lib.pwscf.get_stress(lines, cells)`

`dpgen.generator.lib.pwscf.get_types(lines)`

`dpgen.generator.lib.pwscf.make_pwscf_01_runctrl_dict(sys_data, idict)`

`dpgen.generator.lib.pwscf.make_pwscf_input(sys_data, fp_pp_files, fp_params, user_input=True)`

dpgen.generator.lib.run_calypso module

calypso as model devi engine: 1. gen_structures 2. analysis 3. model devi.

`dpgen.generator.lib.run_calypso.analysis(iter_index, jdata, calypso_model_devi_path)`

`dpgen.generator.lib.run_calypso.gen_main(iter_index, jdata, mdata, caly_run_opt_list, gen_idx)`

`dpgen.generator.lib.run_calypso.gen_structures(iter_index, jdata, mdata, caly_run_path, current_idx, length_of_caly_runopt_list)`

`dpgen.generator.lib.run_calypso.run_calypso_model_devi(iter_index, jdata, mdata)`

dpgen.generator.lib.siesta module

`dpgen.generator.lib.siesta.make_siesta_input(sys_data, fp_pp_files, fp_params)`

dpgen.generator.lib.utils module

`dpgen.generator.lib.utils.cmd_append_log(cmd, log_file)`
`dpgen.generator.lib.utils.copy_file_list(file_list, from_path, to_path)`
`dpgen.generator.lib.utils.create_path(path)`
`dpgen.generator.lib.utils.log_iter(task, ii, jj)`
`dpgen.generator.lib.utils.log_task(message)`
`dpgen.generator.lib.utils.make_iter_name(iter_index)`
`dpgen.generator.lib.utils.record_iter(record, ii, jj)`
`dpgen.generator.lib.utils.repeat_to_length(string_to_expand, length)`
`dpgen.generator.lib.utils.replace(file_name, pattern, subst)`
`dpgen.generator.lib.utils.symlink_user_forward_files(mdata, task_type, work_path,
task_format=None)`
Symlink user-defined forward_common_files Current path should be work_path, such as 00.train.

Parameters

mdata

[dict] machine parameters

task_type

[str] task_type, such as “train”

work_path

[str] work_path, such as “iter.000001/00.train”

task_format

[dict] formats of tasks

Returns

None

dpgen.generator.lib.vasp module

`dpgen.generator.lib.vasp.incar_upper(dincar)`
`dpgen.generator.lib.vasp.make_vasp_incar_user_dict(fp_params)`
`dpgen.generator.lib.vasp.write_incar_dict(incar_dict)`

Submodules

dpgen.generator.arginfo module

`dpgen.generator.arginfo.basic_args()` → list[Argument]
`dpgen.generator.arginfo.data_args()` → list[Argument]
`dpgen.generator.arginfo.fp_args()` → list[Argument]

`dpgen.generator.arginfo.fp_style_abacus_args()` → list[Argument]

`dpgen.generator.arginfo.fp_style_amber_diff_args()` → list[Argument]

Arguments for FP style amber/diff.

Returns

list[dargs.Argument]

list of amber/diff fp style arguments

`dpgen.generator.arginfo.fp_style_cp2k_args()` → list[Argument]

`dpgen.generator.arginfo.fp_style_custom_args()` → list[Argument]

Arguments for FP style custom.

Returns

list[dargs.Argument]

list of custom fp style arguments

`dpgen.generator.arginfo.fp_style_gaussian_args()` → list[Argument]

Gaussian fp style arguments.

Returns

list[dargs.Argument]

list of Gaussian fp style arguments

`dpgen.generator.arginfo.fp_style_pwscf_args()` → list[Argument]

Arguments for FP style pwscf (Quantum Espresso).

Returns

list[dargs.Argument]

list of pwscf fp style arguments

`dpgen.generator.arginfo.fp_style_siesta_args()` → list[Argument]

`dpgen.generator.arginfo.fp_style_variant_type_args()` → Variant

`dpgen.generator.arginfo.fp_style_vasp_args()` → list[Argument]

`dpgen.generator.arginfo.model_devi_amber_args()` → list[Argument]

Amber engine arguments.

`dpgen.generator.arginfo.model_devi_args()` → list[Variant]

`dpgen.generator.arginfo.model_devi_jobs_args()` → list[Argument]

`dpgen.generator.arginfo.model_devi_jobs_rev_mat_args()` → Argument

`dpgen.generator.arginfo.model_devi_jobs_template_args()` → Argument

`dpgen.generator.arginfo.model_devi_lmp_args()` → list[Argument]

`dpgen.generator.arginfo.run_jdata_arginfo()` → Argument

Argument information for dpngen run mdata.

Returns

Argument

argument information

`dpgen.generator.arginfo.run_mdata_arginfo()` → Argument

Generate arginfo for dpngen run mdata.

Returns

Argument

arginfo

`dpgen.generator.arginfo.training_args()` → list[Argument]

Traning arguments.

Returns

list[dargs.Argument]

List of training arguments.

dpngen.generator.run module

init: data iter:

00.train 01.model_devi 02.vasp 03.data.

`dpngen.generator.run.check_bad_box(conf_name, criteria, fmt='lammps/dump')`

`dpngen.generator.run.check_cluster(conf_name, fp_cluster_vacuum, fmt='lammps/dump')`

`dpngen.generator.run.copy_model(numb_model, prv_iter_index, cur_iter_index)`

`dpngen.generator.run.detect_batch_size(batch_size, system=None)`

`dpngen.generator.run.dump_to_deepmd_raw(dump, deepmd_raw, type_map, fmt='gromacs/gro',
charge=None)`

`dpngen.generator.run.expand_idx(in_list)`

`dpngen.generator.run.expand_matrix_values(target_list, cur_idx=0)`

`dpngen.generator.run.find_only_one_key(lmp_lines, key)`

`dpngen.generator.run.gen_run(args)`

`dpngen.generator.run.get_atomic_masses(atom)`

`dpngen.generator.run.get_job_names(jdata)`

`dpngen.generator.run.get_nframes(system)`

`dpngen.generator.run.get_sys_index(task)`

`dpngen.generator.run.make_fp(iter_index, jdata, mdata)`

Select the candidate strutures and make the input file of FP calculation.

Parameters

iter_index

[int] iter index

jdata

[dict] Run parameters.

mdata

[dict] Machine parameters.

`dpngen.generator.run.make_fp_abacus_scf(iter_index, jdata)`

`dpngen.generator.run.make_fp_amber_diff(iter_index: int, jdata: dict)`

Run amber twice to calculate high-level and low-level potential, and then generate difference between them.

Besides AMBER, one needs to install *dpamber* package, which is avaiable at <https://github.com/njzjz/dpamber>

Currently, it should be used with the AMBER model_devi driver.

Parameters

iter_index

[int] iter index

jdata

[dict]

Run parameters. The following parameters are used in this method:

mdin_prefix

[str] The path prefix to AMBER mdin files

qm_region

[list[str]] AMBER mask of the QM region. Each mask maps to a system.

qm_charge

[list[int]] Charge of the QM region. Each charge maps to a system.

high_level

[str] high level method

low_level

[str] low level method

fp_params

[dict]

This parameters includes:

high_level_mdin

[str] High-level AMBER mdin file. %qm_theory%, %qm_region%, and %qm_charge% will be replace.

low_level_mdin

[str] Low-level AMBER mdin file. %qm_theory%, %qm_region%, and %qm_charge% will be replace.

parm7_prefix

[str] The path prefix to AMBER PARM7 files

parm7

[list[str]] List of paths to AMBER PARM7 files. Each file maps to a system.

References

[1]

`dpngen.generator.run.make_fp_calculation(iter_index, jdata, mdata)`

Make the input file of FP calculation.

Parameters

iter_index

[int] iter index

jdata

[dict] Run parameters.

mdata

[dict] Machine parameters.

`dpngen.generator.run.make_fp_cp2k(iter_index, jdata)`

`dpngen.generator.run.make_fp_custom(iter_index, jdata)`

Make input file for customized FP style.

Convert the POSCAR file to custom format.

Parameters**iter_index**

[int] iter index

jdata

[dict] Run parameters.

```
dpngen.generator.run.make_fp_gaussian(iter_index, jdata)
dpngen.generator.run.make_fp_pwmat(iter_index, jdata)
dpngen.generator.run.make_fp_pwscf(iter_index, jdata)
dpngen.generator.run.make_fp_siesta(iter_index, jdata)
dpngen.generator.run.make_fp_task_name(sys_idx, counter)
dpngen.generator.run.make_fp_vasp(iter_index, jdata)
dpngen.generator.run.make_fp_vasp_cp_cvasp(iter_index, jdata)
dpngen.generator.run.make_fp_vasp_incar(iter_index, jdata, nbands_esti=None)
dpngen.generator.run.make_fp_vasp_kp(iter_index, jdata)
dpngen.generator.run.make_model_devi(iter_index, jdata, mdata)
dpngen.generator.run.make_model_devi_conf_name(sys_idx, conf_idx)
dpngen.generator.run.make_model_devi_task_name(sys_idx, task_idx)
dpngen.generator.run.make_pwmat_input(jdata, filename)
dpngen.generator.run.make_train(iter_index, jdata, mdata)
dpngen.generator.run.make_vasp_incar(jdata, filename)
dpngen.generator.run.make_vasp_incar_ele_temp(jdata, filename, ele_temp, nbands_esti=None)
dpngen.generator.run.parse_cur_job(cur_job)
dpngen.generator.run.parse_cur_job_revmat(cur_job, use_plm=False)
dpngen.generator.run.parse_cur_job_sys_revmat(cur_job, sys_idx, use_plm=False)
dpngen.generator.run.poscar_natoms(lines)
dpngen.generator.run.poscar_shuffle(poscar_in, poscar_out)
dpngen.generator.run.poscar_to_conf(poscar, conf)
dpngen.generator.run.post_fp(iter_index, jdata)
dpngen.generator.run.post_fp_abacus_scf(iter_index, jdata)
dpngen.generator.run.post_fp_amber_diff(iter_index, jdata)
dpngen.generator.run.post_fp_check_fail(iter_index, jdata, rfailed=None)
dpngen.generator.run.post_fp_cp2k(iter_index, jdata, rfailed=None)
```

`dpgen.generator.run.post_fp_custom(iter_index, jdata)`

Post fp for custom fp. Collect data from user-defined *output_fn*.

Parameters

iter_index

[int] The index of the current iteration.

jdata

[dict] The parameter data.

`dpgen.generator.run.post_fp_gaussian(iter_index, jdata)`

`dpgen.generator.run.post_fp_pwmat(iter_index, jdata, rfailed=None)`

`dpgen.generator.run.post_fp_pwscf(iter_index, jdata)`

`dpgen.generator.run.post_fp_siesta(iter_index, jdata)`

`dpgen.generator.run.post_fp_vasp(iter_index, jdata, rfailed=None)`

`dpgen.generator.run.post_model_devi(iter_index, jdata, mdata)`

`dpgen.generator.run.post_train(iter_index, jdata, mdata)`

`dpgen.generator.run.revise_by_keys(lmp_lines, keys, values)`

`dpgen.generator.run.revise_lmp_input_dump(lmp_lines, trj_freq, model_devi_merge_traj=False)`

`dpgen.generator.run.revise_lmp_input_model(lmp_lines, task_model_list, trj_freq, deepmd_version='1')`

`dpgen.generator.run.revise_lmp_input_plm(lmp_lines, in_plm, out_plm='output.plumed')`

`dpgen.generator.run.run_fp(iter_index, jdata, mdata)`

`dpgen.generator.run.run_fp_inner(iter_index, jdata, mdata, forward_files, backward_files, check_fn, log_file='fp.log', forward_common_files=[])`

`dpgen.generator.run.run_iter(param_file, machine_file)`

`dpgen.generator.run.run_md_model_devi(iter_index, jdata, mdata)`

`dpgen.generator.run.run_model_devi(iter_index, jdata, mdata)`

`dpgen.generator.run.run_train(iter_index, jdata, mdata)`

`dpgen.generator.run.set_version(mdata)`

`dpgen.generator.run.sys_link_fp_vasp_pp(iter_index, jdata)`

`dpgen.generator.run.update_mass_map(jdata)`

dpgen.remote package

Submodules

dpgen.remote.decide_machine module

`dpgen.remote.decide_machine.convert_mdata(mdata, task_types=['train', 'model_devi', 'fp'])`

Convert mdata for DP-GEN main process. New conversion is like `mdata["fp"]["machine"]`, DP-GEN needs `mdata["fp_machine"]`.

Notice that we deprecate the function which can automatically select one most available machine, since this function was only used by Angus, and only supports for Slurm. In the future this can be implemented.

Parameters

mdata

[dict] Machine parameters to be converted.

task_types

[list of string] Type of tasks, default is ["train", "model_devi", "fp"]

Returns

dict

mdata converted

dpgen.simplify package

Submodules

dpgen.simplify.arginfo module

`dpgen.simplify.arginfo.fp_args()` → list[Argument]

Generate arginfo for fp.

Returns

List[Argument]

arginfo

`dpgen.simplify.arginfo.fp_style_variant_type_args()` → Variant

Generate variant for fp style variant type.

Returns

Variant

variant for fp style

`dpgen.simplify.arginfo.general_simplify_arginfo()` → Argument

General simplify arginfo.

Returns

Argument

arginfo

`dpgen.simplify.arginfo.simplify_jdata_arginfo()` → Argument

Generate arginfo for dpgen simplify jdata.

Returns

Argument

arginfo

`dpgen.simplify.arginfo.simplify_mdata_arginfo()` → Argument

Generate arginfo for dpgen simplify mdata.

Returns

Argument

arginfo

dpngen.simplify.simplify module

Simplify dataset (minimize the dataset size).

Init: pick up init data from dataset randomly

Iter: 00: train models (same as generator) 01: calculate model deviations of the rest dataset, pick up data with proper model deviation 02: fp (optional, if the original dataset do not have fp data, same as generator)

`dpngen.simplify.simplify.gen_simplify(args)`

`dpngen.simplify.simplify.get_multi_system(path: str | list[str], jdata: dict) → MultiSystems`

Get MultiSystems from a path or list of paths.

Both NumPy and HDF5 formats are supported. For details of two formats, refer to DeePMD-kit documentation.

If *labeled* in *jdata* is True, returns MultiSystems with LabeledSystem. Otherwise, returns MultiSystems with System.

Parameters

path

[str or list of str] path or list of paths to the dataset

jdata

[dict] parameters which may contain *labeled* key

Returns

dpdata.MultiSystems

MultiSystems with LabeledSystem or System

`dpngen.simplify.simplify.get_system_cls(jdata)`

`dpngen.simplify.simplify.init_model(iter_index, jdata, mdata)`

`dpngen.simplify.simplify.init_pick(iter_index, jdata, mdata)`

Pick up init data from dataset randomly.

`dpngen.simplify.simplify.make_fp(iter_index, jdata, mdata)`

`dpngen.simplify.simplify.make_fp_configs(iter_index, jdata)`

`dpngen.simplify.simplify.make_fp_labeled(iter_index, jdata)`

`dpngen.simplify.simplify.make_model_devi(iter_index, jdata, mdata)`

Calculate the model deviation of the rest idx.

`dpngen.simplify.simplify.post_model_devi(iter_index, jdata, mdata)`

Calculate the model deviation.

`dpngen.simplify.simplify.run_iter(param_file, machine_file)`

Init (iter 0): init_pick.

tasks (iter > 0): 00 make_train (same as generator) 01 run_train (same as generator) 02 post_train (same as generator) 03 make_model_devi 04 run_model_devi 05 post_model_devi 06 make_fp 07 run_fp (same as generator) 08 post_fp (same as generator)

`dpngen.simplify.simplify.run_model_devi(iter_index, jdata, mdata)`

Submit dp test tasks.

dpngen.tools package

Submodules

dpngen.tools.auto_gen_param module

```
class dpngen.tools.auto_gen_param.Iteration(temps, nsteps_list=[500, 500, 1000, 1000, 3000, 3000, 6000, 6000], sub_iteration_num=8, ensemble='npt', press=[1.0, 10.0, 100.0, 1000.0, 5000.0, 10000.0, 20000.0, 50000.0], trj_freq=10)
```

Bases: object

Attributes

`index_iteration`

Methods

<code>gen_sub_iter</code>
<code>register_iteration</code>
<code>register_sub_iteartion</code>

```
current_num_of_itearation = 0
```

```
current_num_of_sub_itearation = 0
```

```
gen_sub_iter(system_list)
```

```
property index_iteration
```

```
classmethod register_iteration()
```

```
classmethod register_sub_iteartion()
```

```
class dpngen.tools.auto_gen_param.System(system_prefix="")
```

Bases: object

Attributes

`index_system`

Methods

<code>add_sub_system</code>
<code>get_sub_system</code>
<code>register_sub_system</code>
<code>register_system</code>

```
add_sub_system(idx2, files_list)
```

```
current_num_of_sub_systems = 0
```

```
current_num_of_system = 0
```



```

    get_sub_system()

    property index_system

    classmethod register_sub_system()

    classmethod register_system()

dpgen.tools.auto_gen_param.auto_gen_param(args)

dpgen.tools.auto_gen_param.default_map_generator(map_list=[1, 1, 2, 2, 2, 4, 4, 4], data_list=None)

dpgen.tools.auto_gen_param.default_temps_generator(melt_point, temps_interval=0.1, num_temps=5)

dpgen.tools.auto_gen_param.get_basic_param_json(melt_point, out_param_filename='param_basic.json',
                                                  scan_dir='./', file_name='POSCAR',
                                                  init_file_name='type.raw', min_allow_files_num=16,
                                                  map_list=[1, 1, 2, 2, 2, 4, 4, 4], meta_iter_num=4,
                                                  sub_iteration_num=8, map_iterator=None,
                                                  nsteps_list=[500, 500, 1000, 1000, 3000, 3000, 6000,
                                                                6000], press=[1.0, 10.0, 100.0, 1000.0, 5000.0,
                                                                10000.0, 20000.0, 50000.0], temps_iterator=None,
                                                  ensemble='npt', trj_freq=10, temps_interval=0.1,
                                                  num_temps=5)

dpgen.tools.auto_gen_param.get_init_data_sys(scan_dir='./', init_file_name='type.raw')

dpgen.tools.auto_gen_param.get_model_devi_jobs(melt_point, system_list, nsteps_list=[500, 500, 1000,
                                                                                   1000, 3000, 3000, 6000, 6000],
                                                                                   press=[1.0, 10.0,
                                                                                   100.0, 1000.0, 5000.0, 10000.0,
                                                                                   20000.0, 50000.0],
                                                                                   meta_iter_num=4, sub_iteration_num=8,
                                                                                   temps_iterator=None, ensemble='npt',
                                                                                   trj_freq=10, temps_interval=0.1,
                                                                                   num_temps=5)

dpgen.tools.auto_gen_param.get_sys_configs(system_list)

dpgen.tools.auto_gen_param.get_system_list(system_dict, map_list=[1, 1, 2, 2, 2, 4, 4, 4],
                                           meta_iter_num=4, sub_iteration_num=8,
                                           map_iterator=None, file_name='POSCAR')

:Exmaple [['000000', '000001'], ['00000[2-9]'], ['00001?', '000020'],]

dpgen.tools.auto_gen_param.scan_files(scan_dir='./', file_name='POSCAR', min_allow_files_num=20)

```

dpgen.tools.collect_data module

```

dpgen.tools.collect_data.collect_data(target_folder, param_file, output, verbose=True)

dpgen.tools.collect_data.file_len(fname)

```

dpgen.tools.relabel module

```
dpngen.tools.relabel.copy_pp_files(tdir, fp_pp_path, fp_pp_files)
dpngen.tools.relabel.create_init_tasks(target_folder, param_file, output, fp_json, verbose=True)
dpngen.tools.relabel.create_tasks(target_folder, param_file, output, fp_json, verbose=True, numb_iter=-1)
dpngen.tools.relabel.get_lmp_info(input_file)
dpngen.tools.relabel.link_pp_files(tdir, fp_pp_path, fp_pp_files)
dpngen.tools.relabel.make_pwscf(tdir, fp_params, mass_map, fp_pp_path, fp_pp_files, user_input)
dpngen.tools.relabel.make_siesta(tdir, fp_params, fp_pp_path, fp_pp_files)
dpngen.tools.relabel.make_vasp(tdir, fp_params)
dpngen.tools.relabel.make_vasp_incar(tdir, fp_incar)
```

dpngen.tools.run_report module

```
dpngen.tools.run_report.run_report(args)
```

dpngen.tools.stat_iter module

```
dpngen.tools.stat_iter.stat_iter(target_folder, param_file='param.json', verbose=True, mute=False)
```

dpngen.tools.stat_sys module

```
dpngen.tools.stat_sys.ascii_hist(count)
dpngen.tools.stat_sys.run_report(args)
dpngen.tools.stat_sys.stat_sys(target_folder, param_file='param.json', verbose=True, mute=False)
```

dpngen.tools.stat_time module

```
dpngen.tools.stat_time.stat_time(target_folder, param_file='param.json', verbose=True, mute=False)
```

10.1.2 Submodules

10.1.3 dpngen.arginfo module

```
dpngen.arginfo.general_mdata_arginfo(name: str, tasks: tuple[str]) → Argument
```

Generate arginfo for general mdata.

Parameters

name

[str] mdata name

tasks
 [tuple[str]] tuple of task keys, e.g. (“train”, “model_devi”, “fp”)
Returns
Argument
 arginfo

10.1.4 dpngen.gui module

DP-GUI entrypoint.

`dpngen.gui.start_dpgui(args: Namespace)`

Host DP-GUI server.

Parameters

args

[argparse.Namespace] Arguments from argparse.

Raises

ModuleNotFoundError

The dpgui package is not installed

10.1.5 dpngen.main module

`dpngen.main.main()`

`dpngen.main.main_parser()` → ArgumentParser

Returns parser for *dpngen* command.

Returns

argparse.ArgumentParser

parser for *dpngen* command

10.1.6 dpngen.util module

`dpngen.util.box_center(ch='', fill=' ', sp='|')`

Put the string at the center of | |.

`dpngen.util.convert_training_data_to_hdf5(input_files: list[str], h5_file: str)`

Convert training data to HDF5 format and update the input files.

Parameters

input_files

[list of str] DeePMD-kit input file names

h5_file

[str] HDF5 file name

`dpngen.util.expand_sys_str(root_dir: str | Path) → list[str]`

Recursively iterate over directories taking those that contain *type.raw* file.

If root_dir is a file but not a directory, it will be assumed as an HDF5 file.

Parameters

root_dir

[Union[str, Path]] starting directory

Returns

List[str]

list of string pointing to system directories

Raises**RuntimeError**

No system was found in the directory

`dpgen.util.load_file(filename: str | PathLike) → dict`

Load data from a JSON or YAML file.

Parameters**filename**

[str or os.PathLike] The filename to load data from, whose suffix should be .json, .yaml, or .yaml

Returns**dict**

The data loaded from the file

Raises**ValueError**

If the file format is not supported

`dpgen.util.normalize(arginfo: Argument, data: dict, strict_check: bool = True) → dict`

Normalize and check input data.

Parameters**arginfo**

[dargs.Argument] argument information

data

[dict] input data

strict_check

[bool, default=True] strict check data or not

Returns**dict**

normalized data

`dpgen.util.sepline(ch='-', sp='-', screen=False)`

Seperate the output by ‘-‘.

`dpgen.util.set_directory(path: Path)`

Sets the current working path within the context.

Parameters**path**

[Path] The path to the cwd

Yields**None**

Examples

```
>>> with set_directory("some_path"):
...     do_something()
```

`dpgen.util.setup_ele_temp(atomic: bool)`

Set electronic temperature as required input data.

Parameters**atomic**

[bool] Whether to use atomic temperature or frame temperature

AUTHORS

- A bot of @njzjz
- AnguseZhang
- Anopaul
- BaozCWJ
- Chenqqian Zhang
- Cloudac7
- DULinhan
- EC2 Default User
- Ericwang6
- Futaki Haduki
- Futaki Hatuki
- Han Wang
- HuangJiameng
- Jinzhe Zeng
- Kick-H
- Levi Zhou
- LiangWenshuo1118
- Liu Renxi
- Liu-RX
- LiuGroupHNU
- Manyi Yang
- Pan Xiang
- Peng Xingliang
- Pinchen Xie
- Sian Chen
- Silvia-liu
- TaipingHu

- Tongqi Wen
- TongqiWen
- TrellixVulnTeam
- Waikit Chan
- Wanrun Jiang
- Yifan Li
- Yingze Wang
- Yixiao Chen
- Yoh
- Yongbin Zhuang
- Yuan Fengbo
- Yuan Fengbo ()
- Yunfan Xu
- Yunpei Liu
- Yuzhi Zhang
- Zhiwei Zhang
- Zhuoyuan
- baihuyu12
- cherushui
- cyFortneu
- deepmodeling
- dependabot[bot]
- dingzhaohan
- dinngzhaohan
- felix5572
- fqgong
- haidi
- hongriTianqi
- jameswind
- kiwi
- likefallwind
- pee8379
- pre-commit-ci[bot]
- pxxingliang
- robinzhuang
- robinzyb

- root
- shazj99
- tianhongzhen
- tuoping
- unknown
- yuzhi
- zhang yuzhi
- zhangbei07
- zhaohan
- zhengming-HIT
- zhenyu
- ziqi-hu
-
- genindex
- modindex
- search

BIBLIOGRAPHY

- [1] Development of Range-Corrected Deep Learning Potentials for Fast, Accurate Quantum Mechanical/Molecular Mechanical Simulations of Chemical Reactions in Solution, Jinzhe Zeng, Timothy J. Giese, Şölen Ekesan, and Darrin M. York, *Journal of Chemical Theory and Computation* 2021 17 (11), 6993-7009

PYTHON MODULE INDEX

d

dpngen, 251
dpngen.arginfo, 302
dpngen.auto_test, 251
dpngen.auto_test.ABACUS, 259
dpngen.auto_test.calculator, 271
dpngen.auto_test.common_equi, 272
dpngen.auto_test.common_prop, 272
dpngen.auto_test.Elastic, 261
dpngen.auto_test.EOS, 261
dpngen.auto_test.Gamma, 263
dpngen.auto_test.gen_confs, 272
dpngen.auto_test.Interstitial, 264
dpngen.auto_test.Lammps, 264
dpngen.auto_test.lib, 251
dpngen.auto_test.lib.abacus, 251
dpngen.auto_test.lib.crys, 252
dpngen.auto_test.lib.lammps, 252
dpngen.auto_test.lib.lmp, 253
dpngen.auto_test.lib.mfp_eosfit, 253
dpngen.auto_test.lib.pwscf, 257
dpngen.auto_test.lib.siesta, 258
dpngen.auto_test.lib.util, 258
dpngen.auto_test.lib.utils, 258
dpngen.auto_test.lib.vasp, 258
dpngen.auto_test.mpdb, 272
dpngen.auto_test.Property, 266
dpngen.auto_test.refine, 272
dpngen.auto_test.reproduce, 273
dpngen.auto_test.run, 273
dpngen.auto_test.Surface, 267
dpngen.auto_test.Task, 268
dpngen.auto_test.Vacancy, 271
dpngen.auto_test.VASP, 269
dpngen.collect, 273
dpngen.collect.collect, 273
dpngen.data, 273
dpngen.data.arginfo, 276
dpngen.data.gen, 277
dpngen.data.reaction, 278
dpngen.data.surf, 278
dpngen.data.tools, 273
dpngen.data.tools.bcc, 273
dpngen.data.tools.cessp2force_lin, 274
dpngen.data.tools.create_random_disturb, 274
dpngen.data.tools.diamond, 274
dpngen.data.tools.fcc, 275
dpngen.data.tools.hcp, 275
dpngen.data.tools.io_lammps, 275
dpngen.data.tools.sc, 276
dpngen.database, 279
dpngen.database.entry, 283
dpngen.database.run, 284
dpngen.database.vasp, 284
dpngen.dispatcher, 287
dpngen.dispatcher.Dispatcher, 287
dpngen.generator, 288
dpngen.generator.arginfo, 292
dpngen.generator.lib, 288
dpngen.generator.lib.abacus_scf, 288
dpngen.generator.lib.cp2k, 289
dpngen.generator.lib.cvasp, 289
dpngen.generator.lib.ele_temp, 289
dpngen.generator.lib.gaussian, 290
dpngen.generator.lib.lammps, 290
dpngen.generator.lib.make_calypso, 290
dpngen.generator.lib.parse_calypso, 290
dpngen.generator.lib.pwmat, 290
dpngen.generator.lib.pwscf, 291
dpngen.generator.lib.run_calypso, 291
dpngen.generator.lib.siesta, 291
dpngen.generator.lib.utils, 292
dpngen.generator.lib.vasp, 292
dpngen.generator.run, 294
dpngen.gui, 303
dpngen.main, 303
dpngen.remote, 297
dpngen.remote.decide_machine, 297
dpngen.simplify, 298
dpngen.simplify.arginfo, 298
dpngen.simplify.simplify, 299
dpngen.tools, 300
dpngen.tools.auto_gen_param, 300
dpngen.tools.collect_data, 301

`dpgen.tools.relabel`, [302](#)
`dpgen.tools.run_report`, [302](#)
`dpgen.tools.stat_iter`, [302](#)
`dpgen.tools.stat_sys`, [302](#)
`dpgen.tools.stat_time`, [302](#)
`dpgen.util`, [303](#)

A

ABACUS (class in *dpgen.auto_test.ABACUS*), 259
 add_sub_system() (*dp-gen.tools.auto_gen_param.System* method), 300
 analysis() (in module *dp-gen.generator.lib.run_calypso*), 291
 api_version:
 init_bulk_mdata/api_version (Argument), 81
 init_reaction_mdata/api_version (Argument), 114
 init_surf_mdata/api_version (Argument), 99
 run_mdata/api_version (Argument), 40
 simplify_mdata/api_version (Argument), 168
 append_script:
 init_bulk_mdata/fp/resources/append_script (Argument), 89
 init_reaction_mdata/build/resources/append_script (Argument), 135
 init_reaction_mdata/fp/resources/append_script (Argument), 148
 init_reaction_mdata/reaxff/resources/append_script (Argument), 123
 init_surf_mdata/fp/resources/append_script (Argument), 107
 run_mdata/fp/resources/append_script (Argument), 72
 run_mdata/model_devi/resources/append_script (Argument), 60
 run_mdata/train/resources/append_script (Argument), 48
 simplify_mdata/fp/resources/append_script (Argument), 201
 simplify_mdata/model_devi/resources/append_script (Argument), 189
 simplify_mdata/train/resources/append_script (Argument), 177
 apply_type_map() (in module *dp-gen.auto_test.lib.lammps*), 252
 as_dict() (*dpgen.database.DPPotcar* method), 279
 as_dict() (*dpgen.database.Entry* method), 280
 as_dict() (*dpgen.database.entry.Entry* method), 284

as_dict() (*dpgen.database.vasp.DPPotcar* method), 285
 as_dict() (*dpgen.database.vasp.VaspInput* method), 286
 as_dict() (*dpgen.database.VaspInput* method), 282
 ascii_hist() (in module *dpgen.tools.stat_sys*), 302
 ase2lammpsdata() (in module *dp-gen.data.tools.io_lammps*), 275
 atom_masses:
 init_bulk_jdata[ABACUS]/atom_masses (Argument), 81
 auto_gen_param() (in module *dp-gen.tools.auto_gen_param*), 301

B

backward_files (*dpgen.auto_test.Task.Task* property), 268
 backward_files() (*dpgen.auto_test.ABACUS.ABACUS* method), 259
 backward_files() (*dpgen.auto_test.Lammps.Lammps* method), 265
 backward_files() (*dpgen.auto_test.VASP.VASP* method), 269
 basic_args() (in module *dpgen.generator.arginfo*), 292
 basis_set:
 run_jdata[fp_style=gaussian]/fp_params/basis_set (Argument), 34
 simplify_jdata[gaussian]/fp_params/basis_set (Argument), 163
 batch_type:
 init_bulk_mdata/fp/machine/batch_type (Argument), 81
 init_bulk_mdata/fp/resources/batch_type (Argument), 89
 init_reaction_mdata/build/machine/batch_type (Argument), 127
 init_reaction_mdata/build/resources/batch_type (Argument), 136
 init_reaction_mdata/fp/machine/batch_type (Argument), 140
 init_reaction_mdata/fp/resources/batch_type (Argument), 148

init_reaction_mdata/reaxff/machine/batch_type calc_props_morse_6p() (in module dp-
 (Argument), 115 gen.auto_test.lib.mfp_eosfit), 254
 init_reaction_mdata/reaxff/resources/batch_type calc_props_SJX_5p() (in module dp-
 (Argument), 123 gen.auto_test.lib.mfp_eosfit), 254
 init_surf_mdata/fp/machine/batch_type calc_props_vinet() (in module dp-
 (Argument), 100 gen.auto_test.lib.mfp_eosfit), 254
 init_surf_mdata/fp/resources/batch_type calc_v0_mBM4poly() (in module dp-
 (Argument), 108 gen.auto_test.lib.mfp_eosfit), 254
 run_mdata/fp/machine/batch_type (Argument), 64 calc_v0_mBM5poly() (in module dp-
 run_mdata/fp/resources/batch_type (Argument), 72 car2dir() (in module dpgen.data.tools.io_lammps), 275
 run_mdata/model_devi/machine/batch_type cell_type:
 (Argument), 52 init_bulk_jdata/cell_type (Argument), 78
 run_mdata/model_devi/resources/batch_type (Argument), 61 init_surf_jdata/cell_type (Argument), 96
 run_mdata/train/machine/batch_type (Argument), 41 centralize_slab() (dpgen.auto_test.Gamma.Gamma
 run_mdata/train/resources/batch_type (Argument), 48 static method), 263
 simplify_mdata/fp/machine/batch_type (Argument), 194 charge:
 simplify_mdata/fp/resources/batch_type (Argument), 201 run_jdata[fp_style=gaussian]/fp_params/charge
 simplify_mdata/model_devi/machine/batch_type (Argument), 181 (Argument), 34
 simplify_mdata/model_devi/resources/batch_type (Argument), 190 simplify_jdata[gaussian]/fp_params/charge
 simplify_mdata/train/machine/batch_type (Argument), 169 (Argument), 163
 simplify_mdata/train/resources/batch_type (Argument), 177 check_apikey() (in module dpgen.auto_test.mpdb), 272
 bcc() (in module dpgen.auto_test.lib.crys), 252 check_bad_box() (in module dpgen.generator.run), 294
 birch() (in module dpgen.auto_test.lib.mfp_eosfit), 254 check_cluster() (in module dpgen.generator.run), 294
 BM4() (in module dpgen.auto_test.lib.mfp_eosfit), 253 check_finished() (in module dp-
 BM5() (in module dpgen.auto_test.lib.mfp_eosfit), 253 gen.auto_test.lib.abacus), 251
 box2lmpbox() (in module dpgen.auto_test.lib.lmp), 253 check_finished() (in module dp-
 box_center() (in module dpgen.util), 303 gen.auto_test.lib.lammps), 252
 build: check_finished_new() (in module dp-
 init_reaction_mdata/build (Argument), 127 gen.auto_test.lib.lammps), 252
 check_stru_fixed() (in module dp-
 gen.auto_test.lib.abacus), 251
 class_cell_type() (in module dpgen.data.gen), 277
 class_cell_type() (in module dpgen.data.surf), 278
 clean_asynchronously:
 init_bulk_mdata/fp/machine/clean_asynchronously
 (Argument), 82
 init_reaction_mdata/build/machine/clean_asynchronously
 (Argument), 128
 init_reaction_mdata/fp/machine/clean_asynchronously
 (Argument), 140
 init_reaction_mdata/reaxff/machine/clean_asynchronously
 (Argument), 115
 init_surf_mdata/fp/machine/clean_asynchronously
 (Argument), 100
 run_mdata/fp/machine/clean_asynchronously
 (Argument), 65
 run_mdata/model_devi/machine/clean_asynchronously
 (Argument), 53
 run_mdata/train/machine/clean_asynchronously
 (Argument), 41
 simplify_mdata/fp/machine/clean_asynchronously

C
 calc_props_BM4() (in module dp-
 gen.auto_test.lib.mfp_eosfit), 254
 calc_props_LOG4() (in module dp-
 gen.auto_test.lib.mfp_eosfit), 254
 calc_props_mBM4() (in module dp-
 gen.auto_test.lib.mfp_eosfit), 254
 calc_props_mBM4poly() (in module dp-
 gen.auto_test.lib.mfp_eosfit), 254
 calc_props_mBM5poly() (in module dp-
 gen.auto_test.lib.mfp_eosfit), 254
 calc_props_morse() (in module dp-
 gen.auto_test.lib.mfp_eosfit), 254

- (Argument), 194
- simplify_mdata/model_devi/machine/clean_asyncronously (dpgen.auto_test.ABACUS.ABACUS method), 259
- (Argument), 181
- simplify_mdata/train/machine/clean_asyncronously() (dpgen.auto_test.Lammps.Lammps method), 265
- (Argument), 169
- cluster_cutoff:
- run_jdata[fp_style=gaussian]/cluster_cutoff compute() (dpgen.auto_test.Property.Property method), 266
- (Argument), 33
- run_jdata[fp_style=siesta]/cluster_cutoff compute() (dpgen.auto_test.Task.Task method), 268
- (Argument), 34
- compute() (dpgen.auto_test.VASP.VASP method), 270
- simplify_jdata[gaussian]/cluster_cutoff context_type:
- (Argument), 162
- init_bulk_mdata/fp/machine/context_type
- simplify_jdata[siesta]/cluster_cutoff (Argument), 82
- (Argument), 164
- init_reaction_mdata/build/machine/context_type
- (Argument), 128
- cluster_cutoff_hard:
- run_jdata[fp_style=gaussian]/cluster_cutoff_hard (Argument), 140
- (Argument), 33
- init_reaction_mdata/fp/machine/context_type
- simplify_jdata[gaussian]/cluster_cutoff_hard (Argument), 115
- (Argument), 162
- init_surf_mdata/fp/machine/context_type
- (Argument), 100
- cluster_minify:
- run_jdata[fp_style=gaussian]/cluster_minify run_mdata/fp/machine/context_type (Argument), 65
- (Argument), 33
- simplify_jdata[gaussian]/cluster_minify run_mdata/model_devi/machine/context_type (Argument), 53
- (Argument), 162
- cmd_append_log() (in module dpgen.auto_test.lib.utils), 258
- cmd_append_log() (in module dpgen.generator.lib.utils), 292
- coll_abacus_md() (in module dpgen.data.gen), 277
- coll_ndata:
- init_bulk_jdata/coll_ndata (Argument), 80
- init_surf_jdata/coll_ndata (Argument), 99
- coll_vasp_md() (in module dpgen.data.gen), 277
- collect_data() (in module dpgen.collect.collect), 273
- collect_data() (in module dpgen.tools.collect_data), 301
- collect_task() (in module dpgen.auto_test.lib.util), 258
- command:
- init_bulk_mdata/fp/command (Argument), 81
- init_reaction_mdata/build/command (Argument), 127
- init_reaction_mdata/fp/command (Argument), 139
- init_reaction_mdata/reaxff/command (Argument), 114
- init_surf_mdata/fp/command (Argument), 99
- run_mdata/fp/command (Argument), 64
- run_mdata/model_devi/command (Argument), 52
- run_mdata/train/command (Argument), 40
- simplify_mdata/fp/command (Argument), 193
- simplify_mdata/model_devi/command (Argument), 181
- simplify_mdata/train/command (Argument), 169
- compute() (dpgen.auto_test.ABACUS.ABACUS method), 259
- compute() (dpgen.auto_test.Lammps.Lammps method), 265
- compute() (dpgen.auto_test.Property.Property method), 266
- compute() (dpgen.auto_test.Task.Task method), 268
- compute() (dpgen.auto_test.VASP.VASP method), 270
- context_type:
- init_bulk_mdata/fp/machine/context_type (Argument), 82
- init_reaction_mdata/build/machine/context_type (Argument), 128
- init_reaction_mdata/fp/machine/context_type (Argument), 140
- init_reaction_mdata/reaxff/machine/context_type (Argument), 115
- init_surf_mdata/fp/machine/context_type (Argument), 100
- run_mdata/fp/machine/context_type (Argument), 65
- run_mdata/model_devi/machine/context_type (Argument), 53
- run_mdata/train/machine/context_type (Argument), 41
- simplify_mdata/fp/machine/context_type (Argument), 194
- simplify_mdata/model_devi/machine/context_type (Argument), 182
- simplify_mdata/train/machine/context_type (Argument), 170
- control:
- init_reaction_jdata/reaxff/control (Argument), 113
- convert_cell() (in module dpgen.data.tools.io_lammps), 275
- convert_data() (in module dpgen.data.reaction), 278
- convert_forces() (in module dpgen.data.tools.io_lammps), 275
- convert_mdata() (in module dpgen.remote.decide_machine), 297
- convert_positions() (in module dpgen.data.tools.io_lammps), 275
- convert_stress() (in module dpgen.data.tools.io_lammps), 275
- convert_training_data_to_hdf5() (in module dpgen.util), 303
- copy_file_list() (in module dpgen.auto_test.lib.utils), 258
- copy_file_list() (in module dpgen.generator.lib.utils), 292
- copy_model() (in module dpgen.generator.run), 294
- copy_pp_files() (in module dpgen.tools.relabel), 302

<p>cpu_per_node:</p> <p><code>init_bulk_mdata/fp/resources/cpu_per_node</code> (Argument), 87</p> <p><code>init_reaction_mdata/build/resources/cpu_per_node</code> (Argument), 133</p> <p><code>init_reaction_mdata/fp/resources/cpu_per_node</code> (Argument), 145</p> <p><code>init_reaction_mdata/reaxff/resources/cpu_per_node</code> (Argument), 120</p> <p><code>init_surf_mdata/fp/resources/cpu_per_node</code> (Argument), 105</p> <p><code>run_mdata/fp/resources/cpu_per_node</code> (Argument), 70</p> <p><code>run_mdata/model_devi/resources/cpu_per_node</code> (Argument), 58</p> <p><code>run_mdata/train/resources/cpu_per_node</code> (Argument), 46</p> <p><code>simplify_mdata/fp/resources/cpu_per_node</code> (Argument), 199</p> <p><code>simplify_mdata/model_devi/resources/cpu_per_node</code> (Argument), 187</p> <p><code>simplify_mdata/train/resources/cpu_per_node</code> (Argument), 174</p> <p><code>create_disturbs_abacus_dev()</code> (in module <code>dp-gen.data.tools.create_random_disturb</code>), 274</p> <p><code>create_disturbs_ase()</code> (in module <code>dp-gen.data.tools.create_random_disturb</code>), 274</p> <p><code>create_disturbs_ase_dev()</code> (in module <code>dp-gen.data.tools.create_random_disturb</code>), 274</p> <p><code>create_disturbs_atomsk()</code> (in module <code>dp-gen.data.tools.create_random_disturb</code>), 274</p> <p><code>create_init_tasks()</code> (in module <code>dp-gen.tools.relabel</code>), 302</p> <p><code>create_path()</code> (in module <code>dp-gen.auto_test.lib.utils</code>), 258</p> <p><code>create_path()</code> (in module <code>dp-gen.data.gen</code>), 277</p> <p><code>create_path()</code> (in module <code>dp-gen.data.surf</code>), 278</p> <p><code>create_path()</code> (in module <code>dp-gen.generator.lib.utils</code>), 292</p> <p><code>create_random_alloys()</code> (in module <code>dp-gen.data.tools.create_random_disturb</code>), 274</p> <p><code>create_tasks()</code> (in module <code>dp-gen.tools.relabel</code>), 302</p> <p><code>current_num_of_itearation</code> (<code>dp-gen.tools.auto_gen_param.Iteration</code> attribute), 300</p> <p><code>current_num_of_sub_itearation</code> (<code>dp-gen.tools.auto_gen_param.Iteration</code> attribute), 300</p> <p><code>current_num_of_sub_systems</code> (<code>dp-gen.tools.auto_gen_param.System</code> attribute), 300</p> <p><code>current_num_of_system</code> (<code>dp-gen.tools.auto_gen_param.System</code> attribute), 300</p>	<p>custom_flags:</p> <p><code>init_bulk_mdata/fp/resources/custom_flags</code> (Argument), 87</p> <p><code>init_reaction_mdata/build/resources/custom_flags</code> (Argument), 133</p> <p><code>init_reaction_mdata/fp/resources/custom_flags</code> (Argument), 146</p> <p><code>init_reaction_mdata/reaxff/resources/custom_flags</code> (Argument), 121</p> <p><code>init_surf_mdata/fp/resources/custom_flags</code> (Argument), 105</p> <p><code>run_mdata/fp/resources/custom_flags</code> (Argument), 70</p> <p><code>run_mdata/model_devi/resources/custom_flags</code> (Argument), 58</p> <p><code>run_mdata/train/resources/custom_flags</code> (Argument), 46</p> <p><code>simplify_mdata/fp/resources/custom_flags</code> (Argument), 199</p> <p><code>simplify_mdata/model_devi/resources/custom_flags</code> (Argument), 187</p> <p><code>simplify_mdata/train/resources/custom_flags</code> (Argument), 175</p> <p>custom_gpu_line:</p> <p><code>init_bulk_mdata/fp/resources[LSF]/kwargs/custom_gpu_line</code> (Argument), 92</p> <p><code>init_bulk_mdata/fp/resources[SlurmJobArray]/kwargs/custom_gpu_line</code> (Argument), 90</p> <p><code>init_bulk_mdata/fp/resources[Slurm]/kwargs/custom_gpu_line</code> (Argument), 91</p> <p><code>init_reaction_mdata/build/resources[LSF]/kwargs/custom_gpu_line</code> (Argument), 139</p> <p><code>init_reaction_mdata/build/resources[SlurmJobArray]/kwargs/custom_gpu_line</code> (Argument), 137</p> <p><code>init_reaction_mdata/build/resources[Slurm]/kwargs/custom_gpu_line</code> (Argument), 137</p> <p><code>init_reaction_mdata/fp/resources[LSF]/kwargs/custom_gpu_line</code> (Argument), 151</p> <p><code>init_reaction_mdata/fp/resources[SlurmJobArray]/kwargs/custom_gpu_line</code> (Argument), 149</p> <p><code>init_reaction_mdata/fp/resources[Slurm]/kwargs/custom_gpu_line</code> (Argument), 149</p> <p><code>init_reaction_mdata/reaxff/resources[LSF]/kwargs/custom_gpu_line</code> (Argument), 126</p> <p><code>init_reaction_mdata/reaxff/resources[SlurmJobArray]/kwargs/custom_gpu_line</code> (Argument), 124</p> <p><code>init_reaction_mdata/reaxff/resources[Slurm]/kwargs/custom_gpu_line</code> (Argument), 125</p> <p><code>init_surf_mdata/fp/resources[LSF]/kwargs/custom_gpu_line</code> (Argument), 111</p> <p><code>init_surf_mdata/fp/resources[SlurmJobArray]/kwargs/custom_gpu_line</code> (Argument), 109</p> <p><code>init_surf_mdata/fp/resources[Slurm]/kwargs/custom_gpu_line</code> (Argument), 109</p>
---	---

run_mdata/fp/resources[LSF]/kwargs/custom_gpu_line (Argument), 200
 (Argument), 75
 run_mdata/fp/resources[SlurmJobArray]/kwargs/custom_gpu_line (Argument), 188
 (Argument), 73
 run_mdata/fp/resources[Slurm]/kwargs/custom_gpu_line (Argument), 176
 (Argument), 74
 run_mdata/model_devi/resources[LSF]/kwargs/custom_gpu_line (Argument), 113
 (Argument), 64
 run_mdata/model_devi/resources[SlurmJobArray]/kwargs/custom_gpu_line (Argument), 62
 run_mdata/model_devi/resources[Slurm]/kwargs/custom_gpu_line (Argument), 62
 run_mdata/train/resources[LSF]/kwargs/custom_gpu_line (Argument), 161
 (Argument), 51
 run_mdata/train/resources[SlurmJobArray]/kwargs/custom_gpu_line (Argument), 49
 run_mdata/train/resources[Slurm]/kwargs/custom_gpu_line (Argument), 50
 simplify_mdata/fp/resources[LSF]/kwargs/custom_gpu_line (Argument), 204
 simplify_mdata/fp/resources[SlurmJobArray]/kwargs/custom_gpu_line (Argument), 202
 simplify_mdata/fp/resources[Slurm]/kwargs/custom_gpu_line (Argument), 203
 simplify_mdata/model_devi/resources[LSF]/kwargs/custom_gpu_line (Argument), 193
 simplify_mdata/model_devi/resources[SlurmJobArray]/kwargs/custom_gpu_line (Argument), 191
 simplify_mdata/model_devi/resources[Slurm]/kwargs/custom_gpu_line (Argument), 191
 simplify_mdata/train/resources[LSF]/kwargs/custom_gpu_line (Argument), 180
 simplify_mdata/train/resources[SlurmJobArray]/kwargs/custom_gpu_line (Argument), 178
 simplify_mdata/train/resources[Slurm]/kwargs/custom_gpu_line (Argument), 179
 customized_script_header_template_file: 169
 init_bulk_mdata/fp/resources/strategy/customized_script_header_template_file (Argument), 88
 init_reaction_mdata/build/resources/strategy/customized_script_header_template_file (Argument), 134
 init_reaction_mdata/fp/resources/strategy/customized_script_header_template_file (Argument), 146
 init_reaction_mdata/reaxff/resources/strategy/customized_script_header_template_file (Argument), 122
 init_surf_mdata/fp/resources/strategy/customized_script_header_template_file (Argument), 106
 run_mdata/fp/resources/strategy/customized_script_header_template_file (Argument), 71
 run_mdata/model_devi/resources/strategy/customized_script_header_template_file (Argument), 59
 run_mdata/train/resources/strategy/customized_script_header_template_file (Argument), 47
 simplify_mdata/fp/resources/strategy/customized_script_header_template_file (Argument), 252

D
 data: 32
 data_args() (in module dp-gen.generator.arginfo), 292
 dataset_size: 114
 deepmd_version: 81
 detailed_report_make_fp: 22
 detect_batch_size() (in module dp-gen.generator.run), 294
 detect_multiplicity() (in module dp-gen.generator.lib.gaussian), 290
 detect_multiplicity() (in module dp-gen.generator.lib.crys), 252

diamond() (in module *dpgen.auto_test.lib.crys*), 252
 dir2car() (in module *dpgen.data.tools.io_lammps*), 275
 disang:
 run_jdata[model_devi_engine=amber]/disang
 (Argument), 31
 disang_prefix:
 run_jdata[model_devi_engine=amber]/disang_prefix
 (Argument), 31
 dp_compress:
 run_jdata/dp_compress (Argument), 20
 simplify_jdata/dp_compress (Argument), 159
 dp_train_skip_neighbor_stat:
 run_jdata/dp_train_skip_neighbor_stat
 (Argument), 20
 simplify_jdata/dp_train_skip_neighbor_stat
 (Argument), 158
 dpgen
 module, 251
 dpgen.arginfo
 module, 302
 dpgen.auto_test
 module, 251
 dpgen.auto_test.ABACUS
 module, 259
 dpgen.auto_test.calculator
 module, 271
 dpgen.auto_test.common_equi
 module, 272
 dpgen.auto_test.common_prop
 module, 272
 dpgen.auto_test.Elastic
 module, 261
 dpgen.auto_test.EOS
 module, 261
 dpgen.auto_test.Gamma
 module, 263
 dpgen.auto_test.gen_confs
 module, 272
 dpgen.auto_test.Interstitial
 module, 264
 dpgen.auto_test.Lammps
 module, 264
 dpgen.auto_test.lib
 module, 251
 dpgen.auto_test.lib.abacus
 module, 251
 dpgen.auto_test.lib.crys
 module, 252
 dpgen.auto_test.lib.lammps
 module, 252
 dpgen.auto_test.lib.lmp
 module, 253
 dpgen.auto_test.lib.mfp_eosfit
 module, 253
 dpgen.auto_test.lib.pwscf
 module, 257
 dpgen.auto_test.lib.siesta
 module, 258
 dpgen.auto_test.lib.util
 module, 258
 dpgen.auto_test.lib.utils
 module, 258
 dpgen.auto_test.lib.vasp
 module, 258
 dpgen.auto_test.mpdb
 module, 272
 dpgen.auto_test.Property
 module, 266
 dpgen.auto_test.refine
 module, 272
 dpgen.auto_test.reproduce
 module, 273
 dpgen.auto_test.run
 module, 273
 dpgen.auto_test.Surface
 module, 267
 dpgen.auto_test.Task
 module, 268
 dpgen.auto_test.Vacancy
 module, 271
 dpgen.auto_test.VASP
 module, 269
 dpgen.collect
 module, 273
 dpgen.collect.collect
 module, 273
 dpgen.data
 module, 273
 dpgen.data.arginfo
 module, 276
 dpgen.data.gen
 module, 277
 dpgen.data.reaction
 module, 278
 dpgen.data.surf
 module, 278
 dpgen.data.tools
 module, 273
 dpgen.data.tools.bcc
 module, 273
 dpgen.data.tools.cessp2force_lin
 module, 274
 dpgen.data.tools.create_random_disturb
 module, 274
 dpgen.data.tools.diamond
 module, 274
 dpgen.data.tools.fcc
 module, 275

dpgen.data.tools.hcp
 module, 275
 dpgen.data.tools.io_lammps
 module, 275
 dpgen.data.tools.sc
 module, 276
 dpgen.database
 module, 279
 dpgen.database.entry
 module, 283
 dpgen.database.run
 module, 284
 dpgen.database.vasp
 module, 284
 dpgen.dispatcher
 module, 287
 dpgen.dispatcher.Dispatcher
 module, 287
 dpgen.generator
 module, 288
 dpgen.generator.arginfo
 module, 292
 dpgen.generator.lib
 module, 288
 dpgen.generator.lib.abacus_scf
 module, 288
 dpgen.generator.lib.cp2k
 module, 289
 dpgen.generator.lib.cvasp
 module, 289
 dpgen.generator.lib.ele_temp
 module, 289
 dpgen.generator.lib.gaussian
 module, 290
 dpgen.generator.lib.lammps
 module, 290
 dpgen.generator.lib.make_calypso
 module, 290
 dpgen.generator.lib.parse_calypso
 module, 290
 dpgen.generator.lib.pwmat
 module, 290
 dpgen.generator.lib.pwscf
 module, 291
 dpgen.generator.lib.run_calypso
 module, 291
 dpgen.generator.lib.siesta
 module, 291
 dpgen.generator.lib.utils
 module, 292
 dpgen.generator.lib.vasp
 module, 292
 dpgen.generator.run
 module, 294

dpgen.gui
 module, 303
 dpgen.main
 module, 303
 dpgen.remote
 module, 297
 dpgen.remote.decide_machine
 module, 297
 dpgen.simplify
 module, 298
 dpgen.simplify.arginfo
 module, 298
 dpgen.simplify.simplify
 module, 299
 dpgen.tools
 module, 300
 dpgen.tools.auto_gen_param
 module, 300
 dpgen.tools.collect_data
 module, 301
 dpgen.tools.relabel
 module, 302
 dpgen.tools.run_report
 module, 302
 dpgen.tools.stat_iter
 module, 302
 dpgen.tools.stat_sys
 module, 302
 dpgen.tools.stat_time
 module, 302
 dpgen.util
 module, 303
 DPPotcar (*class in dpgen.database*), 279
 DPPotcar (*class in dpgen.database.vasp*), 284
 dt:
 init_reaction_jdata/reaxff/dt (*Argument*),
 113
 dump_freq:
 init_reaction_jdata/reaxff/dump_freq (*Argument*), 113
 dump_to_deepmd_raw() (*in module dp-*
 gen.generator.run), 294
E
 ecut:
 run_jdata[fp_style=pwscf]/fp_params/ecut
 (*Argument*), 38
 run_jdata[fp_style=siesta]/fp_params/ecut
 (*Argument*), 35
 simplify_jdata[pwscf]/fp_params/ecut (*Argument*), 167
 simplify_jdata[siesta]/fp_params/ecut
 (*Argument*), 164
 ediff:

run_jdata[fp_style=pwscf]/fp_params/ediff
 (Argument), 38
 run_jdata[fp_style=siesta]/fp_params/ediff
 (Argument), 35
 simplify_jdata[pwscf]/fp_params/ediff
 (Argument), 167
 simplify_jdata[siesta]/fp_params/ediff
 (Argument), 164
 Elastic (class in *dpgen.auto_test.Elastic*), 261
 element_list() (in module *dp-
 gen.auto_test.lib.lammps*), 252
 elements:
 init_bulk_jdata/elements (Argument), 78
 init_surf_jdata/elements (Argument), 96
 email:
 init_bulk_mdata/fp/machine[BohriumContext]/remote_profile/email
 (Argument), 85
 init_reaction_mdata/build/machine[BohriumContext]/remote_profile/email
 (Argument), 131
 init_reaction_mdata/fp/machine[BohriumContext]/remote_profile/email
 (Argument), 143
 init_reaction_mdata/reaxff/machine[BohriumContext]/remote_profile/email
 (Argument), 118
 init_surf_mdata/fp/machine[BohriumContext]/remote_profile/email
 (Argument), 103
 run_mdata/fp/machine[BohriumContext]/remote_profile/email
 (Argument), 68
 run_mdata/model_devi/machine[BohriumContext]/remote_profile/email
 (Argument), 56
 run_mdata/train/machine[BohriumContext]/remote_profile/email
 (Argument), 44
 simplify_mdata/fp/machine[BohriumContext]/remote_profile/email
 (Argument), 197
 simplify_mdata/model_devi/machine[BohriumContext]/remote_profile/email
 (Argument), 185
 simplify_mdata/train/machine[BohriumContext]/remote_profile/email
 (Argument), 172
 ensemble:
 run_jdata[model_devi_engine=lammps]/model_devi_jobs/ensemble
 (Argument), 25
 Entry (class in *dpgen.database*), 280
 Entry (class in *dpgen.database.entry*), 283
 envs:
 init_bulk_mdata/fp/resources/envs (Argument), 89
 init_reaction_mdata/build/resources/envs
 (Argument), 135
 init_reaction_mdata/fp/resources/envs
 (Argument), 147
 init_reaction_mdata/reaxff/resources/envs
 (Argument), 122
 init_surf_mdata/fp/resources/envs (Argument), 107
 run_mdata/fp/resources/envs (Argument), 72
 run_mdata/model_devi/resources/envs
 (Argument), 60
 run_mdata/train/resources/envs (Argument), 48
 simplify_mdata/fp/resources/envs (Argument), 201
 simplify_mdata/model_devi/resources/envs
 (Argument), 189
 simplify_mdata/train/resources/envs
 (Argument), 177
 EOS (class in *dpgen.auto_test.EOS*), 261
 epsilon:
 run_jdata[model_devi_engine=lammps]/epsilon
 (Argument), 29
 epsilon_v:
 run_mdata[model_devi_engine=lammps]/epsilon_v
 (Argument), 29
 expand_matrix_values() (in module *dp-
 gen.generator.run*), 294
 expand_sys_str() (in module *dpgen.util*), 303
 fcc() (in module *dpgen.auto_test.lib.crys*), 252
 fcc() (in module *dpgen.auto_test.lib.crys*), 252
 ff:
 init_reaction_jdata/reaxff/ff (Argument), 113
 file_len() (in module *dpgen.tools.collect_data*), 301
 final_stru() (in module *dpgen.auto_test.lib.abacus*), 251
 find_only_one_key() (in module *dp-
 gen.generator.run*), 294
 forward_common_files (dpgen.auto_test.Task.Task
 property), 269
 forward_common_files() (dp-
 gen.auto_test.ABACUS.ABACUS method), 260
 forward_common_files() (dp-
 gen.auto_test.Lammps.Lammps method), 265
 forward_common_files() (dp-
 gen.auto_test.VASP.VASP method), 270

`forward_files` (*dpngen.auto_test.Task.Task* property), 269
`forward_files()` (*dpngen.auto_test.ABACUS.ABACUS* method), 260
`forward_files()` (*dpngen.auto_test.Lammps.Lammps* method), 265
`forward_files()` (*dpngen.auto_test.VASP.VASP* method), 270
`fp`:
 `init_bulk_mdata/fp` (*Argument*), 81
 `init_reaction_mdata/fp` (*Argument*), 139
 `init_surf_mdata/fp` (*Argument*), 99
 `run_mdata/fp` (*Argument*), 64
 `simplify_mdata/fp` (*Argument*), 193
`fp_accurate_soft_threshold`:
 `run_jdata/fp_accurate_soft_threshold` (*Argument*), 22
 `simplify_jdata/fp_accurate_soft_threshold` (*Argument*), 161
`fp_accurate_threshold`:
 `run_jdata/fp_accurate_threshold` (*Argument*), 22
 `simplify_jdata/fp_accurate_threshold` (*Argument*), 160
`fp_aniso_kspacing`:
 `run_jdata[fp_style=vasp]/fp_aniso_kspacing` (*Argument*), 32
 `simplify_jdata[vasp]/fp_aniso_kspacing` (*Argument*), 161
`fp_args()` (in module *dpngen.generator.arginfo*), 292
`fp_args()` (in module *dpngen.simplify.arginfo*), 298
`fp_cluster_vacuum`:
 `run_jdata/fp_cluster_vacuum` (*Argument*), 22
`fp_dpks_descriptor`:
 `run_jdata[fp_style=abacus]/fp_dpks_descriptor` (*Argument*), 37
 `simplify_jdata[abacus]/fp_dpks_descriptor` (*Argument*), 166
`fp_incar`:
 `run_jdata[fp_style=abacus]/fp_incar` (*Argument*), 36
 `run_jdata[fp_style=vasp]/fp_incar` (*Argument*), 32
 `simplify_jdata[abacus]/fp_incar` (*Argument*), 166
 `simplify_jdata[vasp]/fp_incar` (*Argument*), 161
`fp_kpt_file`:
 `run_jdata[fp_style=abacus]/fp_kpt_file` (*Argument*), 37
 `simplify_jdata[abacus]/fp_kpt_file` (*Argument*), 166
`fp_orb_files`:
 `run_jdata[fp_style=abacus]/fp_orb_files` (*Argument*), 36
 `simplify_jdata[abacus]/fp_orb_files` (*Argument*), 165
`fp_params`:
 `run_jdata[fp_style=amber/diff]/fp_params` (*Argument*), 37
 `run_jdata[fp_style=custom]/fp_params` (*Argument*), 39
 `run_jdata[fp_style=gaussian]/fp_params` (*Argument*), 33
 `run_jdata[fp_style=pwscf]/fp_params` (*Argument*), 38
 `run_jdata[fp_style=siesta]/fp_params` (*Argument*), 34
 `simplify_jdata[custom]/fp_params` (*Argument*), 168
 `simplify_jdata[gaussian]/fp_params` (*Argument*), 162
 `simplify_jdata[pwscf]/fp_params` (*Argument*), 167
 `simplify_jdata[siesta]/fp_params` (*Argument*), 164
`fp_pp_files`:
 `run_jdata[fp_style=abacus]/fp_pp_files` (*Argument*), 36
 `run_jdata[fp_style=pwscf]/fp_pp_files` (*Argument*), 38
 `run_jdata[fp_style=siesta]/fp_pp_files` (*Argument*), 35
 `run_jdata[fp_style=vasp]/fp_pp_files` (*Argument*), 32
 `simplify_jdata[abacus]/fp_pp_files` (*Argument*), 165
 `simplify_jdata[pwscf]/fp_pp_files` (*Argument*), 166
 `simplify_jdata[siesta]/fp_pp_files` (*Argument*), 165
 `simplify_jdata[vasp]/fp_pp_files` (*Argument*), 161
`fp_pp_path`:
 `run_jdata[fp_style=abacus]/fp_pp_path` (*Argument*), 36
 `run_jdata[fp_style=pwscf]/fp_pp_path` (*Argument*), 38
 `run_jdata[fp_style=siesta]/fp_pp_path` (*Argument*), 35
 `run_jdata[fp_style=vasp]/fp_pp_path` (*Argument*), 32
 `simplify_jdata[abacus]/fp_pp_path` (*Argument*), 165
 `simplify_jdata[pwscf]/fp_pp_path` (*Argument*), 166
 `simplify_jdata[siesta]/fp_pp_path` (*Argument*), 164

simplify_jdata[vasp]/fp_pp_path (Argument), 161
 fp_skip_bad_box:
 run_jdata[fp_style=vasp]/fp_skip_bad_box (Argument), 32
 simplify_jdata[vasp]/fp_skip_bad_box (Argument), 162
 fp_style:
 run_jdata/fp_style (Argument), 32
 simplify_jdata/fp_style (Argument), 161
 fp_style_abacus_args() (in module *dp-gen.generator.arginfo*), 292
 fp_style_amber_diff_args() (in module *dp-gen.generator.arginfo*), 293
 fp_style_cp2k_args() (in module *dp-gen.generator.arginfo*), 293
 fp_style_custom_args() (in module *dp-gen.generator.arginfo*), 293
 fp_style_gaussian_args() (in module *dp-gen.generator.arginfo*), 293
 fp_style_pwscf_args() (in module *dp-gen.generator.arginfo*), 293
 fp_style_siesta_args() (in module *dp-gen.generator.arginfo*), 293
 fp_style_variant_type_args() (in module *dp-gen.generator.arginfo*), 293
 fp_style_variant_type_args() (in module *dp-gen.simplify.arginfo*), 298
 fp_style_vasp_args() (in module *dp-gen.generator.arginfo*), 293
 fp_task_max:
 run_jdata/fp_task_max (Argument), 22
 simplify_jdata/fp_task_max (Argument), 160
 fp_task_min:
 run_jdata/fp_task_min (Argument), 22
 simplify_jdata/fp_task_min (Argument), 160
 fragment_guesses:
 run_jdata[fp_style=gaussian]/fp_params/fragment_guesses (Argument), 34
 simplify_jdata[gaussian]/fp_params/fragment_guesses (Argument), 163
 from_dict() (*dpgen.database.DPPotcar* class method), 280
 from_dict() (*dpgen.database.Entry* class method), 280
 from_dict() (*dpgen.database.entry.Entry* class method), 284
 from_dict() (*dpgen.database.vasp.DPPotcar* class method), 285
 from_dict() (*dpgen.database.vasp.VaspInput* class method), 286
 from_dict() (*dpgen.database.VaspInput* class method), 282
 from_directory() (*dpgen.database.vasp.VaspInput* static method), 286
 from_directory() (*dpgen.database.VaspInput* static method), 282
 from_file() (*dpgen.database.DPPotcar* class method), 280
 from_file() (*dpgen.database.vasp.DPPotcar* class method), 285
 from_poscar:
 init_bulk_jdata/from_poscar (Argument), 79
 init_surf_jdata/from_poscar (Argument), 97
 from_poscar_path:
 init_bulk_jdata/from_poscar_path (Argument), 79
 init_surf_jdata/from_poscar_path (Argument), 97
 from_system_data() (in module *dp-gen.auto_test.lib.lmp*), 253

G

Gamma (class in *dpgen.auto_test.Gamma*), 263
 gen_alloy() (in module *dpgen.auto_test.gen_confs*), 272
 gen_box() (in module *dpgen.data.tools.bcc*), 273
 gen_box() (in module *dpgen.data.tools.diamond*), 274
 gen_box() (in module *dpgen.data.tools.fcc*), 275
 gen_box() (in module *dpgen.data.tools.hcp*), 275
 gen_box() (in module *dpgen.data.tools.sc*), 276
 gen_collect() (in module *dpgen.collect.collect*), 273
 gen_ele_std() (in module *dpgen.auto_test.gen_confs*), 272
 gen_element() (in module *dpgen.auto_test.gen_confs*), 272
 gen_element_std() (in module *dp-gen.auto_test.gen_confs*), 272
 gen_init_bulk() (in module *dpgen.data.gen*), 277
 gen_init_reaction() (in module *dp-gen.data.reaction*), 278
 gen_init_surf() (in module *dpgen.data.surf*), 278
 gen_init_surf() (in module *dp-gen.generator.lib.run_calypso*), 291
 gen_random_disturb() (in module *dp-gen.data.tools.create_random_disturb*), 274
 gen_random_emat() (in module *dp-gen.data.tools.create_random_disturb*), 274
 gen_run() (in module *dpgen.generator.run*), 294
 gen_simplify() (in module *dpgen.simplify.simplify*), 299
 gen_structures() (in module *dp-gen.generator.lib.run_calypso*), 291
 gen_sub_iter() (*dpgen.tools.auto_gen_param.Iteration* method), 300
 gen_test() (in module *dpgen.auto_test.run*), 273
 general_mdata_arginfo() (in module *dpgen.arginfo*), 302

`general_simplify_arginfo()` (in module `dp-gen.simplify.arginfo`), 298
`get_abacus_input_parameters()` (in module `dp-gen.generator.lib.abacus_scf`), 288
`get_abacus_STRU()` (in module `dp-gen.generator.lib.abacus_scf`), 288
`get_additional_from_STRU()` (in module `dp-gen.generator.lib.abacus_scf`), 288
`get_all_dumped_forces()` (in module `dp-gen.generator.lib.lammps`), 290
`get_atom_types()` (in module `dp-gen.generator.lib.pwscf`), 291
`get_atomic_masses()` (in module `dp-gen.generator.run`), 294
`get_atoms()` (in module `dp-gen.auto_test.lib.lmp`), 253
`get_atoms_ntypes()` (in module `dp-gen.data.tools.io_lammps`), 275
`get_atype()` (in module `dp-gen.auto_test.lib.lmp`), 253
`get_base_area()` (in module `dp-gen.auto_test.lib.lammps`), 252
`get_basic_param_json()` (in module `dp-gen.tools.auto_gen_param`), 301
`get_block()` (in module `dp-gen.generator.lib.pwscf`), 291
`get_boxes()` (in module `dp-gen.auto_test.lib.vasp`), 258
`get_cell()` (in module `dp-gen.generator.lib.pwscf`), 291
`get_coords()` (in module `dp-gen.generator.lib.pwscf`), 291
`get_dumped_forces()` (in module `dp-gen.generator.lib.lammps`), 290
`get_energies()` (in module `dp-gen.auto_test.lib.vasp`), 258
`get_energy()` (in module `dp-gen.generator.lib.pwscf`), 291
`get_eos_list()` (in module `dp-gen.auto_test.lib.mfp_eosfit`), 254
`get_eos_list_3p()` (in module `dp-gen.auto_test.lib.mfp_eosfit`), 254
`get_eos_list_4p()` (in module `dp-gen.auto_test.lib.mfp_eosfit`), 255
`get_eos_list_5p()` (in module `dp-gen.auto_test.lib.mfp_eosfit`), 255
`get_eos_list_6p()` (in module `dp-gen.auto_test.lib.mfp_eosfit`), 255
`get_force()` (in module `dp-gen.generator.lib.pwscf`), 291
`get_init_data_sys()` (in module `dp-gen.tools.auto_gen_param`), 301
`get_job_names()` (in module `dp-gen.generator.run`), 294
`get_lmp_info()` (in module `dp-gen.tools.relabel`), 302
`get_lmpbox()` (in module `dp-gen.auto_test.lib.lmp`), 253
`get_machine_info()` (in module `dp-gen.auto_test.lib.util`), 258
`get_mass_from_STRU()` (in module `dp-gen.generator.lib.abacus_scf`), 288
`get_model_devi_jobs()` (in module `dp-gen.tools.auto_gen_param`), 301
`get_multi_system()` (in module `dp-gen.simplify.simplify`), 299
`get_natoms()` (in module `dp-gen.auto_test.lib.lmp`), 253
`get_natoms()` (in module `dp-gen.generator.lib.pwscf`), 291
`get_natoms_from_stru()` (in module `dp-gen.generator.lib.abacus_scf`), 288
`get_natoms_vec()` (in module `dp-gen.auto_test.lib.lmp`), 253
`get_natomtypes()` (in module `dp-gen.auto_test.lib.lmp`), 253
`get_nev()` (in module `dp-gen.auto_test.lib.lammps`), 252
`get_nev()` (in module `dp-gen.auto_test.lib.vasp`), 258
`get_nframes()` (in module `dp-gen.generator.run`), 294
`get_outcar_files()` (in module `dp-gen.data.tools.cessp2force_lin`), 274
`get_poscar_natoms()` (in module `dp-gen.auto_test.lib.vasp`), 258
`get_poscar_types()` (in module `dp-gen.auto_test.lib.vasp`), 258
`get_posi()` (in module `dp-gen.auto_test.lib.lmp`), 253
`get_stress()` (in module `dp-gen.auto_test.lib.lammps`), 252
`get_stress()` (in module `dp-gen.auto_test.lib.vasp`), 258
`get_stress()` (in module `dp-gen.generator.lib.pwscf`), 291
`get_structure()` (in module `dp-gen.auto_test.mpdb`), 272
`get_sub_system()` (`dp-gen.tools.auto_gen_param.System` method), 300
`get_sys_configs()` (in module `dp-gen.tools.auto_gen_param`), 301
`get_sys_index()` (in module `dp-gen.generator.run`), 294
`get_system_cls()` (in module `dp-gen.simplify.simplify`), 299
`get_system_list()` (in module `dp-gen.tools.auto_gen_param`), 301
`get_typeid()` (in module `dp-gen.data.tools.io_lammps`), 275
`get_types()` (in module `dp-gen.generator.lib.pwscf`), 291
`gpu_exclusive:`
 `init_bulk_mdata/fp/resources[LSF]/kwargs/gpu_exclusive` (Argument), 92
 `init_reaction_mdata/build/resources[LSF]/kwargs/gpu_exclusive` (Argument), 139
 `init_reaction_mdata/fp/resources[LSF]/kwargs/gpu_exclusive` (Argument), 151
 `init_reaction_mdata/reaxff/resources[LSF]/kwargs/gpu_exclusive` (Argument), 126

init_surf_mdata/fp/resources[LSF]/kwargs/gpu_exclusive_mdata/fp/resources/gpu_per_node
 (Argument), 111

run_mdata/fp/resources[LSF]/kwargs/gpu_exclusive_mdata/model_devi/resources/gpu_per_node
 (Argument), 75

run_mdata/model_devi/resources[LSF]/kwargs/gpu_exclusive_mdata/train/resources/gpu_per_node
 (Argument), 64

run_mdata/train/resources[LSF]/kwargs/gpu_exclusive_mdata/train/resources/gpu_per_node
 (Argument), 51

simplify_mdata/fp/resources[LSF]/kwargs/gpu_exclusive_mdata/build/resources[LSF]/kwargs/gpu_usage
 (Argument), 204

simplify_mdata/model_devi/resources[LSF]/kwargs/gpu_exclusive_mdata/build/resources[LSF]/kwargs/gpu_usage
 (Argument), 193

simplify_mdata/train/resources[LSF]/kwargs/gpu_exclusive_mdata/build/resources[LSF]/kwargs/gpu_usage
 (Argument), 180

gpu_new_syntax:
 init_bulk_mdata/fp/resources[LSF]/kwargs/gpu_new_syntax_mdata/fp/resources[LSF]/kwargs/gpu_usage
 (Argument), 92

init_reaction_mdata/build/resources[LSF]/kwargs/gpu_new_syntax_mdata/build/resources[LSF]/kwargs/gpu_usage
 (Argument), 138

init_reaction_mdata/fp/resources[LSF]/kwargs/gpu_new_syntax_mdata/build/resources[LSF]/kwargs/gpu_usage
 (Argument), 151

init_reaction_mdata/reaxff/resources[LSF]/kwargs/gpu_new_syntax_mdata/build/resources[LSF]/kwargs/gpu_usage
 (Argument), 126

init_surf_mdata/fp/resources[LSF]/kwargs/gpu_new_syntax_mdata/build/resources[LSF]/kwargs/gpu_usage
 (Argument), 110

run_mdata/fp/resources[LSF]/kwargs/gpu_new_syntax_mdata/build/resources[LSF]/kwargs/gpu_usage
 (Argument), 75

run_mdata/model_devi/resources[LSF]/kwargs/gpu_new_syntax_mdata/build/resources[LSF]/kwargs/gpu_usage
 (Argument), 63

run_mdata/train/resources[LSF]/kwargs/gpu_new_syntax_mdata/build/resources[LSF]/kwargs/gpu_usage
 (Argument), 51

simplify_mdata/fp/resources[LSF]/kwargs/gpu_new_syntax_mdata/build/resources[LSF]/kwargs/gpu_usage
 (Argument), 204

simplify_mdata/model_devi/resources[LSF]/kwargs/gpu_new_syntax_mdata/build/resources[LSF]/kwargs/gpu_usage
 (Argument), 192

simplify_mdata/train/resources[LSF]/kwargs/gpu_new_syntax_mdata/build/resources[LSF]/kwargs/gpu_usage
 (Argument), 180

gpu_group_size:
 init_bulk_mdata/fp/resources[LSF]/kwargs/gpu_group_size_mdata/build/resources[LSF]/kwargs/gpu_group_size
 (Argument), 87

init_reaction_mdata/build/resources[LSF]/kwargs/gpu_group_size_mdata/build/resources[LSF]/kwargs/gpu_group_size
 (Argument), 204

init_reaction_mdata/model_devi/resources[LSF]/kwargs/gpu_group_size_mdata/build/resources[LSF]/kwargs/gpu_group_size
 (Argument), 192

init_reaction_mdata/train/resources[LSF]/kwargs/gpu_group_size_mdata/build/resources[LSF]/kwargs/gpu_group_size
 (Argument), 180

init_reaction_mdata/reaxff/resources[LSF]/kwargs/gpu_group_size_mdata/build/resources[LSF]/kwargs/gpu_group_size
 (Argument), 121

init_surf_mdata/fp/resources[LSF]/kwargs/gpu_group_size_mdata/build/resources[LSF]/kwargs/gpu_group_size
 (Argument), 105

run_mdata/fp/resources[LSF]/kwargs/gpu_group_size_mdata/build/resources[LSF]/kwargs/gpu_group_size
 (Argument), 70

run_mdata/model_devi/resources[LSF]/kwargs/gpu_group_size_mdata/build/resources[LSF]/kwargs/gpu_group_size
 (Argument), 58

run_mdata/train/resources[LSF]/kwargs/gpu_group_size_mdata/build/resources[LSF]/kwargs/gpu_group_size
 (Argument), 46

simplify_mdata/fp/resources[LSF]/kwargs/gpu_group_size_mdata/build/resources[LSF]/kwargs/gpu_group_size
 (Argument), 199

simplify_mdata/model_devi/resources[LSF]/kwargs/gpu_group_size_mdata/build/resources[LSF]/kwargs/gpu_group_size
 (Argument), 187

simplify_mdata/train/resources[LSF]/kwargs/gpu_group_size_mdata/build/resources[LSF]/kwargs/gpu_group_size
 (Argument), 175

H

hcp() (in module *dpgen.auto_test.lib.crys*), 252
 head_ratio:
 init_surf_jdata/head_ratio (Argument), 98
 high_level:
 run_jdata[fp_style=amber/diff]/high_level
 (Argument), 37
 high_level_mdin:
 run_jdata[fp_style=amber/diff]/fp_params/high_level_mdin
 (Argument), 37
 hostname:
 init_bulk_mdata/fp/machine[SSHContext]/remote_profile/hostname
 (Argument), 83
 init_reaction_mdata/build/machine[SSHContext]/remote_profile/hostname
 (Argument), 128
 init_reaction_mdata/fp/machine[SSHContext]/remote_profile/hostname
 (Argument), 141
 init_reaction_mdata/reaxff/machine[SSHContext]/remote_profile/hostname
 (Argument), 116
 init_surf_mdata/fp/machine[SSHContext]/remote_profile/hostname
 (Argument), 101
 run_mdata/fp/machine[SSHContext]/remote_profile/hostname
 (Argument), 66
 run_mdata/model_devi/machine[SSHContext]/remote_profile/hostname
 (Argument), 53
 run_mdata/train/machine[SSHContext]/remote_profile/hostname
 (Argument), 42
 simplify_mdata/fp/machine[SSHContext]/remote_profile/hostname
 (Argument), 195
 simplify_mdata/model_devi/machine[SSHContext]/remote_profile/hostname
 (Argument), 182
 simplify_mdata/train/machine[SSHContext]/remote_profile/hostname
 (Argument), 170
 simplify_mdata/model_devi/resources/strategy/if_cuda_multi_devices
 (Argument), 188
 simplify_mdata/train/resources/strategy/if_cuda_multi_devices
 (Argument), 175
 ignore_exit_code:
 init_bulk_mdata/fp/machine[BohriumContext]/remote_profile/hostname
 (Argument), 86
 init_reaction_mdata/build/machine[BohriumContext]/remote_profile/hostname
 (Argument), 131
 init_reaction_mdata/fp/machine[BohriumContext]/remote_profile/hostname
 (Argument), 144
 init_reaction_mdata/reaxff/machine[BohriumContext]/remote_profile/hostname
 (Argument), 119
 init_surf_mdata/fp/machine[BohriumContext]/remote_profile/hostname
 (Argument), 104
 run_mdata/fp/machine[BohriumContext]/remote_profile/hostname
 (Argument), 69
 run_mdata/model_devi/machine[BohriumContext]/remote_profile/hostname
 (Argument), 57
 run_mdata/train/machine[BohriumContext]/remote_profile/hostname
 (Argument), 44
 simplify_mdata/fp/machine[BohriumContext]/remote_profile/hostname
 (Argument), 198
 simplify_mdata/model_devi/machine[BohriumContext]/remote_profile/hostname
 (Argument), 185
 simplify_mdata/train/machine[BohriumContext]/remote_profile/hostname
 (Argument), 173
 incarc_upper() (in module *dpgen.generator.lib.vasp*), 292
 index_iteration: (dp-gen.tools.auto_gen_param.Iteration property), 309
 index_system (dp-gen.tools.auto_gen_param.System property), 301
 info() (in module *dpgen*), 251
 init_batch_size:
 run_jdata/init_batch_size (Argument), 19
 simplify_jdata/init_batch_size (Argument), 156
 if_cuda_multi_devices:
 init_bulk_mdata/fp/resources/strategy/if_cuda_multi_devices
 (Argument), 87
 init_reaction_mdata/build/resources/strategy/if_cuda_multi_devices
 (Argument), 134
 init_reaction_mdata/fp/resources/strategy/if_cuda_multi_devices
 (Argument), 146
 init_reaction_mdata/reaxff/resources/strategy/if_cuda_multi_devices
 (Argument), 121
 init_surf_mdata/fp/resources/strategy/if_cuda_multi_devices
 (Argument), 106
 run_mdata/fp/resources/strategy/if_cuda_multi_devices
 (Argument), 70
 run_mdata/model_devi/resources/strategy/if_cuda_multi_devices
 (Argument), 59
 run_mdata/train/resources/strategy/if_cuda_multi_devices
 (Argument), 46
 simplify_mdata/fp/resources/strategy/if_cuda_multi_devices
 (Argument), 199
 simplify_mdata/model_devi/resources/strategy/if_cuda_multi_devices
 (Argument), 188
 simplify_mdata/train/resources/strategy/if_cuda_multi_devices
 (Argument), 175
 ignore_exit_code:
 init_bulk_mdata/fp/machine[BohriumContext]/remote_profile/hostname
 (Argument), 86
 init_reaction_mdata/build/machine[BohriumContext]/remote_profile/hostname
 (Argument), 131
 init_reaction_mdata/fp/machine[BohriumContext]/remote_profile/hostname
 (Argument), 144
 init_reaction_mdata/reaxff/machine[BohriumContext]/remote_profile/hostname
 (Argument), 119
 init_surf_mdata/fp/machine[BohriumContext]/remote_profile/hostname
 (Argument), 104
 run_mdata/fp/machine[BohriumContext]/remote_profile/hostname
 (Argument), 69
 run_mdata/model_devi/machine[BohriumContext]/remote_profile/hostname
 (Argument), 57
 run_mdata/train/machine[BohriumContext]/remote_profile/hostname
 (Argument), 44
 simplify_mdata/fp/machine[BohriumContext]/remote_profile/hostname
 (Argument), 198
 simplify_mdata/model_devi/machine[BohriumContext]/remote_profile/hostname
 (Argument), 185
 simplify_mdata/train/machine[BohriumContext]/remote_profile/hostname
 (Argument), 173
 incarc_upper() (in module *dpgen.generator.lib.vasp*), 292
 index_iteration: (dp-gen.tools.auto_gen_param.Iteration property), 309
 index_system (dp-gen.tools.auto_gen_param.System property), 301
 info() (in module *dpgen*), 251
 init_batch_size:
 run_jdata/init_batch_size (Argument), 19
 simplify_jdata/init_batch_size (Argument), 156
 init_bulk_abacus_args() (in module *dp-gen.data.arginfo*), 276
 init_bulk_jdata (Argument)
 init_bulk_jdata: 78
 init_bulk_jdata/cell_type (Argument)
 cell_type: 78
 init_bulk_jdata/coll_ndata (Argument)
 coll_ndata: 80
 init_bulk_jdata/elements (Argument)
 elements: 78
 init_bulk_jdata/from_poscar (Argument)
 from_poscar: 79
 init_bulk_jdata/from_poscar_path (Argument)
 from_poscar_path: 79
 init_bulk_jdata/init_fp_style (Argument)

```

    init_fp_style:, 80
init_bulk_jdata/md_incar (Argument)
    md_incar:, 79
init_bulk_jdata/md_nstep (Argument)
    md_nstep:, 80
init_bulk_jdata/pert_atom (Argument)
    pert_atom:, 80
init_bulk_jdata/pert_box (Argument)
    pert_box:, 79
init_bulk_jdata/pert_numb (Argument)
    pert_numb:, 79
init_bulk_jdata/potcars (Argument)
    potcars:, 78
init_bulk_jdata/relax_incar (Argument)
    relax_incar:, 79
init_bulk_jdata/scale (Argument)
    scale:, 79
init_bulk_jdata/skip_relax (Argument)
    skip_relax:, 79
init_bulk_jdata/stages (Argument)
    stages:, 78
init_bulk_jdata/super_cell (Argument)
    super_cell:, 78
init_bulk_jdata/type_map (Argument)
    type_map:, 80
init_bulk_jdata:
    init_bulk_jdata (Argument), 78
init_bulk_jdata_arginfo() (in module dp-
    gen.data_arginfo), 276
init_bulk_jdata[ABACUS]/atom_masses (Argu-
    ment)
    atom_masses:, 81
init_bulk_jdata[ABACUS]/md_kpt (Argument)
    md_kpt:, 80
init_bulk_jdata[ABACUS]/relax_kpt (Argument)
    relax_kpt:, 80
init_bulk_mdata (Argument)
    init_bulk_mdata:, 81
init_bulk_mdata/api_version (Argument)
    api_version:, 81
init_bulk_mdata/deepmd_version (Argument)
    deepmd_version:, 81
init_bulk_mdata/fp (Argument)
    fp:, 81
init_bulk_mdata/fp/command (Argument)
    command:, 81
init_bulk_mdata/fp/machine (Argument)
    machine:, 81
init_bulk_mdata/fp/machine/batch_type (Argu-
    ment)
    batch_type:, 81
init_bulk_mdata/fp/machine/clean_asynchronously
    (Argument)
    clean_asynchronously:, 82

```

```

init_bulk_mdata/fp/machine/context_type
    (Argument)
    context_type:, 82
init_bulk_mdata/fp/machine/local_root (Argu-
    ment)
    local_root:, 82
init_bulk_mdata/fp/machine/remote_root (Argu-
    ment)
    remote_root:, 82
init_bulk_mdata/fp/machine[BohriumContext]/remote_profile
    (Argument)
    remote_profile:, 85
init_bulk_mdata/fp/machine[BohriumContext]/remote_profile/
    (Argument)
    email:, 85
init_bulk_mdata/fp/machine[BohriumContext]/remote_profile/
    (Argument)
    ignore_exit_code:, 86
init_bulk_mdata/fp/machine[BohriumContext]/remote_profile/
    (Argument)
    input_data:, 86
init_bulk_mdata/fp/machine[BohriumContext]/remote_profile/
    (Argument)
    keep_backup:, 86
init_bulk_mdata/fp/machine[BohriumContext]/remote_profile/
    (Argument)
    password:, 85
init_bulk_mdata/fp/machine[BohriumContext]/remote_profile/
    (Argument)
    program_id:, 85
init_bulk_mdata/fp/machine[BohriumContext]/remote_profile/
    (Argument)
    retry_count:, 85
init_bulk_mdata/fp/machine[HDF5Context]/remote_profile
    (Argument)
    remote_profile:, 85
init_bulk_mdata/fp/machine[LazyLocalContext]/remote_profile
    (Argument)
    remote_profile:, 84
init_bulk_mdata/fp/machine[LocalContext]/remote_profile
    (Argument)
    remote_profile:, 86
init_bulk_mdata/fp/machine[OpenAPIContext]/remote_profile
    (Argument)
    remote_profile:, 86
init_bulk_mdata/fp/machine[SSHContext]/remote_profile
    (Argument)
    remote_profile:, 82
init_bulk_mdata/fp/machine[SSHContext]/remote_profile/host
    (Argument)
    hostname:, 83
init_bulk_mdata/fp/machine[SSHContext]/remote_profile/key_
    (Argument)
    key_filename:, 83

```

```

init_bulk_mdata/fp/machine[SSHContext]/remote_profile/look_for_keys: 89
    (Argument)
    look_for_keys: 84
init_bulk_mdata/fp/machine[SSHContext]/remote_profile/passphrase: 87
    (Argument)
    passphrase: 83
init_bulk_mdata/fp/machine[SSHContext]/remote_profile/password: 88
    (Argument)
    password: 83
init_bulk_mdata/fp/machine[SSHContext]/remote_profile/prepare_script: 89
    (Argument)
    port: 83
init_bulk_mdata/fp/machine[SSHContext]/remote_profile/tar_compress: 87
    (Argument)
    tar_compress: 84
init_bulk_mdata/fp/machine[SSHContext]/remote_profile/timeout: 88
    (Argument)
    timeout: 84
init_bulk_mdata/fp/machine[SSHContext]/remote_profile/totp_secret: 87
    (Argument)
    totp_secret: 84
init_bulk_mdata/fp/machine[SSHContext]/remote_profile/username: 88
    (Argument)
    username: 83
init_bulk_mdata/fp/resources (Argument)
    resources: 86
init_bulk_mdata/fp/resources/append_script
    (Argument)
    append_script: 89
init_bulk_mdata/fp/resources/batch_type
    (Argument)
    batch_type: 89
init_bulk_mdata/fp/resources/cpu_per_node
    (Argument)
    cpu_per_node: 87
init_bulk_mdata/fp/resources/custom_flags
    (Argument)
    custom_flags: 87
init_bulk_mdata/fp/resources/envs (Argument)
    envs: 89
init_bulk_mdata/fp/resources/gpu_per_node
    (Argument)
    gpu_per_node: 87
init_bulk_mdata/fp/resources/group_size
    (Argument)
    group_size: 87
init_bulk_mdata/fp/resources/module_list (Argument)
    module_list: 89
init_bulk_mdata/fp/resources/module_purge
    (Argument)
    module_purge: 88
init_bulk_mdata/fp/resources/module_unload_list
    (Argument)
    module_unload_list: 89
init_bulk_mdata/fp/resources/number_node
    (Argument)
init_bulk_mdata/fp/resources/para_deg
    (Argument)
init_bulk_mdata/fp/resources/prepare_script
    (Argument)
init_bulk_mdata/fp/resources/queue_name
    (Argument)
init_bulk_mdata/fp/resources/source_list
    (Argument)
init_bulk_mdata/fp/resources/strategy
    (Argument)
init_bulk_mdata/fp/resources/strategy/customized_script_header
    (Argument)
init_bulk_mdata/fp/resources/strategy/customized_script_header_template_file:
    88
init_bulk_mdata/fp/resources/strategy/if_cuda_multi_device
    (Argument)
    if_cuda_multi_devices: 87
init_bulk_mdata/fp/resources/strategy/ratio_unfinished
    (Argument)
    ratio_unfinished: 88
init_bulk_mdata/fp/resources/wait_time (Argument)
    wait_time: 89
init_bulk_mdata/fp/resources[Bohrium]/kwargs
    (Argument)
    kwargs: 91
init_bulk_mdata/fp/resources[DistributedShell]/kwargs
    (Argument)
    kwargs: 90
init_bulk_mdata/fp/resources[Fugaku]/kwargs
    (Argument)
    kwargs: 92
init_bulk_mdata/fp/resources[LSF]/kwargs (Argument)
    kwargs: 92
init_bulk_mdata/fp/resources[LSF]/kwargs/custom_gpu_line
    (Argument)
    custom_gpu_line: 92
init_bulk_mdata/fp/resources[LSF]/kwargs/gpu_exclusive
    (Argument)
    gpu_exclusive: 92
init_bulk_mdata/fp/resources[LSF]/kwargs/gpu_new_syntax
    (Argument)
    gpu_new_syntax: 92
init_bulk_mdata/fp/resources[LSF]/kwargs/gpu_usage

```

```

        (Argument)
    gpu_usage:, 92
init_bulk_mdata/fp/resources[OpenAPI]/kwargs
    (Argument)
    kwargs:, 91
init_bulk_mdata/fp/resources[PBS]/kwargs (Argument)
    kwargs:, 90
init_bulk_mdata/fp/resources[SGE]/kwargs (Argument)
    kwargs:, 91
init_bulk_mdata/fp/resources[Shell]/kwargs
    (Argument)
    kwargs:, 90
init_bulk_mdata/fp/resources[SlurmJobArray]/kwargs
    (Argument)
    kwargs:, 90
init_bulk_mdata/fp/resources[SlurmJobArray]/kwargs/custom_gpu_line
    (Argument)
    custom_gpu_line:, 90
init_bulk_mdata/fp/resources[SlurmJobArray]/kwargs/slurm_job_size
    (Argument)
    slurm_job_size:, 91
init_bulk_mdata/fp/resources[Slurm]/kwargs
    (Argument)
    kwargs:, 91
init_bulk_mdata/fp/resources[Slurm]/kwargs/custom_gpu_line
    (Argument)
    custom_gpu_line:, 91
init_bulk_mdata/fp/resources[Torque]/kwargs
    (Argument)
    kwargs:, 92
init_bulk_mdata/fp/user_backward_files (Argument)
    user_backward_files:, 93
init_bulk_mdata/fp/user_forward_files (Argument)
    user_forward_files:, 93
init_bulk_mdata:
    init_bulk_mdata (Argument), 81
init_bulk_mdata_arginfo() (in module dp-gen.data.arginfo), 276
init_bulk_variant_type_args() (in module dp-gen.data.arginfo), 276
init_bulk_vasp_args() (in module dp-gen.data.arginfo), 276
init_data_prefix:
    run_jdata/init_data_prefix (Argument), 19
    simplify_jdata/init_data_prefix (Argument), 156
init_data_sys:
    run_jdata/init_data_sys (Argument), 19
    simplify_jdata/init_data_sys (Argument), 156
init_fp_style:
    init_bulk_jdata/init_fp_style (Argument), 80
init_guess() (in module dp-gen.auto_test.lib.mfp_eosfit), 255
init_model() (in module dp-gen.simplify.simplify), 299
init_pick() (in module dp-gen.simplify.simplify), 299
init_pick_number:
    simplify_jdata/init_pick_number (Argument), 157
init_reaction_jdata (Argument)
    init_reaction_jdata:, 112
init_reaction_jdata/cutoff (Argument)
    cutoff:, 113
init_reaction_jdata/dataset_size (Argument)
    dataset_size:, 114
init_reaction_jdata/qmkeywords (Argument)
    qmkeywords:, 114
init_reaction_jdata/reaxff (Argument)
    reaxff:, 112
init_reaction_jdata/reaxff/control (Argument)
    control:, 113
init_reaction_jdata/reaxff/data (Argument)
    data:, 112
init_reaction_jdata/reaxff/dt (Argument)
    dt:, 113
init_reaction_jdata/reaxff/dump_freq (Argument)
    dump_freq:, 113
init_reaction_jdata/reaxff/ff (Argument)
    ff:, 113
init_reaction_jdata/reaxff/nstep (Argument)
    nstep:, 113
init_reaction_jdata/reaxff/tau_t (Argument)
    tau_t:, 113
init_reaction_jdata/reaxff/temp (Argument)
    temp:, 113
init_reaction_jdata/type_map (Argument)
    type_map:, 112
init_reaction_jdata:
    init_reaction_jdata (Argument), 112
init_reaction_jdata_arginfo() (in module dp-gen.data.arginfo), 276
init_reaction_mdata (Argument)
    init_reaction_mdata:, 114
init_reaction_mdata/api_version (Argument)
    api_version:, 114
init_reaction_mdata/build (Argument)
    build:, 127
init_reaction_mdata/build/command (Argument)
    command:, 127
init_reaction_mdata/build/machine (Argument)
    machine:, 127

```

init_reaction_mdata/build/machine/batch_type	init_reaction_mdata/build/machine[SSHContext]/remote_profile
(Argument)	(Argument)
batch_type:, 127	hostname:, 128
init_reaction_mdata/build/machine/clean_asynchronously	init_reaction_mdata/build/machine[SSHContext]/remote_profile
(Argument)	(Argument)
clean_asynchronously:, 128	key_filename:, 129
init_reaction_mdata/build/machine/context_type	init_reaction_mdata/build/machine[SSHContext]/remote_profile
(Argument)	(Argument)
context_type:, 128	look_for_keys:, 130
init_reaction_mdata/build/machine/local_root	init_reaction_mdata/build/machine[SSHContext]/remote_profile
(Argument)	(Argument)
local_root:, 127	passphrase:, 129
init_reaction_mdata/build/machine/remote_root	init_reaction_mdata/build/machine[SSHContext]/remote_profile
(Argument)	(Argument)
remote_root:, 127	password:, 129
init_reaction_mdata/build/machine[BohriumContext]/remote_profile	init_reaction_mdata/build/machine[SSHContext]/remote_profile
(Argument)	(Argument)
remote_profile:, 131	port:, 129
init_reaction_mdata/build/machine[BohriumContext]/remote_profile/email	init_reaction_mdata/build/machine[SSHContext]/remote_profile
(Argument)	(Argument)
email:, 131	tar_compress:, 130
init_reaction_mdata/build/machine[BohriumContext]/remote_profile/ignore_exit_code	init_reaction_mdata/build/machine[SSHContext]/remote_profile
(Argument)	(Argument)
ignore_exit_code:, 131	timeout:, 129
init_reaction_mdata/build/machine[BohriumContext]/remote_profile/input_data	init_reaction_mdata/build/machine[SSHContext]/remote_profile
(Argument)	(Argument)
input_data:, 132	totp_secret:, 129
init_reaction_mdata/build/machine[BohriumContext]/remote_profile/keep_backup	init_reaction_mdata/build/machine[SSHContext]/remote_profile
(Argument)	(Argument)
keep_backup:, 132	username:, 128
init_reaction_mdata/build/machine[BohriumContext]/remote_profile/password	init_reaction_mdata/build/resources (Argument)
(Argument)	(Argument)
password:, 131	resources:, 132
init_reaction_mdata/build/machine[BohriumContext]/remote_profile/resources/append_script	init_reaction_mdata/build/resources/append_script
(Argument)	(Argument)
program_id:, 131	append_script:, 135
init_reaction_mdata/build/machine[BohriumContext]/remote_profile/resources/batch_type	init_reaction_mdata/build/resources/batch_type
(Argument)	(Argument)
retry_count:, 131	batch_type:, 136
init_reaction_mdata/build/machine[HDFSContext]/remote_profile	init_reaction_mdata/build/resources/cpu_per_node
(Argument)	(Argument)
remote_profile:, 130	cpu_per_node:, 133
init_reaction_mdata/build/machine[LazyLocalContext]/remote_profile	init_reaction_mdata/build/resources/custom_flags
(Argument)	(Argument)
remote_profile:, 130	custom_flags:, 133
init_reaction_mdata/build/machine[LocalContext]/remote_profile	init_reaction_mdata/build/resources/envs (Argument)
(Argument)	(Argument)
remote_profile:, 132	envs:, 135
init_reaction_mdata/build/machine[OpenAPIContext]/remote_profile	init_reaction_mdata/build/resources/gpu_per_node
(Argument)	(Argument)
remote_profile:, 132	gpu_per_node:, 133
init_reaction_mdata/build/machine[SSHContext]/remote_profile	init_reaction_mdata/build/resources/group_size
(Argument)	(Argument)
remote_profile:, 128	group_size:, 133

init_reaction_mdata/build/resources/module_list custom_gpu_line:, 139
 (Argument) init_reaction_mdata/build/resources[LSF]/kwargs/gpu_exclus
 module_list:, 135 (Argument)
 init_reaction_mdata/build/resources/module_purge gpu_exclusive:, 139
 (Argument) init_reaction_mdata/build/resources[LSF]/kwargs/gpu_new_sy
 module_purge:, 134 (Argument)
 init_reaction_mdata/build/resources/module_unload dist_new_syntax:, 138
 (Argument) init_reaction_mdata/build/resources[LSF]/kwargs/gpu_usage
 module_unload_list:, 135 (Argument)
 init_reaction_mdata/build/resources/number_node gpu_usage:, 138
 (Argument) init_reaction_mdata/build/resources[OpenAPI]/kwargs
 number_node:, 133 (Argument)
 init_reaction_mdata/build/resources/para_deg kwargs:, 137
 (Argument) init_reaction_mdata/build/resources[PBS]/kwargs
 para_deg:, 134 (Argument)
 init_reaction_mdata/build/resources/prepend_script kwargs:, 136
 (Argument) init_reaction_mdata/build/resources[SGE]/kwargs
 prepend_script:, 135 (Argument)
 init_reaction_mdata/build/resources/queue_name kwargs:, 138
 (Argument) init_reaction_mdata/build/resources[Shell]/kwargs
 queue_name:, 133 (Argument)
 init_reaction_mdata/build/resources/source_list kwargs:, 136
 (Argument) init_reaction_mdata/build/resources[SlurmJobArray]/kwargs
 source_list:, 134 (Argument)
 init_reaction_mdata/build/resources/strategy kwargs:, 136
 (Argument) init_reaction_mdata/build/resources[SlurmJobArray]/kwargs/
 strategy:, 133 (Argument)
 init_reaction_mdata/build/resources/strategy/customized_script_header def_template_file
 (Argument) init_reaction_mdata/build/resources[SlurmJobArray]/kwargs/
 customized_script_header_template_file:, (Argument)
 134 slurm_job_size:, 137
 init_reaction_mdata/build/resources/strategy/include_multi_devices init_reaction_mdata/build/resources[Slurm]/kwargs
 (Argument) (Argument)
 if_cuda_multi_devices:, 134 kwargs:, 137
 init_reaction_mdata/build/resources/strategy/ratio_unfinished init_reaction_mdata/build/resources[Slurm]/kwargs/custom_g
 (Argument) (Argument)
 ratio_unfinished:, 134 custom_gpu_line:, 137
 init_reaction_mdata/build/resources/wait_time init_reaction_mdata/build/resources[Torque]/kwargs
 (Argument) (Argument)
 wait_time:, 135 kwargs:, 138
 init_reaction_mdata/build/resources[Bohrium]/kwargs init_reaction_mdata/build/user_backward_files
 (Argument) (Argument)
 kwargs:, 137 user_backward_files:, 139
 init_reaction_mdata/build/resources[DistributedShell]/kwargs init_reaction_mdata/build/user_forward_files
 (Argument) (Argument)
 kwargs:, 136 user_forward_files:, 139
 init_reaction_mdata/build/resources[Fugaku]/kwargs init_reaction_mdata/deepmd_version (Argument)
 (Argument) deepmd_version:, 114
 kwargs:, 138 init_reaction_mdata/fp (Argument)
 init_reaction_mdata/build/resources[LSF]/kwargs fp:, 139
 (Argument) init_reaction_mdata/fp/command (Argument)
 kwargs:, 138 command:, 139
 init_reaction_mdata/build/resources[LSF]/kwargs/init_reaction_mdata/fp/machine (Argument)
 (Argument) machine:, 139

init_reaction_mdata/fp/machine/batch_type (Argument) batch_type:, 140	init_reaction_mdata/fp/machine[SSHContext]/remote_profile/ (Argument) hostname:, 141
init_reaction_mdata/fp/machine/clean_asynchronously: (Argument) clean_asynchronously:, 140	init_reaction_mdata/fp/machine[SSHContext]/remote_profile/ (Argument) key_filename:, 141
init_reaction_mdata/fp/machine/context_type (Argument) context_type:, 140	init_reaction_mdata/fp/machine[SSHContext]/remote_profile/ (Argument) look_for_keys:, 142
init_reaction_mdata/fp/machine/local_root (Argument) local_root:, 140	init_reaction_mdata/fp/machine[SSHContext]/remote_profile/ (Argument) passphrase:, 142
init_reaction_mdata/fp/machine/remote_root (Argument) remote_root:, 140	init_reaction_mdata/fp/machine[SSHContext]/remote_profile/ (Argument) password:, 141
init_reaction_mdata/fp/machine[BohriumContext]/remote_profile: (Argument) remote_profile:, 143	init_reaction_mdata/fp/machine[SSHContext]/remote_profile/ (Argument) port:, 141
init_reaction_mdata/fp/machine[BohriumContext]/remote_profile/email: (Argument) email:, 143	init_reaction_mdata/fp/machine[SSHContext]/remote_profile/ (Argument) tar_compress:, 142
init_reaction_mdata/fp/machine[BohriumContext]/remote_profile/ignore_exit_code: (Argument) ignore_exit_code:, 144	init_reaction_mdata/fp/machine[SSHContext]/remote_profile/ (Argument) timeout:, 142
init_reaction_mdata/fp/machine[BohriumContext]/remote_profile/input_data: (Argument) input_data:, 144	init_reaction_mdata/fp/machine[SSHContext]/remote_profile/ (Argument) totp_secret:, 142
init_reaction_mdata/fp/machine[BohriumContext]/remote_profile/keep_backup: (Argument) keep_backup:, 144	init_reaction_mdata/fp/machine[SSHContext]/remote_profile/ (Argument) username:, 141
init_reaction_mdata/fp/machine[BohriumContext]/remote_profile/password: (Argument) password:, 143	init_reaction_mdata/fp/resources (Argument) resources:, 145
init_reaction_mdata/fp/machine[BohriumContext]/remote_profile/program_id: (Argument) program_id:, 144	init_reaction_mdata/fp/resources/append_script (Argument) append_script:, 148
init_reaction_mdata/fp/machine[BohriumContext]/remote_profile/retry_count: (Argument) retry_count:, 144	init_reaction_mdata/fp/resources/batch_type (Argument) batch_type:, 148
init_reaction_mdata/fp/machine[HDFSText]/remote_profile: (Argument) remote_profile:, 143	init_reaction_mdata/fp/resources/cpu_per_node (Argument) cpu_per_node:, 145
init_reaction_mdata/fp/machine[LazyLocalContext]/remote_profile: (Argument) remote_profile:, 143	init_reaction_mdata/fp/resources/custom_flags (Argument) custom_flags:, 146
init_reaction_mdata/fp/machine[LocalContext]/remote_profile: (Argument) remote_profile:, 145	init_reaction_mdata/fp/resources/envs (Argument) envs:, 147
init_reaction_mdata/fp/machine[OpenAPIContext]/remote_profile: (Argument) remote_profile:, 145	init_reaction_mdata/fp/resources/gpu_per_node (Argument) gpu_per_node:, 145
init_reaction_mdata/fp/machine[SSHContext]/remote_profile: (Argument) remote_profile:, 141	init_reaction_mdata/fp/resources/group_size (Argument) group_size:, 146
	init_reaction_mdata/fp/resources/module_list

```

        (Argument)
    module_list:, 147
init_reaction_mdata/fp/resources/module_purge
    (Argument)
    module_purge:, 147
init_reaction_mdata/fp/resources/module_unload_list
    (Argument)
    module_unload_list:, 147
init_reaction_mdata/fp/resources/number_node
    (Argument)
    number_node:, 145
init_reaction_mdata/fp/resources/para_deg
    (Argument)
    para_deg:, 147
init_reaction_mdata/fp/resources/prepend_script
    (Argument)
    prepend_script:, 148
init_reaction_mdata/fp/resources/queue_name
    (Argument)
    queue_name:, 145
init_reaction_mdata/fp/resources/source_list
    (Argument)
    source_list:, 147
init_reaction_mdata/fp/resources/strategy
    (Argument)
    strategy:, 146
init_reaction_mdata/fp/resources/strategy/customized_script_header_template_file
    (Argument)
    customized_script_header_template_file:,
    146
init_reaction_mdata/fp/resources/strategy/if_cuda_multi_devices
    (Argument)
    if_cuda_multi_devices:, 146
init_reaction_mdata/fp/resources/strategy/ratio_unfinished
    (Argument)
    ratio_unfinished:, 146
init_reaction_mdata/fp/resources/wait_time
    (Argument)
    wait_time:, 148
init_reaction_mdata/fp/resources[Bohrium]/kwargs
    (Argument)
    kwargs:, 150
init_reaction_mdata/fp/resources[DistributedShell]/kwargs
    (Argument)
    kwargs:, 149
init_reaction_mdata/fp/resources[Fugaku]/kwargs
    (Argument)
    kwargs:, 150
init_reaction_mdata/fp/resources[LSF]/kwargs
    (Argument)
    kwargs:, 150
init_reaction_mdata/fp/resources[LSF]/kwargs/custom_gpu_line
    (Argument)
    custom_gpu_line:, 151
init_reaction_mdata/fp/resources[LSF]/kwargs/gpu_exclusive
    (Argument)
    gpu_exclusive:, 151
init_reaction_mdata/fp/resources[LSF]/kwargs/gpu_new_syntax
    (Argument)
    gpu_new_syntax:, 151
init_reaction_mdata/fp/resources[LSF]/kwargs/gpu_usage
    (Argument)
    gpu_usage:, 151
init_reaction_mdata/fp/resources[OpenAPI]/kwargs
    (Argument)
    kwargs:, 150
init_reaction_mdata/fp/resources[PBS]/kwargs
    (Argument)
    kwargs:, 149
init_reaction_mdata/fp/resources[SGE]/kwargs
    (Argument)
    kwargs:, 150
init_reaction_mdata/fp/resources[Shell]/kwargs
    (Argument)
    kwargs:, 148
init_reaction_mdata/fp/resources[SlurmJobArray]/kwargs
    (Argument)
    kwargs:, 149
init_reaction_mdata/fp/resources[SlurmJobArray]/kwargs/custom_gpu_line
    (Argument)
    custom_gpu_line:, 149
init_reaction_mdata/fp/resources[SlurmJobArray]/kwargs/slurm_job_size
    (Argument)
    slurm_job_size:, 149
init_reaction_mdata/fp/resources[Slurm]/kwargs
    (Argument)
    kwargs:, 149
init_reaction_mdata/fp/resources[Slurm]/kwargs/custom_gpu_line
    (Argument)
    custom_gpu_line:, 149
init_reaction_mdata/fp/resources[Torque]/kwargs
    (Argument)
    kwargs:, 150
init_reaction_mdata/fp/user_backward_files
    (Argument)
    user_backward_files:, 151
init_reaction_mdata/fp/user_forward_files
    (Argument)
    user_forward_files:, 151
init_reaction_mdata/reaxff (Argument)
    reaxff:, 114
init_reaction_mdata/reaxff/command (Argument)
    command:, 114
init_reaction_mdata/reaxff/machine (Argument)
    machine:, 115
init_reaction_mdata/reaxff/machine/batch_type
    (Argument)
    batch_type:, 115

```


init_reaction_mdata/reaxff/machine/clean_asynchronously:	init_reaction_mdata/reaxff/machine[SSHContext]/remote_profile:
(Argument)	(Argument)
clean_asynchronously:, 115	key_filename:, 116
init_reaction_mdata/reaxff/machine/context_type:	init_reaction_mdata/reaxff/machine[SSHContext]/remote_profile:
(Argument)	(Argument)
context_type:, 115	look_for_keys:, 117
init_reaction_mdata/reaxff/machine/local_root:	init_reaction_mdata/reaxff/machine[SSHContext]/remote_profile:
(Argument)	(Argument)
local_root:, 115	passphrase:, 117
init_reaction_mdata/reaxff/machine/remote_root:	init_reaction_mdata/reaxff/machine[SSHContext]/remote_profile:
(Argument)	(Argument)
remote_root:, 115	password:, 116
init_reaction_mdata/reaxff/machine[BohriumContext]/remote_profile:	init_reaction_mdata/reaxff/machine[SSHContext]/remote_profile:
(Argument)	(Argument)
remote_profile:, 118	port:, 116
init_reaction_mdata/reaxff/machine[BohriumContext]/remote_profile:	init_reaction_mdata/reaxff/machine[SSHContext]/remote_profile:
(Argument)	(Argument)
email:, 118	tar_compress:, 117
init_reaction_mdata/reaxff/machine[BohriumContext]/remote_profile:	init_reaction_mdata/reaxff/machine[SSHContext]/remote_profile:
(Argument)	(Argument)
ignore_exit_code:, 119	timeout:, 117
init_reaction_mdata/reaxff/machine[BohriumContext]/remote_profile:	init_reaction_mdata/reaxff/machine[SSHContext]/remote_profile:
(Argument)	(Argument)
input_data:, 119	totp_secret:, 117
init_reaction_mdata/reaxff/machine[BohriumContext]/remote_profile:	init_reaction_mdata/reaxff/machine[SSHContext]/remote_profile:
(Argument)	(Argument)
keep_backup:, 119	username:, 116
init_reaction_mdata/reaxff/machine[BohriumContext]/remote_profile:	init_reaction_mdata/reaxff/resources (Argument):
(Argument)	password:, 118
password:, 118	resources:, 120
init_reaction_mdata/reaxff/machine[BohriumContext]/remote_profile:	init_reaction_mdata/reaxff/resources/append_script:
(Argument)	(Argument)
program_id:, 119	append_script:, 123
init_reaction_mdata/reaxff/machine[BohriumContext]/remote_profile:	init_reaction_mdata/reaxff/resources/batch_type:
(Argument)	(Argument)
retry_count:, 119	batch_type:, 123
init_reaction_mdata/reaxff/machine[HDFSContext]/remote_profile:	init_reaction_mdata/reaxff/resources/cpu_per_node:
(Argument)	(Argument)
remote_profile:, 118	cpu_per_node:, 120
init_reaction_mdata/reaxff/machine[LazyLocalContext]/remote_profile:	init_reaction_mdata/reaxff/resources/custom_flags:
(Argument)	(Argument)
remote_profile:, 118	custom_flags:, 121
init_reaction_mdata/reaxff/machine[LocalContext]/remote_profile:	init_reaction_mdata/reaxff/resources/envs:
(Argument)	(Argument)
remote_profile:, 120	envs:, 122
init_reaction_mdata/reaxff/machine[OpenAPIContext]/remote_profile:	init_reaction_mdata/reaxff/resources/gpu_per_node:
(Argument)	(Argument)
remote_profile:, 120	gpu_per_node:, 120
init_reaction_mdata/reaxff/machine[SSHContext]/remote_profile:	init_reaction_mdata/reaxff/resources/group_size:
(Argument)	(Argument)
remote_profile:, 116	group_size:, 121
init_reaction_mdata/reaxff/machine[SSHContext]/remote_profile:	init_reaction_mdata/reaxff/resources/module_list:
(Argument)	(Argument)
hostname:, 116	module_list:, 122

init_reaction_mdata/reaxff/resources/module_purge gpu_exclusive:, 126
 (Argument) init_reaction_mdata/reaxff/resources[LSF]/kwargs/gpu_new_syntax:, 126
 module_purge:, 122 (Argument)
 init_reaction_mdata/reaxff/resources/module_unload_list:, 126
 (Argument) init_reaction_mdata/reaxff/resources[LSF]/kwargs/gpu_usage:, 126
 module_unload_list:, 122 (Argument)
 init_reaction_mdata/reaxff/resources/number_node gpu_usage:, 126
 (Argument) init_reaction_mdata/reaxff/resources[OpenAPI]/kwargs
 number_node:, 120 (Argument)
 init_reaction_mdata/reaxff/resources/para_deg kwargs:, 125
 (Argument) init_reaction_mdata/reaxff/resources[PBS]/kwargs
 para_deg:, 122 (Argument)
 init_reaction_mdata/reaxff/resources/prepend_script kwargs:, 124
 (Argument) init_reaction_mdata/reaxff/resources[SGE]/kwargs
 prepend_script:, 123 (Argument)
 init_reaction_mdata/reaxff/resources/queue_name kwargs:, 125
 (Argument) init_reaction_mdata/reaxff/resources[Shell]/kwargs
 queue_name:, 121 (Argument)
 init_reaction_mdata/reaxff/resources/source_list kwargs:, 123
 (Argument) init_reaction_mdata/reaxff/resources[SlurmJobArray]/kwargs
 source_list:, 122 (Argument)
 init_reaction_mdata/reaxff/resources/strategy kwargs:, 124
 (Argument) init_reaction_mdata/reaxff/resources[SlurmJobArray]/kwargs
 strategy:, 121 (Argument)
 init_reaction_mdata/reaxff/resources/strategy/customized_script_header_template_file
 (Argument) init_reaction_mdata/reaxff/resources[SlurmJobArray]/kwargs
 customized_script_header_template_file:, (Argument)
 122 slurm_job_size:, 124
 init_reaction_mdata/reaxff/resources/strategy/if_cuda_multi_devices init_reaction_mdata/reaxff/resources[Slurm]/kwargs
 (Argument) (Argument)
 if_cuda_multi_devices:, 121 kwargs:, 124
 init_reaction_mdata/reaxff/resources/strategy/init_ratio_unfinished init_reaction_mdata/reaxff/resources[Slurm]/kwargs/custom
 (Argument) (Argument)
 ratio_unfinished:, 121 custom_gpu_line:, 125
 init_reaction_mdata/reaxff/resources/wait_time init_reaction_mdata/reaxff/resources[Torque]/kwargs
 (Argument) (Argument)
 wait_time:, 123 kwargs:, 125
 init_reaction_mdata/reaxff/resources[Bohrium]/kwargs init_reaction_mdata/reaxff/user_backward_files
 (Argument) (Argument)
 kwargs:, 125 user_backward_files:, 127
 init_reaction_mdata/reaxff/resources[DistributedShell]/kwargs init_reaction_mdata/reaxff/user_forward_files
 (Argument) (Argument)
 kwargs:, 124 user_forward_files:, 126
 init_reaction_mdata/reaxff/resources[Fugaku]/kwargs init_reaction_mdata:
 (Argument) init_reaction_mdata (Argument), 114
 kwargs:, 125 init_reaction_mdata_arginfo() (in module dp-
 init_reaction_mdata/reaxff/resources[LSF]/kwargs gen.data.arginfo), 276
 (Argument) init_surf_jdata (Argument)
 kwargs:, 126 init_surf_jdata:, 96
 init_reaction_mdata/reaxff/resources[LSF]/kwargs/init_surf_jdata/coll_ndata (Argument)
 (Argument) (Argument)
 custom_gpu_line:, 126 cell_type:, 96
 init_reaction_mdata/reaxff/resources[LSF]/kwargs/init_surf_jdata/coll_ndata (Argument)
 (Argument) (Argument)
 kwargs/gpu_exclusive:, 126 init_surf_jdata/elements (Argument)

elements:, 96
 init_surf_jdata/from_poscar (Argument)
 from_poscar:, 97
 init_surf_jdata/from_poscar_path (Argument)
 from_poscar_path:, 97
 init_surf_jdata/head_ratio (Argument)
 head_ratio:, 98
 init_surf_jdata/latt (Argument)
 latt:, 97
 init_surf_jdata/layer_numb (Argument)
 layer_numb:, 97
 init_surf_jdata/mid_point (Argument)
 mid_point:, 98
 init_surf_jdata/millers (Argument)
 millers:, 98
 init_surf_jdata/pert_atom (Argument)
 pert_atom:, 99
 init_surf_jdata/pert_box (Argument)
 pert_box:, 98
 init_surf_jdata/pert_numb (Argument)
 pert_numb:, 98
 init_surf_jdata/potcars (Argument)
 potcars:, 96
 init_surf_jdata/relax_incar (Argument)
 relax_incar:, 98
 init_surf_jdata/scale (Argument)
 scale:, 98
 init_surf_jdata/skip_relax (Argument)
 skip_relax:, 98
 init_surf_jdata/stages (Argument)
 stages:, 96
 init_surf_jdata/super_cell (Argument)
 super_cell:, 96
 init_surf_jdata/vacuum_max (Argument)
 vacuum_max:, 97
 init_surf_jdata/vacuum_min (Argument)
 vacuum_min:, 97
 init_surf_jdata/vacuum_numb (Argument)
 vacuum_numb:, 97
 init_surf_jdata/vacuum_resol (Argument)
 vacuum_resol:, 97
 init_surf_jdata/z_min (Argument)
 z_min:, 97
 init_surf_jdata:
 init_surf_jdata (Argument), 96
 init_surf_jdata_arginfo() (in module dp-
 gen.data_arginfo), 276
 init_surf_mdata (Argument)
 init_surf_mdata:, 99
 init_surf_mdata/api_version (Argument)
 api_version:, 99
 init_surf_mdata/deepmd_version (Argument)
 deepmd_version:, 99
 init_surf_mdata/fp (Argument)

 fp:, 99
 init_surf_mdata/fp/command (Argument)
 command:, 99
 init_surf_mdata/fp/machine (Argument)
 machine:, 99
 init_surf_mdata/fp/machine/batch_type (Argument)
 batch_type:, 100
 init_surf_mdata/fp/machine/clean_asynchronously
 (Argument)
 clean_asynchronously:, 100
 init_surf_mdata/fp/machine/context_type
 (Argument)
 context_type:, 100
 init_surf_mdata/fp/machine/local_root (Argument)
 local_root:, 100
 init_surf_mdata/fp/machine/remote_root (Argument)
 remote_root:, 100
 init_surf_mdata/fp/machine[BohriumContext]/remote_profile
 (Argument)
 remote_profile:, 103
 init_surf_mdata/fp/machine[BohriumContext]/remote_profile/
 (Argument)
 email:, 103
 init_surf_mdata/fp/machine[BohriumContext]/remote_profile/
 (Argument)
 ignore_exit_code:, 104
 init_surf_mdata/fp/machine[BohriumContext]/remote_profile/
 (Argument)
 input_data:, 104
 init_surf_mdata/fp/machine[BohriumContext]/remote_profile/
 (Argument)
 keep_backup:, 104
 init_surf_mdata/fp/machine[BohriumContext]/remote_profile/
 (Argument)
 password:, 103
 init_surf_mdata/fp/machine[BohriumContext]/remote_profile/
 (Argument)
 program_id:, 103
 init_surf_mdata/fp/machine[BohriumContext]/remote_profile/
 (Argument)
 retry_count:, 103
 init_surf_mdata/fp/machine[HDF5Context]/remote_profile
 (Argument)
 remote_profile:, 103
 init_surf_mdata/fp/machine[LazyLocalContext]/remote_profile
 (Argument)
 remote_profile:, 102
 init_surf_mdata/fp/machine[LocalContext]/remote_profile
 (Argument)
 remote_profile:, 104
 init_surf_mdata/fp/machine[OpenAPIContext]/remote_profile

```

        (Argument)
remote_profile:, 104
init_surf_mdata/fp/machine[SSHContext]/remote_profile_size:, 105
        (Argument)
remote_profile:, 101
init_surf_mdata/fp/machine[SSHContext]/remote_profile_size:, 107
        (Argument)
hostname:, 101
init_surf_mdata/fp/machine[SSHContext]/remote_profile_size:, 107
        (Argument)
key_filename:, 101
init_surf_mdata/fp/machine[SSHContext]/remote_profile_size:, 107
        (Argument)
look_for_keys:, 102
init_surf_mdata/fp/machine[SSHContext]/remote_profile_size:, 105
        (Argument)
passphrase:, 102
init_surf_mdata/fp/machine[SSHContext]/remote_profile_size:, 105
        (Argument)
password:, 101
init_surf_mdata/fp/machine[SSHContext]/remote_profile_size:, 107
        (Argument)
port:, 101
init_surf_mdata/fp/machine[SSHContext]/remote_profile_size:, 105
        (Argument)
tar_compress:, 102
init_surf_mdata/fp/machine[SSHContext]/remote_profile_size:, 106
        (Argument)
timeout:, 102
init_surf_mdata/fp/machine[SSHContext]/remote_profile_size:, 105
        (Argument)
totp_secret:, 102
init_surf_mdata/fp/machine[SSHContext]/remote_profile_size:, 106
        (Argument)
username:, 101
init_surf_mdata/fp/resources (Argument)
resources:, 105
init_surf_mdata/fp/resources/append_script
        (Argument)
append_script:, 107
init_surf_mdata/fp/resources/batch_type
        (Argument)
batch_type:, 108
init_surf_mdata/fp/resources/cpu_per_node
        (Argument)
cpu_per_node:, 105
init_surf_mdata/fp/resources/custom_flags
        (Argument)
custom_flags:, 105
init_surf_mdata/fp/resources/envs (Argument)
envs:, 107
init_surf_mdata/fp/resources/gpu_per_node
        (Argument)
gpu_per_node:, 105
init_surf_mdata/fp/resources/group_size
        (Argument)
group_size:, 105
init_surf_mdata/fp/resources/module_list (Argument)
init_surf_mdata/fp/resources/module_purge
        (Argument)
init_surf_mdata/fp/resources/module_unload_list
        (Argument)
init_surf_mdata/fp/resources/number_node (Argument)
init_surf_mdata/fp/resources/para_deg (Argument)
init_surf_mdata/fp/resources/para_size
        (Argument)
init_surf_mdata/fp/resources/prepend_script
        (Argument)
init_surf_mdata/fp/resources/queue_name
        (Argument)
init_surf_mdata/fp/resources/source_list (Argument)
init_surf_mdata/fp/resources/strategy (Argument)
init_surf_mdata/fp/resources/strategy/customized_script_header
        (Argument)
init_surf_mdata/fp/resources/strategy/if_cuda_multi_device
        (Argument)
if_cuda_multi_devices:, 106
init_surf_mdata/fp/resources/strategy/ratio_unfinished
        (Argument)
ratio_unfinished:, 106
init_surf_mdata/fp/resources/wait_time (Argument)
wait_time:, 107
init_surf_mdata/fp/resources[Bohrium]/kwargs
        (Argument)
kwargs:, 109
init_surf_mdata/fp/resources[DistributedShell]/kwargs
        (Argument)
kwargs:, 108
init_surf_mdata/fp/resources[Fugaku]/kwargs
        (Argument)
kwargs:, 110
init_surf_mdata/fp/resources[LSF]/kwargs (Argument)

```

kwargs:, 110
 init_surf_mdata/fp/resources[LSF]/kwargs/custom_gpu_line: (Argument), 86
 custom_gpu_line:, 111
 init_surf_mdata/fp/resources[LSF]/kwargs/gpu_exclusive: (Argument), 132
 gpu_exclusive:, 111
 init_surf_mdata/fp/resources[LSF]/kwargs/gpu_new_syntax: (Argument), 144
 gpu_new_syntax:, 119
 init_surf_mdata/fp/resources[LSF]/kwargs/gpu_usage: (Argument), 119
 gpu_usage:, 110
 init_surf_mdata/fp/resources[OpenAPI]/kwargs (Argument), 104
 kwargs:, 109
 init_surf_mdata/fp/resources[PBS]/kwargs (Argument), 69
 kwargs:, 108
 init_surf_mdata/fp/resources[SGE]/kwargs (Argument), 57
 kwargs:, 110
 init_surf_mdata/fp/resources[Shell]/kwargs (Argument), 45
 kwargs:, 108
 init_surf_mdata/fp/resources[SlurmJobArray]/kwargs (Argument), 198
 kwargs:, 108
 init_surf_mdata/fp/resources[SlurmJobArray]/kwargs/custom_gpu_line: (Argument), 186
 custom_gpu_line:, 109
 init_surf_mdata/fp/resources[SlurmJobArray]/kwargs/slurm_job_size: (Argument), 174
 slurm_job_size:, 109
 init_surf_mdata/fp/resources[Slurm]/kwargs (Argument), 39
 kwargs:, 109
 init_surf_mdata/fp/resources[Slurm]/kwargs/custom_gpu_line: (Argument), 39
 custom_gpu_line:, 109
 init_surf_mdata/fp/resources[Torque]/kwargs (Argument), 168
 kwargs:, 110
 init_surf_mdata/fp/user_backward_files (Argument), 290
 user_backward_files:, 111
 init_surf_mdata/fp/user_forward_files (Argument), 258
 user_forward_files:, 111
 init_surf_mdata: (Argument), 300
 init_surf_mdata_arginfo() (in module dp-gen.data.arginfo), 275
 input_data:

init_bulk_mdata/fp/machine[BohriumContext]/remote_profile (Argument), 86
 init_reaction_mdata/build/machine[BohriumContext]/remote_profile (Argument), 132
 init_reaction_mdata/fp/machine[BohriumContext]/remote_profile (Argument), 144
 init_reaction_mdata/reaxff/machine[BohriumContext]/remote_profile (Argument), 119
 init_surf_mdata/fp/machine[BohriumContext]/remote_profile (Argument), 104
 run_mdata/fp/machine[BohriumContext]/remote_profile/in (Argument), 69
 run_mdata/model_devi/machine[BohriumContext]/remote_profile (Argument), 57
 run_mdata/train/machine[BohriumContext]/remote_profile (Argument), 45
 simplify_mdata/fp/machine[BohriumContext]/remote_profile (Argument), 198
 simplify_mdata/model_devi/machine[BohriumContext]/remote_profile (Argument), 186
 simplify_mdata/train/machine[BohriumContext]/remote_profile (Argument), 174
 input_fmt: run_jdata[fp_style=custom]/fp_params/input_fmt (Argument), 39
 simplify_jdata[custom]/fp_params/input_fmt (Argument), 168
 input_fn: run_jdata[fp_style=custom]/fp_params/input_fn (Argument), 39
 simplify_jdata[custom]/fp_params/input_fn (Argument), 168
 input_upper() (in module dpgen.generator.lib.pwmat), 290
 insert_data() (in module dpgen.auto_test.lib.util), 258
 inter_deepmd() (in module dpgen.auto_test.lib.lammps), 252
 inter_deepmd_in() (in module dpgen.auto_test.lib.lammps), 252
 inter_eam_fs() (in module dpgen.auto_test.lib.lammps), 252
 inter_meam() (in module dpgen.auto_test.lib.lammps), 252
 Interstitial (class in dpgen.auto_test.Interstitial), 264
 is_upper_triangular() (in module dpgen.data.tools.io_lammps), 275
 iter_pick_number: simplify_jdata/iter_pick_number (Argument), 157
 Iteration (class in dpgen.tools.auto_gen_param), 300
 iterdict() (in module dpgen.generator.lib.cp2k), 289

K

k_points:


```

run_jdata[fp_style=abacus]/k_points (Argument), 162
(Argument), 37 keywords_high_multiplicity:
simplify_jdata[abacus]/k_points (Argument), 166 run_jdata[fp_style=gaussian]/fp_params/keywords_high_multi-
(Argument), 34
keep_backup: simplify_jdata[gaussian]/fp_params/keywords_high_multi-
(Argument), 166
init_bulk_mdata/fp/machine[BohriumContext]/remote_profile/keep_backup
(Argument), 86 kspacing:
init_reaction_mdata/build/machine[BohriumContext]/remote_profile/keep_backup
(Argument), 132 (Argument), 39
init_reaction_mdata/fp/machine[BohriumContext]/remote_profile/keep_backup
(Argument), 144 (Argument), 35
init_reaction_mdata/reaxff/machine[BohriumContext]/remote_profile/keep_backup
(Argument), 119 (Argument), 167
init_surf_mdata/fp/machine[BohriumContext]/remote_profile/keep_backup
(Argument), 104 (Argument), 164
run_mdata/fp/machine[BohriumContext]/remote_profile/keep_backup
(Argument), 69 init_bulk_mdata/fp/resources[Bohrium]/kwargs
run_mdata/model_devi/machine[BohriumContext]/remote_profile/keep_backup
(Argument), 57 init_bulk_mdata/fp/resources[DistributedShell]/kwargs
run_mdata/train/machine[BohriumContext]/remote_profile/keep_backup
(Argument), 45 init_bulk_mdata/fp/resources[Fugaku]/kwargs
simplify_mdata/fp/machine[BohriumContext]/remote_profile/keep_backup
(Argument), 198 init_bulk_mdata/fp/resources[LSF]/kwargs
simplify_mdata/model_devi/machine[BohriumContext]/remote_profile/keep_backup
(Argument), 186 init_bulk_mdata/fp/resources[OpenAPI]/kwargs
simplify_mdata/train/machine[BohriumContext]/remote_profile/keep_backup
(Argument), 173 init_bulk_mdata/fp/resources[PBS]/kwargs
key_filename: (Argument), 90
init_bulk_mdata/fp/machine[SSHContext]/remote_profile/key_filename
(Argument), 83 init_bulk_mdata/fp/resources[SGE]/kwargs
(Argument), 91
init_reaction_mdata/build/machine[SSHContext]/remote_profile/key_filename
(Argument), 129 init_bulk_mdata/fp/resources[Shell]/kwargs
(Argument), 90
init_reaction_mdata/fp/machine[SSHContext]/remote_profile/key_filename
(Argument), 141 init_bulk_mdata/fp/resources[SlurmJobArray]/kwargs
(Argument), 90
init_reaction_mdata/reaxff/machine[SSHContext]/remote_profile/key_filename
(Argument), 116 init_bulk_mdata/fp/resources[Slurm]/kwargs
(Argument), 91
init_surf_mdata/fp/machine[SSHContext]/remote_profile/key_filename
(Argument), 101 init_bulk_mdata/fp/resources[Torque]/kwargs
(Argument), 92
run_mdata/fp/machine[SSHContext]/remote_profile/key_filename
(Argument), 66 init_reaction_mdata/build/resources[Bohrium]/kwargs
(Argument), 137
run_mdata/model_devi/machine[SSHContext]/remote_profile/key_filename
(Argument), 54 init_reaction_mdata/build/resources[DistributedShell]/kwargs
(Argument), 136
run_mdata/train/machine[SSHContext]/remote_profile/key_filename
(Argument), 42 init_reaction_mdata/build/resources[Fugaku]/kwargs
(Argument), 138
simplify_mdata/fp/machine[SSHContext]/remote_profile/key_filename
(Argument), 195 init_reaction_mdata/build/resources[LSF]/kwargs
(Argument), 138
simplify_mdata/model_devi/machine[SSHContext]/remote_profile/key_filename
(Argument), 183 init_reaction_mdata/build/resources[OpenAPI]/kwargs
(Argument), 137
simplify_mdata/train/machine[SSHContext]/remote_profile/key_filename
(Argument), 171 init_reaction_mdata/build/resources[PBS]/kwargs
(Argument), 136
keywords: init_reaction_mdata/build/resources[SGE]/kwargs
run_jdata[fp_style=gaussian]/fp_params/keywords (Argument), 138
(Argument), 33 init_reaction_mdata/build/resources[Shell]/kwargs
simplify_jdata[gaussian]/fp_params/keywords (Argument), 136

```

init_reaction_mdata/build/resources[SlurmJobArray]/kwargs
 (Argument), 136
 init_reaction_mdata/build/resources[Slurm]/kwargs
 (Argument), 137
 init_reaction_mdata/build/resources[Torque]/kwargs
 (Argument), 138
 init_reaction_mdata/fp/resources[Bohrium]/kwargs
 (Argument), 150
 init_reaction_mdata/fp/resources[DistributedShell]/kwargs
 (Argument), 149
 init_reaction_mdata/fp/resources[Fugaku]/kwargs
 (Argument), 150
 init_reaction_mdata/fp/resources[LSF]/kwargs
 (Argument), 150
 init_reaction_mdata/fp/resources[OpenAPI]/kwargs
 (Argument), 150
 init_reaction_mdata/fp/resources[PBS]/kwargs
 (Argument), 149
 init_reaction_mdata/fp/resources[SGE]/kwargs
 (Argument), 150
 init_reaction_mdata/fp/resources[Shell]/kwargs
 (Argument), 148
 init_reaction_mdata/fp/resources[SlurmJobArray]/kwargs
 (Argument), 149
 init_reaction_mdata/fp/resources[Slurm]/kwargs
 (Argument), 149
 init_reaction_mdata/fp/resources[Torque]/kwargs
 (Argument), 150
 init_reaction_mdata/reaxff/resources[Bohrium]/kwargs
 (Argument), 125
 init_reaction_mdata/reaxff/resources[DistributedShell]/kwargs
 (Argument), 124
 init_reaction_mdata/reaxff/resources[Fugaku]/kwargs
 (Argument), 125
 init_reaction_mdata/reaxff/resources[LSF]/kwargs
 (Argument), 126
 init_reaction_mdata/reaxff/resources[OpenAPI]/kwargs
 (Argument), 125
 init_reaction_mdata/reaxff/resources[PBS]/kwargs
 (Argument), 124
 init_reaction_mdata/reaxff/resources[SGE]/kwargs
 (Argument), 125
 init_reaction_mdata/reaxff/resources[Shell]/kwargs
 (Argument), 123
 init_reaction_mdata/reaxff/resources[SlurmJobArray]/kwargs
 (Argument), 124
 init_reaction_mdata/reaxff/resources[Slurm]/kwargs
 (Argument), 124
 init_reaction_mdata/reaxff/resources[Torque]/kwargs
 (Argument), 125
 init_surf_mdata/fp/resources[Bohrium]/kwargs
 (Argument), 109
 init_surf_mdata/fp/resources[DistributedShell]/kwargs
 (Argument), 108

<code>run_mdata/model_devi/resources[Shell]/kwargs</code> (Argument), 61	<code>simplify_mdata/model_devi/resources[DistributedShell]/</code> (Argument), 190
<code>run_mdata/model_devi/resources[SlurmJobArray]/kwargs</code> (Argument), 61	<code>simplify_mdata/model_devi/resources[Fugaku]/kwargs</code> (Argument), 192
<code>run_mdata/model_devi/resources[Slurm]/kwargs</code> (Argument), 62	<code>simplify_mdata/model_devi/resources[LSF]/kwargs</code> (Argument), 192
<code>run_mdata/model_devi/resources[Torque]/kwargs</code> (Argument), 63	<code>simplify_mdata/model_devi/resources[OpenAPI]/kwargs</code> (Argument), 191
<code>run_mdata/train/resources[Bohrium]/kwargs</code> (Argument), 50	<code>simplify_mdata/model_devi/resources[PBS]/kwargs</code> (Argument), 190
<code>run_mdata/train/resources[DistributedShell]/kwargs</code> (Argument), 49	<code>simplify_mdata/model_devi/resources[SGE]/kwargs</code> (Argument), 192
<code>run_mdata/train/resources[Fugaku]/kwargs</code> (Argument), 51	<code>simplify_mdata/model_devi/resources[Shell]/kwargs</code> (Argument), 190
<code>run_mdata/train/resources[LSF]/kwargs</code> (Argument), 51	<code>simplify_mdata/model_devi/resources[SlurmJobArray]/kwargs</code> (Argument), 190
<code>run_mdata/train/resources[OpenAPI]/kwargs</code> (Argument), 50	<code>simplify_mdata/model_devi/resources[Slurm]/kwargs</code> (Argument), 191
<code>run_mdata/train/resources[PBS]/kwargs</code> (Argument), 49	<code>simplify_mdata/model_devi/resources[Torque]/kwargs</code> (Argument), 192
<code>run_mdata/train/resources[SGE]/kwargs</code> (Argument), 50	<code>simplify_mdata/train/resources[Bohrium]/kwargs</code> (Argument), 179
<code>run_mdata/train/resources[Shell]/kwargs</code> (Argument), 49	<code>simplify_mdata/train/resources[DistributedShell]/kwargs</code> (Argument), 178
<code>run_mdata/train/resources[SlurmJobArray]/kwargs</code> (Argument), 49	<code>simplify_mdata/train/resources[Fugaku]/kwargs</code> (Argument), 179
<code>run_mdata/train/resources[Slurm]/kwargs</code> (Argument), 50	<code>simplify_mdata/train/resources[LSF]/kwargs</code> (Argument), 180
<code>run_mdata/train/resources[Torque]/kwargs</code> (Argument), 50	<code>simplify_mdata/train/resources[OpenAPI]/kwargs</code> (Argument), 179
<code>simplify_mdata/fp/resources[Bohrium]/kwargs</code> (Argument), 203	<code>simplify_mdata/train/resources[PBS]/kwargs</code> (Argument), 178
<code>simplify_mdata/fp/resources[DistributedShell]/kwargs</code> (Argument), 202	<code>simplify_mdata/train/resources[SGE]/kwargs</code> (Argument), 179
<code>simplify_mdata/fp/resources[Fugaku]/kwargs</code> (Argument), 204	<code>simplify_mdata/train/resources[Shell]/kwargs</code> (Argument), 177
<code>simplify_mdata/fp/resources[LSF]/kwargs</code> (Argument), 204	<code>simplify_mdata/train/resources[SlurmJobArray]/kwargs</code> (Argument), 178
<code>simplify_mdata/fp/resources[OpenAPI]/kwargs</code> (Argument), 203	<code>simplify_mdata/train/resources[Slurm]/kwargs</code> (Argument), 178
<code>simplify_mdata/fp/resources[PBS]/kwargs</code> (Argument), 202	<code>simplify_mdata/train/resources[Torque]/kwargs</code> (Argument), 179
<code>simplify_mdata/fp/resources[SGE]/kwargs</code> (Argument), 203	
<code>simplify_mdata/fp/resources[Shell]/kwargs</code> (Argument), 202	
<code>simplify_mdata/fp/resources[SlurmJobArray]/kwargs</code> (Argument), 202	<code>simplify_jdata/labeled</code> (Argument), 157
<code>simplify_mdata/fp/resources[Slurm]/kwargs</code> (Argument), 203	<code>Lammps</code> (class in <code>dpgen.auto_test.Lammps</code>), 264
<code>simplify_mdata/fp/resources[Torque]/kwargs</code> (Argument), 204	<code>latt:</code>
<code>simplify_mdata/model_devi/resources[Bohrium]/kwargs</code> (Argument), 191	<code>init_surf_jdata/latt</code> (Argument), 97
	<code>layer_numb:</code>
	<code>init_surf_jdata/layer_numb</code> (Argument), 97
	<code>Li4p()</code> (in module <code>dpgen.auto_test.lib.mfp_eosfit</code>), 254
	<code>link_fp_input()</code> (in module <code>dpgen.data.reaction</code>), 278
	<code>link_pp_files()</code> (in module <code>dpgen.tools.relabel</code>), 302

link_reaxff() (in module *dpgen.data.reaction*), 278
 link_trj() (in module *dpgen.data.reaction*), 278
 lmp:
 run_jdata[model_devi_engine=lammps]/model_devi/machine[SSHContext]/remote_profile/look_for_keys (Argument), 23
 run_jdata[model_devi_engine=lammps]/model_devi/machine[SSHContext]/remote_profile/look_for_keys (Argument), 23
 lmpbox2box() (in module *dpgen.auto_test.lib.lmp*), 253
 load_file() (in module *dpgen.util*), 304
 local_root:
 init_bulk_mdata/fp/machine/local_root (Argument), 82
 init_reaction_mdata/build/machine/local_root (Argument), 127
 init_reaction_mdata/fp/machine/local_root (Argument), 140
 init_reaction_mdata/reaxff/machine/local_root (Argument), 115
 init_surf_mdata/fp/machine/local_root (Argument), 100
 run_mdata/fp/machine/local_root (Argument), 65
 run_mdata/model_devi/machine/local_root (Argument), 52
 run_mdata/train/machine/local_root (Argument), 41
 simplify_mdata/fp/machine/local_root (Argument), 194
 simplify_mdata/model_devi/machine/local_root (Argument), 181
 simplify_mdata/train/machine/local_root (Argument), 169
 LOG4() (in module *dpgen.auto_test.lib.mfp_eosfit*), 253
 LOG5() (in module *dpgen.auto_test.lib.mfp_eosfit*), 254
 log_iter() (in module *dpgen.auto_test.lib.utils*), 258
 log_iter() (in module *dpgen.generator.lib.utils*), 292
 log_task() (in module *dpgen.auto_test.lib.utils*), 258
 log_task() (in module *dpgen.generator.lib.utils*), 292
 look_for_keys:
 init_bulk_mdata/fp/machine[SSHContext]/remote_profile/look_for_keys (Argument), 84
 init_reaction_mdata/build/machine[SSHContext]/remote_profile/look_for_keys (Argument), 130
 init_reaction_mdata/fp/machine[SSHContext]/remote_profile/look_for_keys (Argument), 142
 init_reaction_mdata/reaxff/machine[SSHContext]/remote_profile/look_for_keys (Argument), 117
 init_surf_mdata/fp/machine[SSHContext]/remote_profile/look_for_keys (Argument), 102
 run_mdata/fp/machine[SSHContext]/remote_profile/look_for_keys (Argument), 67
 run_mdata/model_devi/machine[SSHContext]/remote_profile/look_for_keys (Argument), 55
 run_mdata/train/machine[SSHContext]/remote_profile/look_for_keys (Argument), 43
 simplify_mdata/fp/machine[SSHContext]/remote_profile/look_for_keys (Argument), 196
 simplify_mdata/build/machine[SSHContext]/remote_profile/look_for_keys (Argument), 184
 simplify_mdata/fp/machine[SSHContext]/remote_profile/look_for_keys (Argument), 172
 low_level:
 run_jdata[model_devi_engine=amber]/low_level (Argument), 30
 low_level_mdin:
 run_jdata[fp_style=amber/diff]/fp_params/low_level_mdin (Argument), 38
 lsqfit_eos() (in module *dpgen.auto_test.lib.mfp_eosfit*), 255
 Machine:
 init_bulk_mdata/fp/machine (Argument), 81
 init_reaction_mdata/build/machine (Argument), 127
 init_reaction_mdata/fp/machine (Argument), 139
 init_reaction_mdata/reaxff/machine (Argument), 115
 init_surf_mdata/fp/machine (Argument), 99
 run_mdata/fp/machine (Argument), 64
 run_mdata/model_devi/machine (Argument), 52
 run_mdata/train/machine (Argument), 40
 simplify_mdata/fp/machine (Argument), 193
 simplify_mdata/model_devi/machine (Argument), 181
 simplify_mdata/train/machine (Argument), 169
 main() (in module *dpgen.main*), 303
 main_parser() (in module *dpgen.main*), 303
 make_abacus_md() (in module *dpgen.data.gen*), 277
 make_abacus_relax() (in module *dpgen.data.gen*), 277
 make_abacus_scf_input() (in module *dpgen.generator.lib.abacus_scf*), 288
 make_abacus_scf_kpt() (in module *dpgen.generator.lib.abacus_scf*), 288
 make_abacus_scf_stru() (in module *dpgen.generator.lib.abacus_scf*), 288
 make_calculator() (in module *dpgen.auto_test.calculator*), 274
 make_calypso_input() (in module *dpgen.generator.lib.make_calypso*), 290
 make_combines() (in module *dpgen.data.gen*), 277
 make_combines() (in module *dpgen.data.surf*), 278
 make_confs() (*dpgen.auto_test.Elastic.Elastic method*), 262
 make_confs() (*dpgen.auto_test.EOS.EOS method*), 261

`make_confs()` (*dpgen.auto_test.Gamma.Gamma* method), 263
`make_confs()` (*dpgen.auto_test.Interstitial.Interstitial* method), 264
`make_confs()` (*dpgen.auto_test.Property.Property* method), 266
`make_confs()` (*dpgen.auto_test.Surface.Surface* method), 267
`make_confs()` (*dpgen.auto_test.Vacancy.Vacancy* method), 271
`make_cp2k_input()` (in module *dp-gen.generator.lib.cp2k*), 289
`make_cp2k_input_from_external()` (in module *dp-gen.generator.lib.cp2k*), 289
`make_cp2k_xyz()` (in module *dpgen.generator.lib.cp2k*), 289
`make_equi()` (in module *dp-gen.auto_test.common_equi*), 272
`make_fp()` (in module *dpgen.generator.run*), 294
`make_fp()` (in module *dpgen.simplify.simplify*), 299
`make_fp_abacus_scf()` (in module *dp-gen.generator.run*), 294
`make_fp_amber_diff()` (in module *dp-gen.generator.run*), 294
`make_fp_calculation()` (in module *dp-gen.generator.run*), 295
`make_fp_configs()` (in module *dp-gen.simplify.simplify*), 299
`make_fp_cp2k()` (in module *dpgen.generator.run*), 295
`make_fp_custom()` (in module *dpgen.generator.run*), 295
`make_fp_gaussian()` (in module *dpgen.generator.run*), 296
`make_fp_labeled()` (in module *dp-gen.simplify.simplify*), 299
`make_fp_pwmat()` (in module *dpgen.generator.run*), 296
`make_fp_pwscf()` (in module *dpgen.generator.run*), 296
`make_fp_siesta()` (in module *dpgen.generator.run*), 296
`make_fp_task_name()` (in module *dp-gen.generator.run*), 296
`make_fp_vasp()` (in module *dpgen.generator.run*), 296
`make_fp_vasp_cp_cvasp()` (in module *dp-gen.generator.run*), 296
`make_fp_vasp_incar()` (in module *dp-gen.generator.run*), 296
`make_fp_vasp_kp()` (in module *dpgen.generator.run*), 296
`make_gaussian_input()` (in module *dp-gen.generator.lib.gaussian*), 290
`make_input_file()` (*dp-gen.auto_test.ABACUS.ABACUS* method), 260
`make_input_file()` (*dpgen.auto_test.Lammps.Lammps* method), 265
`make_input_file()` (*dpgen.auto_test.Task.Task* method), 269
`make_input_file()` (*dpgen.auto_test.VASP.VASP* method), 270
`make_iter_name()` (in module *dp-gen.auto_test.lib.utils*), 258
`make_iter_name()` (in module *dp-gen.generator.lib.utils*), 292
`make_kspacing_kpoints()` (in module *dp-gen.auto_test.lib.vasp*), 258
`make_kspacing_kpoints_stru()` (in module *dp-gen.generator.lib.abacus_scf*), 288
`make_kspacing_kpt()` (in module *dp-gen.auto_test.lib.abacus*), 251
`make_lammps_elastic()` (in module *dp-gen.auto_test.lib.lammps*), 253
`make_lammps_equi()` (in module *dp-gen.auto_test.lib.lammps*), 253
`make_lammps_eval()` (in module *dp-gen.auto_test.lib.lammps*), 253
`make_lammps_input()` (in module *dp-gen.generator.lib.lammps*), 290
`make_lammps_phonon()` (in module *dp-gen.auto_test.lib.lammps*), 253
`make_lammps_press_relax()` (in module *dp-gen.auto_test.lib.lammps*), 253
`make_lmp()` (in module *dpgen.data.reaction*), 278
`make_model_devi()` (in module *dpgen.generator.run*), 296
`make_model_devi()` (in module *dp-gen.simplify.simplify*), 299
`make_model_devi_conf_name()` (in module *dp-gen.generator.run*), 296
`make_model_devi_task_name()` (in module *dp-gen.generator.run*), 296
`make_path_mp()` (in module *dp-gen.auto_test.gen_confs*), 272
`make_potential_files()` (*dp-gen.auto_test.ABACUS.ABACUS* method), 260
`make_potential_files()` (*dp-gen.auto_test.Lammps.Lammps* method), 265
`make_potential_files()` (*dpgen.auto_test.Task.Task* method), 269
`make_potential_files()` (*dp-gen.auto_test.VASP.VASP* method), 270
`make_property()` (in module *dp-gen.auto_test.common_prop*), 272
`make_property_instance()` (in module *dp-gen.auto_test.common_prop*), 272
`make_pwmat_input()` (in module *dpgen.generator.run*), 296

`make_pwmat_input_dict()` (in module `dp-gen.generator.lib.pwmat`), 290
`make_pwmat_input_user_dict()` (in module `dp-gen.generator.lib.pwmat`), 290
`make_pwscf()` (in module `dpgen.tools.relabel`), 302
`make_pwscf_01_runctrl_dict()` (in module `dp-gen.generator.lib.pwscf`), 291
`make_pwscf_input()` (in module `dp-gen.auto_test.lib.pwscf`), 257
`make_pwscf_input()` (in module `dp-gen.generator.lib.pwscf`), 291
`make_refine()` (in module `dpgen.auto_test.refine`), 272
`make_repro()` (in module `dpgen.auto_test.reproduce`), 273
`make_scale()` (in module `dpgen.data.gen`), 277
`make_scale()` (in module `dpgen.data.surf`), 278
`make_scale_ABACUS()` (in module `dpgen.data.gen`), 277
`make_siesta()` (in module `dpgen.tools.relabel`), 302
`make_siesta_input()` (in module `dp-gen.auto_test.lib.siesta`), 258
`make_siesta_input()` (in module `dp-gen.generator.lib.siesta`), 291
`make_submission()` (in module `dp-gen.dispatcher.Dispatcher`), 287
`make_submission_compat()` (in module `dp-gen.dispatcher.Dispatcher`), 287
`make_super_cell()` (in module `dpgen.data.gen`), 277
`make_super_cell_ABACUS()` (in module `dp-gen.data.gen`), 277
`make_super_cell_poscar()` (in module `dp-gen.data.gen`), 277
`make_super_cell_pymatgen()` (in module `dp-gen.data.surf`), 278
`make_super_cell_STRU()` (in module `dpgen.data.gen`), 277
`make_supercell_abacus()` (in module `dp-gen.generator.lib.abacus_scf`), 288
`make_train()` (in module `dpgen.generator.run`), 296
`make_unit_cell()` (in module `dpgen.data.gen`), 277
`make_unit_cell()` (in module `dpgen.data.surf`), 279
`make_unit_cell_ABACUS()` (in module `dp-gen.data.gen`), 277
`make_vasp()` (in module `dpgen.tools.relabel`), 302
`make_vasp_incar()` (in module `dpgen.generator.run`), 296
`make_vasp_incar()` (in module `dpgen.tools.relabel`), 302
`make_vasp_incar_ele_temp()` (in module `dp-gen.generator.run`), 296
`make_vasp_incar_user_dict()` (in module `dp-gen.generator.lib.vasp`), 292
`make_vasp_kpoints()` (in module `dp-gen.auto_test.lib.vasp`), 259
`make_vasp_kpoints_from_incar()` (in module `dp-gen.auto_test.lib.vasp`), 259
`make_vasp_md()` (in module `dpgen.data.gen`), 277
`make_vasp_phonon_incar()` (in module `dp-gen.auto_test.lib.vasp`), 259
`make_vasp_relax()` (in module `dpgen.data.gen`), 277
`make_vasp_relax()` (in module `dpgen.data.surf`), 279
`make_vasp_relax_incar()` (in module `dp-gen.auto_test.lib.vasp`), 259
`make_vasp_static_incar()` (in module `dp-gen.auto_test.lib.vasp`), 259
`make_work_path()` (in module `dpgen.auto_test.lib.util`), 258
mass_map:
 `run_jdata/mass_map` (Argument), 18
 `simplify_jdata/mass_map` (Argument), 156
mBM4() (in module `dpgen.auto_test.lib.mfp_eosfit`), 255
mBM4poly() (in module `dpgen.auto_test.lib.mfp_eosfit`), 255
mBM5() (in module `dpgen.auto_test.lib.mfp_eosfit`), 255
mBM5poly() (in module `dpgen.auto_test.lib.mfp_eosfit`), 255
md_incar:
 `init_bulk_jdata/md_incar` (Argument), 79
md_kpt:
 `init_bulk_jdata[ABACUS]/md_kpt` (Argument), 80
md_nstep:
 `init_bulk_jdata/md_nstep` (Argument), 80
mdata_arginfo() (in module `dp-gen.dispatcher.Dispatcher`), 288
mdin:
 `run_jdata[model_devi_engine=amber]/mdin` (Argument), 30
mdin_prefix:
 `run_jdata[model_devi_engine=amber]/mdin_prefix` (Argument), 30
mid_point:
 `init_surf_jdata/mid_point` (Argument), 98
mie() (in module `dpgen.auto_test.lib.mfp_eosfit`), 255
mie_simple() (in module `dp-gen.auto_test.lib.mfp_eosfit`), 255
millers:
 `init_surf_jdata/millers` (Argument), 98
mixingWeight:
 `run_jdata[fp_style=siesta]/fp_params/mixingWeight` (Argument), 35
 `simplify_jdata[siesta]/fp_params/mixingWeight` (Argument), 164
model_devi:
 `run_mdata/model_devi` (Argument), 52
 `simplify_mdata/model_devi` (Argument), 181
model_devi_activation_func:
 `run_jdata/model_devi_activation_func` (Argument), 21

simplify_jdata/model_devi_activation_func (Argument), 160
 model_devi_adapt_trust_lo: run_jdata[model_devi_engine=lammps]/model_devi_adapt_trust_loaj: (Argument), 27
 model_devi_amber_args() (in module dp-gen.generator.arginfo), 293
 model_devi_args() (in module dp-gen.generator.arginfo), 293
 model_devi_clean_traj: run_jdata[model_devi_engine=lammps]/model_devi_clean_traj (Argument), 28
 model_devi_dt: run_jdata[model_devi_engine=lammps]/model_devi_dt (Argument), 26
 model_devi_e_trust_hi: simplify_jdata/model_devi_e_trust_hi (Argument), 157
 model_devi_e_trust_lo: simplify_jdata/model_devi_e_trust_lo (Argument), 157
 model_devi_engine: run_jdata/model_devi_engine (Argument), 23
 model_devi_f_avg_relative: run_jdata[model_devi_engine=lammps]/model_devi_f_avg_relative (Argument), 28
 model_devi_f_trust_hi: run_jdata[model_devi_engine=amber]/model_devi_f_trust_hi (Argument), 31
 run_jdata[model_devi_engine=lammps]/model_devi_f_trust_lo (Argument), 26
 run_jdata[model_devi_engine=lammps]/model_devi_f_trust_hi (Argument), 25
 simplify_jdata/model_devi_f_trust_hi (Argument), 157
 model_devi_f_trust_lo: run_jdata[model_devi_engine=amber]/model_devi_f_trust_lo (Argument), 31
 run_jdata[model_devi_engine=lammps]/model_devi_f_trust_lo (Argument), 26
 run_jdata[model_devi_engine=lammps]/model_devi_f_trust_lo (Argument), 25
 simplify_jdata/model_devi_f_trust_lo (Argument), 157
 model_devi_jobs: run_jdata[model_devi_engine=amber]/model_devi_jobs (Argument), 29
 run_jdata[model_devi_engine=lammps]/model_devi_jobs (Argument), 23
 model_devi_jobs_args() (in module dp-gen.generator.arginfo), 293
 model_devi_jobs_rev_mat_args() (in module dp-gen.generator.arginfo), 293
 model_devi_jobs_template_args() (in module dp-gen.generator.arginfo), 293
 model_devi_lmp_args() (in module dp-gen.generator.arginfo), 293
 model_devi_merge_traj: run_jdata[model_devi_engine=lammps]/model_devi_merge_traj (Argument), 28
 model_devi_nopbc: run_jdata[model_devi_engine=lammps]/model_devi_nopbc (Argument), 28
 model_devi_numb_candi_f: run_jdata[model_devi_engine=lammps]/model_devi_numb_candi_f (Argument), 27
 model_devi_numb_candi_v: run_jdata[model_devi_engine=lammps]/model_devi_numb_candi_v (Argument), 27
 model_devi_perc_candi_f: run_jdata[model_devi_engine=lammps]/model_devi_perc_candi_f (Argument), 27
 model_devi_perc_candi_v: run_jdata[model_devi_engine=lammps]/model_devi_perc_candi_v (Argument), 28
 model_devi_plumed: run_jdata[model_devi_engine=lammps]/model_devi_plumed (Argument), 28
 model_devi_plumed_path: run_jdata[model_devi_engine=lammps]/model_devi_plumed_path (Argument), 28
 model_devi_skip: run_jdata[model_devi_engine=lammps]/model_devi_skip (Argument), 26
 model_devi_v_trust_hi: run_jdata[model_devi_engine=lammps]/model_devi_v_trust_hi (Argument), 26
 run_jdata[model_devi_engine=lammps]/model_devi_v_trust_lo (Argument), 26
 modify_stru_path() (in module dp-gen.auto_test.lib.abacus), 251
 module dpgen, 251
 dpgen.arginfo, 302
 dpgen.auto_test, 251
 dpgen.auto_test.ABACUS, 259
 dpgen.auto_test.calculator, 271
 dpgen.auto_test.common_equi, 272
 dpgen.auto_test.common_prop, 272
 dpgen.auto_test.Elastic, 261
 dpgen.auto_test.EOS, 261

dpgen.auto_test.Gamma, 263
 dpgen.auto_test.gen_confs, 272
 dpgen.auto_test.Interstitial, 264
 dpgen.auto_test.Lammps, 264
 dpgen.auto_test.lib, 251
 dpgen.auto_test.lib.abacus, 251
 dpgen.auto_test.lib.crys, 252
 dpgen.auto_test.lib.lammps, 252
 dpgen.auto_test.lib.lmp, 253
 dpgen.auto_test.lib.mfp_eosfit, 253
 dpgen.auto_test.lib.pwscf, 257
 dpgen.auto_test.lib.siesta, 258
 dpgen.auto_test.lib.util, 258
 dpgen.auto_test.lib.utils, 258
 dpgen.auto_test.lib.vasp, 258
 dpgen.auto_test.mpdb, 272
 dpgen.auto_test.Property, 266
 dpgen.auto_test.refine, 272
 dpgen.auto_test.reproduce, 273
 dpgen.auto_test.run, 273
 dpgen.auto_test.Surface, 267
 dpgen.auto_test.Task, 268
 dpgen.auto_test.Vacancy, 271
 dpgen.auto_test.VASP, 269
 dpgen.collect, 273
 dpgen.collect.collect, 273
 dpgen.data, 273
 dpgen.data.arginfo, 276
 dpgen.data.gen, 277
 dpgen.data.reaction, 278
 dpgen.data.surf, 278
 dpgen.data.tools, 273
 dpgen.data.tools.bcc, 273
 dpgen.data.tools.cessp2force_lin, 274
 dpgen.data.tools.create_random_disturb, 274
 dpgen.data.tools.diamond, 274
 dpgen.data.tools.fcc, 275
 dpgen.data.tools.hcp, 275
 dpgen.data.tools.io_lammps, 275
 dpgen.data.tools.sc, 276
 dpgen.database, 279
 dpgen.database.entry, 283
 dpgen.database.run, 284
 dpgen.database.vasp, 284
 dpgen.dispatcher, 287
 dpgen.dispatcher.Dispatcher, 287
 dpgen.generator, 288
 dpgen.generator.arginfo, 292
 dpgen.generator.lib, 288
 dpgen.generator.lib.abacus_scf, 288
 dpgen.generator.lib.cp2k, 289
 dpgen.generator.lib.cvasp, 289
 dpgen.generator.lib.ele_temp, 289
 dpgen.generator.lib.gaussian, 290
 dpgen.generator.lib.lammps, 290
 dpgen.generator.lib.make_calypso, 290
 dpgen.generator.lib.parse_calypso, 290
 dpgen.generator.lib.pwmat, 290
 dpgen.generator.lib.pwscf, 291
 dpgen.generator.lib.run_calypso, 291
 dpgen.generator.lib.siesta, 291
 dpgen.generator.lib.utils, 292
 dpgen.generator.lib.vasp, 292
 dpgen.generator.run, 294
 dpgen.gui, 303
 dpgen.main, 303
 dpgen.remote, 297
 dpgen.remote.decide_machine, 297
 dpgen.simplify, 298
 dpgen.simplify.arginfo, 298
 dpgen.simplify.simplify, 299
 dpgen.tools, 300
 dpgen.tools.auto_gen_param, 300
 dpgen.tools.collect_data, 301
 dpgen.tools.relabel, 302
 dpgen.tools.run_report, 302
 dpgen.tools.stat_iter, 302
 dpgen.tools.stat_sys, 302
 dpgen.tools.stat_time, 302
 dpgen.util, 303
 module_list:
 init_bulk_mdata/fp/resources/module_list
 (Argument), 89
 init_reaction_mdata/build/resources/module_list
 (Argument), 135
 init_reaction_mdata/fp/resources/module_list
 (Argument), 147
 init_reaction_mdata/reaxff/resources/module_list
 (Argument), 122
 init_surf_mdata/fp/resources/module_list
 (Argument), 107
 run_mdata/fp/resources/module_list (Argument), 72
 run_mdata/model_devi/resources/module_list
 (Argument), 60
 run_mdata/train/resources/module_list
 (Argument), 48
 simplify_mdata/fp/resources/module_list
 (Argument), 201
 simplify_mdata/model_devi/resources/module_list
 (Argument), 189
 simplify_mdata/train/resources/module_list
 (Argument), 176
 module_purge:
 init_bulk_mdata/fp/resources/module_purge
 (Argument), 88

init_reaction_mdata/build/resources/module_purge (Argument), 163
 (Argument), 134 murnaghan() (in module dpgen.auto_test.lib.mfp_eosfit),
 init_reaction_mdata/fp/resources/module_purge 255
 (Argument), 147
 init_reaction_mdata/reaxff/resources/module_purge N
 (Argument), 122 NBandsEsti (class in dpgen.generator.lib.ele_temp), 289
 init_surf_mdata/fp/resources/module_purge nbeads:
 (Argument), 107 run_jdata[model_devi_engine=lammps]/model_devi_jobs/nb
 run_mdata/fp/resources/module_purge (Argument), 25
 (Argument), 71 neidelay:
 run_mdata/model_devi/resources/module_purge run_jdata[model_devi_engine=lammps]/model_devi_jobs/ne
 (Argument), 60 (Argument), 25
 run_mdata/train/resources/module_purge normalize() (in module dpgen.util), 304
 (Argument), 47 nproc:
 simplify_mdata/fp/resources/module_purge run_jdata[fp_style=gaussian]/fp_params/nproc
 (Argument), 200 (Argument), 34
 simplify_mdata/model_devi/resources/module_purge simplify_jdata[gaussian]/fp_params/nproc
 (Argument), 188 (Argument), 163
 simplify_mdata/train/resources/module_purge step:
 (Argument), 176 init_reaction_jdata/reaxff/nstep (Argu-
 module_unload_list: ment), 113
 init_bulk_mdata/fp/resources/module_unload_list:
 (Argument), 89 run_jdata[model_devi_engine=amber]/nsteps
 init_reaction_mdata/build/resources/module_unload_list (Argument), 31
 (Argument), 135 run_jdata[model_devi_engine=lammps]/model_devi_jobs/ns
 init_reaction_mdata/fp/resources/module_unload_list (Argument), 25
 (Argument), 147 numb_atoms() (in module dpgen.data.tools.bcc), 273
 init_reaction_mdata/reaxff/resources/module_unload_list numb_atoms() (in module dpgen.data.tools.diamond),
 (Argument), 122 274
 init_surf_mdata/fp/resources/module_unload_list numb_atoms() (in module dpgen.data.tools.fcc), 275
 (Argument), 107 numb_atoms() (in module dpgen.data.tools.hcp), 275
 run_mdata/fp/resources/module_unload_list numb_atoms() (in module dpgen.data.tools.sc), 276
 (Argument), 72 numb_models:
 run_mdata/model_devi/resources/module_unload_list run_jdata/numb_models (Argument), 20
 (Argument), 60 simplify_jdata/numb_models (Argument), 158
 run_mdata/train/resources/module_unload_list number_element (dpgen.database.Entry property), 281
 (Argument), 47 number_element (dpgen.database.entry.Entry prop-
 simplify_mdata/fp/resources/module_unload_list erty), 284
 (Argument), 201 number_node:
 simplify_mdata/model_devi/resources/module_unload_list init_bulk_mdata/fp/resources/number_node
 (Argument), 189 (Argument), 87
 simplify_mdata/train/resources/module_unload_list init_reaction_mdata/build/resources/number_node
 (Argument), 176 (Argument), 133
 morse() (in module dpgen.auto_test.lib.mfp_eosfit), 255 init_reaction_mdata/fp/resources/number_node
 morse_3p() (in module dpgen.auto_test.lib.mfp_eosfit), (Argument), 145
 255 init_reaction_mdata/reaxff/resources/number_node
 morse_6p() (in module dpgen.auto_test.lib.mfp_eosfit), (Argument), 120
 255 init_surf_mdata/fp/resources/number_node
 morse_AB() (in module dpgen.auto_test.lib.mfp_eosfit), (Argument), 105
 255 run_mdata/fp/resources/number_node (Argu-
 multiplicity: ment), 70
 run_jdata[fp_style=gaussian]/fp_params/multiplicity init_mdata/model_devi/resources/number_node
 (Argument), 33 (Argument), 58
 simplify_jdata[gaussian]/fp_params/multiplicity

run_mdata/train/resources/number_node
 (*Argument*), [45](#)
 simplify_mdata/fp/resources/number_node
 (*Argument*), [199](#)
 simplify_mdata/model_devi/resources/number_node
 (*Argument*), [187](#)
 simplify_mdata/train/resources/number_node
 (*Argument*), [174](#)
 NumberPulay:
 run_jdata[fp_style=siesta]/fp_params/NumberPulay
 (*Argument*), [35](#)
 simplify_jdata[siesta]/fp_params/NumberPulay
 (*Argument*), [164](#)
O
 one_h5:
 run_jdata/one_h5 (*Argument*), [21](#)
 simplify_jdata/one_h5 (*Argument*), [160](#)
 out_dir_name() (in module *dpgen.data.gen*), [277](#)
 out_dir_name() (in module *dpgen.data.surf*), [279](#)
 OutcarItemError, [258](#)
 output_fmt:
 run_jdata[fp_style=custom]/fp_params/output_fmt
 (*Argument*), [39](#)
 simplify_jdata[custom]/fp_params/output_fmt
 (*Argument*), [168](#)
 output_fn:
 run_jdata[fp_style=custom]/fp_params/output_fn
 (*Argument*), [40](#)
 simplify_jdata[custom]/fp_params/output_fn
 (*Argument*), [168](#)
P
 para_deg:
 init_bulk_mdata/fp/resources/para_deg
 (*Argument*), [88](#)
 init_reaction_mdata/build/resources/para_deg
 (*Argument*), [134](#)
 init_reaction_mdata/fp/resources/para_deg
 (*Argument*), [147](#)
 init_reaction_mdata/reaxff/resources/para_deg
 (*Argument*), [122](#)
 init_surf_mdata/fp/resources/para_deg
 (*Argument*), [106](#)
 run_mdata/fp/resources/para_deg (*Argument*), [71](#)
 run_mdata/model_devi/resources/para_deg
 (*Argument*), [59](#)
 run_mdata/train/resources/para_deg (*Argument*), [47](#)
 simplify_mdata/fp/resources/para_deg (*Argument*), [200](#)
 simplify_mdata/model_devi/resources/para_deg
 (*Argument*), [188](#)
 simplify_mdata/train/resources/para_deg
 (*Argument*), [176](#)
 parm7:
 run_jdata[model_devi_engine=amber]/parm7
 (*Argument*), [30](#)
 parm7_prefix:
 run_jdata[model_devi_engine=amber]/parm7_prefix
 (*Argument*), [30](#)
 parse_argument() (in module *dp-gen.auto_test.lib.mfp_eosfit*), [255](#)
 parse_cur_job() (in module *dpgen.generator.run*), [296](#)
 parse_cur_job_revmat() (in module *dp-gen.generator.run*), [296](#)
 parse_cur_job_sys_revmat() (in module *dp-gen.generator.run*), [296](#)
 Parser() (in module *dpgen.data.tools.cessp2force_lin*), [274](#)
 parsing_gaussian() (in module *dpgen.database.run*), [284](#)
 parsing_pwscf() (in module *dpgen.database.run*), [284](#)
 parsing_vasp() (in module *dpgen.database.run*), [284](#)
 passphrase:
 init_bulk_mdata/fp/machine[SSHContext]/remote_profile/
 (*Argument*), [83](#)
 init_reaction_mdata/build/machine[SSHContext]/remote_p
 (*Argument*), [129](#)
 init_reaction_mdata/fp/machine[SSHContext]/remote_prof
 (*Argument*), [142](#)
 init_reaction_mdata/reaxff/machine[SSHContext]/remote_
 (*Argument*), [117](#)
 init_surf_mdata/fp/machine[SSHContext]/remote_profile/
 (*Argument*), [102](#)
 run_mdata/fp/machine[SSHContext]/remote_profile/passph
 (*Argument*), [66](#)
 run_mdata/model_devi/machine[SSHContext]/remote_profil
 (*Argument*), [54](#)
 run_mdata/train/machine[SSHContext]/remote_profile/pas
 (*Argument*), [42](#)
 simplify_mdata/fp/machine[SSHContext]/remote_profile/p
 (*Argument*), [195](#)
 simplify_mdata/model_devi/machine[SSHContext]/remote_p
 (*Argument*), [183](#)
 simplify_mdata/train/machine[SSHContext]/remote_profil
 (*Argument*), [171](#)
 password:
 init_bulk_mdata/fp/machine[BohriumContext]/remote_prof
 (*Argument*), [85](#)
 init_bulk_mdata/fp/machine[SSHContext]/remote_profile/
 (*Argument*), [83](#)
 init_reaction_mdata/build/machine[BohriumContext]/remo
 (*Argument*), [131](#)
 init_reaction_mdata/build/machine[SSHContext]/remote_p
 (*Argument*), [129](#)
 init_reaction_mdata/fp/machine[BohriumContext]/remote_

<code>(Argument), 143</code>	<code>run_jdata[model_devi_engine=lammps]/model_devi_jobs/re</code>
<code>init_reaction_mdata/fp/machine[SSHContext]/remote_profile/passwd</code>	<code>run_jdata[model_devi_engine=lammps]/model_devi_jobs/te</code>
<code>(Argument), 141</code>	<code>run_jdata[model_devi_engine=lammps]/model_devi_jobs/te</code>
<code>init_reaction_mdata/reaxff/machine[BohriumContext]/remote_profile/password</code>	<code>run_jdata[model_devi_engine=lammps]/model_devi_jobs/te</code>
<code>(Argument), 118</code>	<code>port:</code>
<code>init_reaction_mdata/reaxff/machine[SSHContext]/remote_profile/passwd</code>	<code>run_jdata[model_devi_engine=lammps]/model_devi_jobs/te</code>
<code>(Argument), 116</code>	<code>(Argument), 83</code>
<code>init_surf_mdata/fp/machine[BohriumContext]/remote_profile/passwd/build/machine[SSHContext]/remote_p</code>	<code>run_jdata[model_devi_engine=lammps]/model_devi_jobs/te</code>
<code>(Argument), 103</code>	<code>(Argument), 129</code>
<code>init_surf_mdata/fp/machine[SSHContext]/remote_profile/passwd</code>	<code>run_jdata[model_devi_engine=lammps]/model_devi_jobs/te</code>
<code>(Argument), 101</code>	<code>(Argument), 141</code>
<code>run_mdata/fp/machine[BohriumContext]/remote_profile/passwd</code>	<code>run_jdata[model_devi_engine=lammps]/model_devi_jobs/te</code>
<code>(Argument), 68</code>	<code>(Argument), 116</code>
<code>run_mdata/fp/machine[SSHContext]/remote_profile/passwd</code>	<code>run_jdata[model_devi_engine=lammps]/model_devi_jobs/te</code>
<code>(Argument), 66</code>	<code>(Argument), 101</code>
<code>run_mdata/model_devi/machine[BohriumContext]/remote_profile/passwd</code>	<code>run_jdata[model_devi_engine=lammps]/model_devi_jobs/te</code>
<code>(Argument), 56</code>	<code>(Argument), 66</code>
<code>run_mdata/model_devi/machine[SSHContext]/remote_profile/passwd</code>	<code>run_jdata[model_devi_engine=lammps]/model_devi_jobs/te</code>
<code>(Argument), 54</code>	<code>(Argument), 54</code>
<code>run_mdata/train/machine[BohriumContext]/remote_profile/passwd</code>	<code>run_jdata[model_devi_engine=lammps]/model_devi_jobs/te</code>
<code>(Argument), 44</code>	<code>(Argument), 42</code>
<code>run_mdata/train/machine[SSHContext]/remote_profile/passwd</code>	<code>run_jdata[model_devi_engine=lammps]/model_devi_jobs/te</code>
<code>(Argument), 42</code>	<code>(Argument), 195</code>
<code>simplify_mdata/fp/machine[BohriumContext]/remote_profile/passwd</code>	<code>run_jdata[model_devi_engine=lammps]/model_devi_jobs/te</code>
<code>(Argument), 197</code>	<code>(Argument), 183</code>
<code>simplify_mdata/fp/machine[SSHContext]/remote_profile/passwd</code>	<code>run_jdata[model_devi_engine=lammps]/model_devi_jobs/te</code>
<code>(Argument), 195</code>	<code>(Argument), 171</code>
<code>simplify_mdata/model_devi/machine[BohriumContext]/remote_profile/passwd_test.lib.abacus),</code>	<code>run_jdata[model_devi_engine=lammps]/model_devi_jobs/te</code>
<code>(Argument), 185</code>	<code>run_jdata[model_devi_engine=lammps]/model_devi_jobs/te</code>
<code>simplify_mdata/model_devi/machine[SSHContext]/remote_profile/passwd</code>	<code>run_jdata[model_devi_engine=lammps]/model_devi_jobs/te</code>
<code>(Argument), 182</code>	<code>run_jdata[model_devi_engine=lammps]/model_devi_jobs/te</code>
<code>simplify_mdata/train/machine[BohriumContext]/remote_profile/passwd</code>	<code>run_jdata[model_devi_engine=lammps]/model_devi_jobs/te</code>
<code>(Argument), 173</code>	<code>run_jdata[model_devi_engine=lammps]/model_devi_jobs/te</code>
<code>simplify_mdata/train/machine[SSHContext]/remote_profile/passwd</code>	<code>run_jdata[model_devi_engine=lammps]/model_devi_jobs/te</code>
<code>(Argument), 170</code>	<code>run_jdata[model_devi_engine=lammps]/model_devi_jobs/te</code>
<code>pert_atom:</code>	<code>run_jdata[model_devi_engine=lammps]/model_devi_jobs/te</code>
<code>init_bulk_jdata/pert_atom (Argument), 80</code>	<code>run_jdata[model_devi_engine=lammps]/model_devi_jobs/te</code>
<code>init_surf_jdata/pert_atom (Argument), 99</code>	<code>run_jdata[model_devi_engine=lammps]/model_devi_jobs/te</code>
<code>pert_box:</code>	<code>run_jdata[model_devi_engine=lammps]/model_devi_jobs/te</code>
<code>init_bulk_jdata/pert_box (Argument), 79</code>	<code>run_jdata[model_devi_engine=lammps]/model_devi_jobs/te</code>
<code>init_surf_jdata/pert_box (Argument), 98</code>	<code>run_jdata[model_devi_engine=lammps]/model_devi_jobs/te</code>
<code>pert_numb:</code>	<code>run_jdata[model_devi_engine=lammps]/model_devi_jobs/te</code>
<code>init_bulk_jdata/pert_numb (Argument), 79</code>	<code>run_jdata[model_devi_engine=lammps]/model_devi_jobs/te</code>
<code>init_surf_jdata/pert_numb (Argument), 98</code>	<code>run_jdata[model_devi_engine=lammps]/model_devi_jobs/te</code>
<code>pert_scaled() (in module dpngen.data.gen), 277</code>	<code>run_jdata[model_devi_engine=lammps]/model_devi_jobs/te</code>
<code>pert_scaled() (in module dpngen.data.surf), 279</code>	<code>run_jdata[model_devi_engine=lammps]/model_devi_jobs/te</code>
<code>perturb_xz() (in module dpngen.auto_test.lib.vasp), 259</code>	<code>run_jdata[model_devi_engine=lammps]/model_devi_jobs/te</code>
<code>pick_data:</code>	<code>run_jdata[model_devi_engine=lammps]/model_devi_jobs/te</code>
<code>simplify_jdata/pick_data (Argument), 157</code>	<code>run_jdata[model_devi_engine=lammps]/model_devi_jobs/te</code>
<code>place_element() (in module dpngen.data.gen), 277</code>	<code>run_jdata[model_devi_engine=lammps]/model_devi_jobs/te</code>
<code>place_element() (in module dpngen.data.surf), 279</code>	<code>run_jdata[model_devi_engine=lammps]/model_devi_jobs/te</code>
<code>place_element_ABACUS() (in module dpngen.data.gen), 277</code>	<code>run_jdata[model_devi_engine=lammps]/model_devi_jobs/te</code>
<code>plm:</code>	<code>run_jdata[model_devi_engine=lammps]/model_devi_jobs/te</code>
	<code>poscar_ele() (in module dpngen.data.surf), 279</code>
	<code>poscar_from_last_dump() (in module dp-</code>
	<code>poscar_natoms() (in module dpngen.auto_test.lib.vasp),</code>
	<code>259</code>
	<code>poscar_scale() (in module dpngen.data.gen), 277</code>
	<code>poscar_scale() (in module dpngen.data.surf), 279</code>
	<code>poscar_scale_abacus() (in module dpngen.data.gen),</code>
	<code>277</code>
	<code>poscar_scale_cartesian() (in module dp-</code>
	<code>gen.data.gen), 278</code>
	<code>poscar_scale_direct() (in module dp-</code>
	<code>gen.data.surf), 279</code>
	<code>poscar_shuffle() (in module dpngen.data.gen), 278</code>

poscar_shuffle() (in module *dpgen.data.surf*), 279
 poscar_shuffle() (in module *dpgen.generator.run*), 296
 poscar_to_conf() (in module *dpgen.generator.run*), 296
 poscar_unit() (in module *dpgen.data.tools.bcc*), 273
 poscar_unit() (in module *dpgen.data.tools.diamond*), 274
 poscar_unit() (in module *dpgen.data.tools.fcc*), 275
 poscar_unit() (in module *dpgen.data.tools.hcp*), 275
 poscar_unit() (in module *dpgen.data.tools.sc*), 276
 poscar_vol() (in module *dpgen.auto_test.lib.vasp*), 259
 post_equi() (in module *dp-gen.auto_test.common_equi*), 272
 post_fp() (in module *dpgen.generator.run*), 296
 post_fp_abacus_scf() (in module *dp-gen.generator.run*), 296
 post_fp_amber_diff() (in module *dp-gen.generator.run*), 296
 post_fp_check_fail() (in module *dp-gen.generator.run*), 296
 post_fp_cp2k() (in module *dpgen.generator.run*), 296
 post_fp_custom() (in module *dpgen.generator.run*), 296
 post_fp_gaussian() (in module *dpgen.generator.run*), 297
 post_fp_pwmat() (in module *dpgen.generator.run*), 297
 post_fp_pwscf() (in module *dpgen.generator.run*), 297
 post_fp_siesta() (in module *dpgen.generator.run*), 297
 post_fp_vasp() (in module *dpgen.generator.run*), 297
 post_model_devi() (in module *dpgen.generator.run*), 297
 post_model_devi() (in module *dp-gen.simplify.simplify*), 299
 post_process() (*dpgen.auto_test.Elastic.Elastic method*), 262
 post_process() (*dpgen.auto_test.EOS.EOS method*), 261
 post_process() (*dpgen.auto_test.Gamma.Gamma method*), 263
 post_process() (*dpgen.auto_test.Interstitial.Interstitial method*), 264
 post_process() (*dpgen.auto_test.Property.Property method*), 267
 post_process() (*dpgen.auto_test.Surface.Surface method*), 267
 post_process() (*dpgen.auto_test.Vacancy.Vacancy method*), 271
 post_property() (in module *dp-gen.auto_test.common_prop*), 272
 post_repro() (in module *dpgen.auto_test.reproduce*), 273
 post_train() (in module *dpgen.generator.run*), 297
 potcars:
 init_bulk_jdata/potcars (*Argument*), 78
 init_surf_jdata/potcars (*Argument*), 96
 predict() (*dpgen.generator.lib.ele_temp.NBandsEsti method*), 290
 prepend_script:
 init_bulk_mdata/fp/resources/prepend_script (*Argument*), 89
 init_reaction_mdata/build/resources/prepend_script (*Argument*), 135
 init_reaction_mdata/fp/resources/prepend_script (*Argument*), 148
 init_reaction_mdata/reaxff/resources/prepend_script (*Argument*), 123
 init_surf_mdata/fp/resources/prepend_script (*Argument*), 107
 run_mdata/fp/resources/prepend_script (*Argument*), 72
 run_mdata/model_devi/resources/prepend_script (*Argument*), 60
 run_mdata/train/resources/prepend_script (*Argument*), 48
 simplify_mdata/fp/resources/prepend_script (*Argument*), 201
 simplify_mdata/model_devi/resources/prepend_script (*Argument*), 189
 simplify_mdata/train/resources/prepend_script (*Argument*), 177
 press:
 run_jdata[model_devi_engine=lammps]/model_devi_jobs/pr (*Argument*), 24
 process_outcar_file_v5_dev() (in module *dp-gen.data.tools.cessp2force_lin*), 274
 program_id:
 init_bulk_mdata/fp/machine[BohriumContext]/remote_prof (*Argument*), 85
 init_reaction_mdata/build/machine[BohriumContext]/remo (*Argument*), 131
 init_reaction_mdata/fp/machine[BohriumContext]/remote_ (*Argument*), 144
 init_reaction_mdata/reaxff/machine[BohriumContext]/rem (*Argument*), 119
 init_surf_mdata/fp/machine[BohriumContext]/remote_prof (*Argument*), 103
 run_mdata/fp/machine[BohriumContext]/remote_profile/pr (*Argument*), 68
 run_mdata/model_devi/machine[BohriumContext]/remote_pr (*Argument*), 56
 run_mdata/train/machine[BohriumContext]/remote_profile (*Argument*), 44
 simplify_mdata/fp/machine[BohriumContext]/remote_profi (*Argument*), 197
 simplify_mdata/model_devi/machine[BohriumContext]/remo (*Argument*), 185

simplify_mdata/train/machine[BohriumContext]/remote_profile/ndata/fp/1/resources/strategy/ratio_unfinished
 (Argument), 173
 Property (class in *dpgen.auto_test.Property*), 266
Q
 qm_charge:
 run_jdata[model_devi_engine=amber]/qm_charge
 (Argument), 31
 qm_region:
 run_jdata[model_devi_engine=amber]/qm_region
 (Argument), 30
 qmkeywords:
 init_reaction_jdata/qmkeywords (Argument),
 114
 queue_name:
 init_bulk_mdata/fp/resources/queue_name
 (Argument), 87
 init_reaction_mdata/build/resources/queue_name
 (Argument), 133
 init_reaction_mdata/fp/resources/queue_name
 (Argument), 145
 init_reaction_mdata/reaxff/resources/queue_name
 (Argument), 121
 init_surf_mdata/fp/resources/queue_name
 (Argument), 105
 run_mdata/fp/resources/queue_name (Argu-
 ment), 70
 run_mdata/model_devi/resources/queue_name
 (Argument), 58
 run_mdata/train/resources/queue_name (Ar-
 gument), 46
 simplify_mdata/fp/resources/queue_name
 (Argument), 199
 simplify_mdata/model_devi/resources/queue_name
 (Argument), 187
 simplify_mdata/train/resources/queue_name
 (Argument), 175
R
 r:
 run_jdata[model_devi_engine=amber]/r (Ar-
 gument), 31
 random_range() (in module *dp-
 gen.data.tools.create_random_disturb*), 274
 RandomDisturbParser() (in module *dp-
 gen.data.tools.create_random_disturb*), 274
 ratio_failed:
 run_jdata/ratio_failed (Argument), 22
 simplify_jdata/ratio_failed (Argument), 161
 ratio_unfinished:
 init_bulk_mdata/fp/resources/strategy/ratio_unfinished
 (Argument), 88
 init_reaction_mdata/build/resources/strategy/ratio_unfinished
 (Argument), 134
 init_reaction_mdata/fp/resources/strategy/ratio_unfinished
 (Argument), 146
 init_reaction_mdata/reaxff/resources/strategy/ratio_unfinished
 (Argument), 121
 init_surf_mdata/fp/resources/strategy/ratio_unfinished
 (Argument), 106
 run_mdata/fp/resources/strategy/ratio_unfinished
 (Argument), 71
 run_mdata/model_devi/resources/strategy/ratio_unfinished
 (Argument), 59
 run_mdata/train/resources/strategy/ratio_unfinished
 (Argument), 47
 simplify_mdata/fp/resources/strategy/ratio_unfinished
 (Argument), 200
 simplify_mdata/model_devi/resources/strategy/ratio_unfinished
 (Argument), 188
 simplify_mdata/train/resources/strategy/ratio_unfinished
 (Argument), 175
 rBM4() (in module *dpgen.auto_test.lib.mfp_eosfit*), 255
 rBM4_pv() (in module *dpgen.auto_test.lib.mfp_eosfit*),
 255
 rBM5() (in module *dpgen.auto_test.lib.mfp_eosfit*), 255
 rBM5_pv() (in module *dpgen.auto_test.lib.mfp_eosfit*),
 256
 read_ve() (in module *dpgen.auto_test.lib.mfp_eosfit*),
 256
 read_velp() (in module *dpgen.auto_test.lib.mfp_eosfit*),
 256
 read_vlp() (in module *dpgen.auto_test.lib.mfp_eosfit*),
 256
 reaxff:
 init_reaction_jdata/reaxff (Argument), 112
 init_reaction_mdata/reaxff (Argument), 114
 reciprocal_box() (in module *dp-
 gen.auto_test.lib.vasp*), 259
 record_iter() (in module *dpgen.auto_test.lib.utils*),
 258
 record_iter() (in module *dpgen.generator.lib.utils*),
 292
 register_iteration() (dp-
 gen.tools.auto_gen_param.Iteration class
 method), 300
 register_sub_iteration() (dp-
 gen.tools.auto_gen_param.Iteration class
 method), 300
 register_sub_system() (dp-
 gen.tools.auto_gen_param.System class
 method), 301
 register_system() (dp-
 gen.tools.auto_gen_param.System class
 method), 301
 regulate_poscar() (in module *dp-
 gen.auto_test.lib.vasp*), 259
 relax_incar:

init_bulk_jdata/relax_incar (*Argument*), 79
 init_surf_jdata/relax_incar (*Argument*), 98
 relax_kpt:
 init_bulk_jdata[ABACUS]/relax_kpt (*Argument*), 80
 remote_profile:
 init_bulk_mdata/fp/machine[BohriumContext]/remote_profile (*Argument*), 85
 init_bulk_mdata/fp/machine[HDFSContext]/remote_profile (*Argument*), 85
 init_bulk_mdata/fp/machine[LazyLocalContext]/remote_profile (*Argument*), 84
 init_bulk_mdata/fp/machine[LocalContext]/remote_profile (*Argument*), 86
 init_bulk_mdata/fp/machine[OpenAPIContext]/remote_profile (*Argument*), 86
 init_bulk_mdata/fp/machine[SSHContext]/remote_profile (*Argument*), 82
 init_reaction_mdata/build/machine[BohriumContext]/remote_profile (*Argument*), 131
 init_reaction_mdata/build/machine[HDFSContext]/remote_profile (*Argument*), 130
 init_reaction_mdata/build/machine[LazyLocalContext]/remote_profile (*Argument*), 130
 init_reaction_mdata/build/machine[LocalContext]/remote_profile (*Argument*), 132
 init_reaction_mdata/build/machine[OpenAPIContext]/remote_profile (*Argument*), 132
 init_reaction_mdata/build/machine[SSHContext]/remote_profile (*Argument*), 128
 init_reaction_mdata/fp/machine[BohriumContext]/remote_profile (*Argument*), 143
 init_reaction_mdata/fp/machine[HDFSContext]/remote_profile (*Argument*), 143
 init_reaction_mdata/fp/machine[LazyLocalContext]/remote_profile (*Argument*), 143
 init_reaction_mdata/fp/machine[LocalContext]/remote_profile (*Argument*), 145
 init_reaction_mdata/fp/machine[OpenAPIContext]/remote_profile (*Argument*), 145
 init_reaction_mdata/fp/machine[SSHContext]/remote_profile (*Argument*), 141
 init_reaction_mdata/reaxff/machine[BohriumContext]/remote_profile (*Argument*), 118
 init_reaction_mdata/reaxff/machine[HDFSContext]/remote_profile (*Argument*), 118
 init_reaction_mdata/reaxff/machine[LazyLocalContext]/remote_profile (*Argument*), 118
 init_reaction_mdata/reaxff/machine[LocalContext]/remote_profile (*Argument*), 120
 init_reaction_mdata/reaxff/machine[OpenAPIContext]/remote_profile (*Argument*), 120
 init_reaction_mdata/reaxff/machine[SSHContext]/remote_profile (*Argument*), 116
 init_surf_mdata/fp/machine[BohriumContext]/remote_profile (*Argument*), 103
 init_surf_mdata/fp/machine[HDFSContext]/remote_profile (*Argument*), 103
 init_surf_mdata/fp/machine[LazyLocalContext]/remote_profile (*Argument*), 102
 init_surf_mdata/fp/machine[LocalContext]/remote_profile (*Argument*), 104
 init_surf_mdata/fp/machine[OpenAPIContext]/remote_profile (*Argument*), 104
 init_surf_mdata/fp/machine[SSHContext]/remote_profile (*Argument*), 101
 init_text_mdata/fp/machine[BohriumContext]/remote_profile (*Argument*), 68
 init_text_mdata/fp/machine[HDFSContext]/remote_profile (*Argument*), 68
 init_text_mdata/fp/machine[LazyLocalContext]/remote_profile (*Argument*), 67
 init_text_mdata/fp/machine[LocalContext]/remote_profile (*Argument*), 69
 init_text_mdata/fp/machine[OpenAPIContext]/remote_profile (*Argument*), 69
 init_text_mdata/fp/machine[SSHContext]/remote_profile (*Argument*), 65
 init_text_mdata/mdevi/machine[BohriumContext]/remote_profile (*Argument*), 56
 init_text_mdata/mdevi/machine[HDFSContext]/remote_profile (*Argument*), 55
 init_text_mdata/mdevi/machine[LazyLocalContext]/remote_profile (*Argument*), 55
 init_text_mdata/mdevi/machine[LocalContext]/remote_profile (*Argument*), 57
 init_text_mdata/mdevi/machine[OpenAPIContext]/remote_profile (*Argument*), 57
 init_text_mdata/mdevi/machine[SSHContext]/remote_profile (*Argument*), 53
 init_text_mdata/mtrain/machine[BohriumContext]/remote_profile (*Argument*), 44
 init_text_mdata/mtrain/machine[HDFSContext]/remote_profile (*Argument*), 43
 init_text_mdata/mtrain/machine[LazyLocalContext]/remote_profile (*Argument*), 43
 init_text_mdata/mtrain/machine[LocalContext]/remote_profile (*Argument*), 45
 init_text_mdata/mtrain/machine[OpenAPIContext]/remote_profile (*Argument*), 45
 init_text_mdata/mtrain/machine[SSHContext]/remote_profile (*Argument*), 41
 init_text_mdata/mtrain/machine[BohriumContext]/remote_profile (*Argument*), 197
 init_text_mdata/mtrain/machine[HDFSContext]/remote_profile (*Argument*), 197
 init_text_mdata/mtrain/machine[LazyLocalContext]/remote_profile (*Argument*), 196

simplify_mdata/fp/machine[LocalContext]/remote_profile (in module dp-gen.auto_test.lib.utils), 258
 (Argument), 198 repeat_to_length() (in module dp-gen.auto_test.lib.utils), 292
 simplify_mdata/fp/machine[OpenAPIContext]/remote_profile (in module dp-gen.data.gen), 278
 (Argument), 198 replace() (in module dp-gen.data.surf), 279
 simplify_mdata/fp/machine[SSHContext]/remote_profile (in module dp-gen.generator.lib.utils), 292
 (Argument), 194 repro_ve() (in module dp-gen.auto_test.lib.mfp_eosfit), 256
 simplify_mdata/model_devi/machine[BohriumContext]/remote_profile repro_vp() (in module dp-gen.auto_test.lib.mfp_eosfit), 256
 (Argument), 185 res_birch() (in module dp-gen.auto_test.lib.mfp_eosfit), 256
 simplify_mdata/model_devi/machine[HDFSContext]/remote_profile res_BM4() (in module dp-gen.auto_test.lib.mfp_eosfit), 256
 (Argument), 184 res_BM5() (in module dp-gen.auto_test.lib.mfp_eosfit), 256
 simplify_mdata/model_devi/machine[LazyLocalContext]/remote_profile res_Li4p() (in module dp-gen.auto_test.lib.mfp_eosfit), 256
 (Argument), 184 res_LOG4() (in module dp-gen.auto_test.lib.mfp_eosfit), 256
 simplify_mdata/model_devi/machine[LocalContext]/remote_profile res_LOG5() (in module dp-gen.auto_test.lib.mfp_eosfit), 256
 (Argument), 186 res_mBM4() (in module dp-gen.auto_test.lib.mfp_eosfit), 256
 simplify_mdata/model_devi/machine[OpenAPIContext]/remote_profile res_mBM4poly() (in module dp-gen.auto_test.lib.mfp_eosfit), 256
 (Argument), 186 res_mBM5() (in module dp-gen.auto_test.lib.mfp_eosfit), 256
 simplify_mdata/model_devi/machine[SSHContext]/remote_profile res_mBM5poly() (in module dp-gen.auto_test.lib.mfp_eosfit), 257
 (Argument), 182 res_mie() (in module dp-gen.auto_test.lib.mfp_eosfit), 257
 simplify_mdata/train/machine[BohriumContext]/remote_profile res_mie_simple() (in module dp-gen.auto_test.lib.mfp_eosfit), 257
 (Argument), 172 res_morse() (in module dp-gen.auto_test.lib.mfp_eosfit), 257
 simplify_mdata/train/machine[HDFSContext]/remote_profile res_morse_3p() (in module dp-gen.auto_test.lib.mfp_eosfit), 257
 (Argument), 172 res_morse_6p() (in module dp-gen.auto_test.lib.mfp_eosfit), 257
 simplify_mdata/train/machine[LazyLocalContext]/remote_profile res_morse_AB() (in module dp-gen.auto_test.lib.mfp_eosfit), 257
 (Argument), 172 res_murnaghan() (in module dp-gen.auto_test.lib.mfp_eosfit), 257
 simplify_mdata/train/machine[LocalContext]/remote_profile res_rBM4() (in module dp-gen.auto_test.lib.mfp_eosfit), 257
 (Argument), 174 res_rBM4_pv() (in module dp-gen.auto_test.lib.mfp_eosfit), 257
 simplify_mdata/train/machine[OpenAPIContext]/remote_profile res_rBM5() (in module dp-gen.auto_test.lib.mfp_eosfit), 257
 (Argument), 174 res_rBM5_pv() (in module dp-gen.auto_test.lib.mfp_eosfit), 257
 simplify_mdata/train/machine[SSHContext]/remote_profile res_rPT4() (in module dp-gen.auto_test.lib.mfp_eosfit), 257
 (Argument), 170
 remote_root: res_mie() (in module dp-gen.auto_test.lib.mfp_eosfit), 257
 init_bulk_mdata/fp/machine/remote_root res_mie_simple() (in module dp-gen.auto_test.lib.mfp_eosfit), 257
 (Argument), 82 res_morse() (in module dp-gen.auto_test.lib.mfp_eosfit), 257
 init_reaction_mdata/build/machine/remote_root res_morse_3p() (in module dp-gen.auto_test.lib.mfp_eosfit), 257
 (Argument), 127 res_morse_6p() (in module dp-gen.auto_test.lib.mfp_eosfit), 257
 init_reaction_mdata/fp/machine/remote_root res_morse_AB() (in module dp-gen.auto_test.lib.mfp_eosfit), 257
 (Argument), 140 res_murnaghan() (in module dp-gen.auto_test.lib.mfp_eosfit), 257
 init_reaction_mdata/reaxff/machine/remote_root res_rBM4() (in module dp-gen.auto_test.lib.mfp_eosfit), 257
 (Argument), 115 res_rBM4_pv() (in module dp-gen.auto_test.lib.mfp_eosfit), 257
 init_surf_mdata/fp/machine/remote_root res_rBM5() (in module dp-gen.auto_test.lib.mfp_eosfit), 257
 (Argument), 100 res_rBM5_pv() (in module dp-gen.auto_test.lib.mfp_eosfit), 257
 run_mdata/fp/machine/remote_root res_rPT4() (in module dp-gen.auto_test.lib.mfp_eosfit), 257
 (Argument), 65
 run_mdata/model_devi/machine/remote_root
 (Argument), 52
 run_mdata/train/machine/remote_root
 (Argument), 41
 simplify_mdata/fp/machine/remote_root
 (Argument), 194
 simplify_mdata/model_devi/machine/remote_root
 (Argument), 181
 simplify_mdata/train/machine/remote_root
 (Argument), 169
 repeat_to_length() (in module dp-

[257](#)
res_rPT4_pv() (in module `dp-gen.auto_test.lib.mfp_eosfit`), [257](#)
res_rPT5() (in module `dp-gen.auto_test.lib.mfp_eosfit`), [257](#)
res_rPT5_pv() (in module `dp-gen.auto_test.lib.mfp_eosfit`), [257](#)
res_SJX_5p() (in module `dp-gen.auto_test.lib.mfp_eosfit`), [256](#)
res_SJX_v2() (in module `dp-gen.auto_test.lib.mfp_eosfit`), [256](#)
res_TEOS() (in module `dp-gen.auto_test.lib.mfp_eosfit`), [256](#)
res_universal() (in module `dp-gen.auto_test.lib.mfp_eosfit`), [257](#)
res_vinet() (in module `dp-gen.auto_test.lib.mfp_eosfit`), [257](#)
res_vinet_pv() (in module `dp-gen.auto_test.lib.mfp_eosfit`), [257](#)
resources:
 init_bulk_mdata/fp/resources (Argument), [86](#)
 init_reaction_mdata/build/resources (Argument), [132](#)
 init_reaction_mdata/fp/resources (Argument), [145](#)
 init_reaction_mdata/reaxff/resources (Argument), [120](#)
 init_surf_mdata/fp/resources (Argument), [105](#)
 run_mdata/fp/resources (Argument), [69](#)
 run_mdata/model_devi/resources (Argument), [58](#)
 run_mdata/train/resources (Argument), [45](#)
 simplify_mdata/fp/resources (Argument), [198](#)
 simplify_mdata/model_devi/resources (Argument), [186](#)
 simplify_mdata/train/resources (Argument), [174](#)
restart_from_iter:
 run_jdata[model_devi_engine=amber]/model_devi_jobs/restart_from_iter (Argument), [29](#)
retry_count:
 init_bulk_mdata/fp/machine[BohriumContext]/remote_profile/build/retry_count (Argument), [85](#)
 init_reaction_mdata/build/machine[BohriumContext]/remote_profile/build/retry_count (Argument), [131](#)
 init_reaction_mdata/fp/machine[BohriumContext]/remote_profile/build/retry_count (Argument), [144](#)
 init_reaction_mdata/reaxff/machine[BohriumContext]/remote_profile/build/retry_count (Argument), [119](#)
 init_surf_mdata/fp/machine[BohriumContext]/remote_profile/build/retry_count (Argument), [103](#)
 run_mdata/fp/machine[BohriumContext]/remote_profile/build/retry_count (Argument), [68](#)
 run_mdata/model_devi/machine[BohriumContext]/remote_profile/build/retry_count (Argument), [56](#)
 run_mdata/train/machine[BohriumContext]/remote_profile/build/retry_count (Argument), [44](#)
 simplify_mdata/fp/machine[BohriumContext]/remote_profile/build/retry_count (Argument), [197](#)
 simplify_mdata/model_devi/machine[BohriumContext]/remote_profile/build/retry_count (Argument), [185](#)
 simplify_mdata/train/machine[BohriumContext]/remote_profile/build/retry_count (Argument), [173](#)
return_direction() (`dp-gen.auto_test.Gamma.Gamma` method), [263](#)
rev_mat:
 run_jdata[model_devi_engine=lammps]/model_devi_jobs/reaxff/rev_mat (Argument), [23](#)
revise_by_keys() (in module `dp-gen.generator.run`), [297](#)
revise_lmp_input_dump() (in module `dp-gen.generator.run`), [297](#)
revise_lmp_input_model() (in module `dp-gen.generator.run`), [297](#)
revise_lmp_input_plm() (in module `dp-gen.generator.run`), [297](#)
rPT4() (in module `dp-gen.auto_test.lib.mfp_eosfit`), [256](#)
rPT4_pv() (in module `dp-gen.auto_test.lib.mfp_eosfit`), [256](#)
rPT5() (in module `dp-gen.auto_test.lib.mfp_eosfit`), [256](#)
rPT5_pv() (in module `dp-gen.auto_test.lib.mfp_eosfit`), [256](#)
run_abacus_md() (in module `dp-gen.data.gen`), [278](#)
run_abacus_relax() (in module `dp-gen.data.gen`), [278](#)
run_build_dataset() (in module `dp-gen.data.reaction`), [278](#)
run_calypso_model_devi() (in module `dp-gen.generator.lib.run_calypso`), [291](#)
run_equi() (in module `dp-gen.auto_test.common_equi`), [272](#)
run_fp() (in module `dp-gen.data.reaction`), [278](#)
run_fp_inner() (in module `dp-gen.generator.run`), [297](#)
run_iter() (in module `dp-gen.generator.run`), [297](#)
run_jdata/build/machine[BohriumContext]/remote_profile/build/retry_count (Argument), [131](#)
run_jdata/default_training_param (Argument), [20](#)
run_jdata/default_training_param.conf (Argument), [20](#)
run_jdata/detailed_report_make_fp (Argument), [20](#)
run_jdata/detailed_report_make_fp.conf (Argument), [20](#)
run_jdata/dp_compress (Argument), [20](#)
run_jdata/dp_train_skip_neighbor_stat (Argument), [20](#)
run_jdata/dp_train_skip_neighbor_stat.conf (Argument), [20](#)

run_jdata/fp_accurate_soft_threshold (Argument)	run_jdata/training_reuse_num_steps (Argument)
fp_accurate_soft_threshold:, 22	training_reuse_num_steps:, 21
run_jdata/fp_accurate_threshold (Argument)	run_jdata/training_reuse_old_ratio (Argument)
fp_accurate_threshold:, 22	training_reuse_old_ratio:, 20
run_jdata/fp_cluster_vacuum (Argument)	run_jdata/training_reuse_start_lr (Argument)
fp_cluster_vacuum:, 22	training_reuse_start_lr:, 21
run_jdata/fp_style (Argument)	run_jdata/training_reuse_start_pref_e (Argument)
fp_style:, 32	training_reuse_start_pref_e:, 21
run_jdata/fp_task_max (Argument)	run_jdata/training_reuse_start_pref_f (Argument)
fp_task_max:, 22	training_reuse_start_pref_f:, 21
run_jdata/fp_task_min (Argument)	run_jdata/type_map (Argument)
fp_task_min:, 22	type_map:, 18
run_jdata/init_batch_size (Argument)	run_jdata/use_ele_temp (Argument)
init_batch_size:, 19	use_ele_temp:, 19
run_jdata/init_data_prefix (Argument)	run_jdata:
init_data_prefix:, 19	run_jdata (Argument), 18
run_jdata/init_data_sys (Argument)	run_jdata_arginfo() (in module dp-gen.generator.arginfo), 293
init_data_sys:, 19	run_jdata[fp_style=abacus]/fp_dpks_descriptor (Argument)
run_jdata/mass_map (Argument)	fp_dpks_descriptor:, 37
mass_map:, 18	run_jdata[fp_style=abacus]/fp_incar (Argument)
run_jdata/model_devi_activation_func (Argument)	fp_incar:, 36
model_devi_activation_func:, 21	run_jdata[fp_style=abacus]/fp_kpt_file (Argument)
run_jdata/model_devi_engine (Argument)	fp_kpt_file:, 37
model_devi_engine:, 23	run_jdata[fp_style=abacus]/fp_orb_files (Argument)
run_jdata/numb_models (Argument)	fp_orb_files:, 36
numb_models:, 20	run_jdata[fp_style=abacus]/fp_pp_files (Argument)
run_jdata/one_h5 (Argument)	fp_pp_files:, 36
one_h5:, 21	run_jdata[fp_style=abacus]/fp_pp_path (Argument)
run_jdata/ratio_failed (Argument)	fp_pp_path:, 36
ratio_failed:, 22	run_jdata[fp_style=abacus]/k_points (Argument)
run_jdata/srtab_file_path (Argument)	k_points:, 37
srtab_file_path:, 21	run_jdata[fp_style=abacus]/user_fp_params (Argument)
run_jdata/sys_batch_size (Argument)	user_fp_params:, 37
sys_batch_size:, 19	run_jdata[fp_style=amber/diff]/fp_params (Argument)
run_jdata/sys_configs (Argument)	fp_params:, 37
sys_configs:, 19	run_jdata[fp_style=amber/diff]/fp_params/high_level_mdin (Argument)
run_jdata/sys_configs_prefix (Argument)	high_level_mdin:, 37
sys_configs_prefix:, 19	run_jdata[fp_style=amber/diff]/fp_params/low_level_mdin (Argument)
run_jdata/sys_format (Argument)	low_level_mdin:, 38
sys_format:, 19	
run_jdata/training_finetune_model (Argument)	
training_finetune_model:, 22	
run_jdata/training_init_frozen_model (Argument)	
training_init_frozen_model:, 21	
run_jdata/training_init_model (Argument)	
training_init_model:, 20	
run_jdata/training_iter0_model_path (Argument)	
training_iter0_model_path:, 20	
run_jdata/training_reuse_iter (Argument)	
training_reuse_iter:, 20	

run_jdata[fp_style=amber/diff]/high_level (Argument) high_level: , 37	run_jdata[fp_style=gaussian]/fp_params/nproc (Argument) nproc: , 34
run_jdata[fp_style=cp2k]/external_input_path (Argument) external_input_path: , 36	run_jdata[fp_style=gaussian]/use_clusters (Argument) use_clusters: , 32
run_jdata[fp_style=cp2k]/user_fp_params (Argument) user_fp_params: , 36	run_jdata[fp_style=pwscf]/fp_params (Argument) fp_params: , 38
run_jdata[fp_style=custom]/fp_params (Argument) fp_params: , 39	run_jdata[fp_style=pwscf]/fp_params/ecut (Argument) ecut: , 38
run_jdata[fp_style=custom]/fp_params/input_fmtrun_jdata[fp_style=custom]/fp_params/input_fmtrun_jdata[fp_style=pwscf]/fp_params/ediff (Argument) input_fmt: , 39	(Argument) ediff: , 38
run_jdata[fp_style=custom]/fp_params/input_fn (Argument) input_fn: , 39	run_jdata[fp_style=pwscf]/fp_params/kspacing (Argument) kspacing: , 39
run_jdata[fp_style=custom]/fp_params/output_fmtrun_jdata[fp_style=custom]/fp_params/output_fmtrun_jdata[fp_style=pwscf]/fp_params/sigma (Argument) output_fmt: , 39	(Argument) sigma: , 39
run_jdata[fp_style=custom]/fp_params/output_fmtrun_jdata[fp_style=custom]/fp_params/output_fmtrun_jdata[fp_style=pwscf]/fp_params/smearing (Argument) output_fn: , 40	(Argument) smearing: , 38
run_jdata[fp_style=gaussian]/cluster_cutoff (Argument) cluster_cutoff: , 33	run_jdata[fp_style=pwscf]/fp_pp_files (Argument) fp_pp_files: , 38
run_jdata[fp_style=gaussian]/cluster_cutoff_hard (Argument) cluster_cutoff_hard: , 33	run_jdata[fp_style=pwscf]/fp_pp_path (Argument) fp_pp_path: , 38
run_jdata[fp_style=gaussian]/cluster_minify (Argument) cluster_minify: , 33	run_jdata[fp_style=pwscf]/user_fp_params (Argument) user_fp_params: , 39
run_jdata[fp_style=gaussian]/fp_params (Argument) fp_params: , 33	run_jdata[fp_style=siesta]/cluster_cutoff (Argument) cluster_cutoff: , 34
run_jdata[fp_style=gaussian]/fp_params/basis_set (Argument) basis_set: , 34	run_jdata[fp_style=siesta]/fp_params (Argument) fp_params: , 34
run_jdata[fp_style=gaussian]/fp_params/charge (Argument) charge: , 34	run_jdata[fp_style=siesta]/fp_params/ecut (Argument) ecut: , 35
run_jdata[fp_style=gaussian]/fp_params/fragment_guesses (Argument) fragment_guesses: , 34	run_jdata[fp_style=siesta]/fp_params/ediff (Argument) ediff: , 35
run_jdata[fp_style=gaussian]/fp_params/keywords (Argument) keywords: , 33	run_jdata[fp_style=siesta]/fp_params/kspacing (Argument) kspacing: , 35
run_jdata[fp_style=gaussian]/fp_params/keywords_high_multiplicity: , 34	run_jdata[fp_style=siesta]/fp_params/mixingWeight (Argument) mixingWeight: , 35
run_jdata[fp_style=gaussian]/fp_params/multiplicity: , 33	run_jdata[fp_style=siesta]/fp_params/NumberPulay (Argument) NumberPulay: , 35

```

run_jdata[fp_style=siesta]/fp_pp_files (Argument)
    fp_pp_files:, 35
run_jdata[fp_style=siesta]/fp_pp_path (Argument)
    fp_pp_path:, 35
run_jdata[fp_style=siesta]/use_clusters (Argument)
    use_clusters:, 34
run_jdata[fp_style=vasp]/cvasp (Argument)
    cvasp:, 32
run_jdata[fp_style=vasp]/fp_aniso_kspacing (Argument)
    fp_aniso_kspacing:, 32
run_jdata[fp_style=vasp]/fp_incar (Argument)
    fp_incar:, 32
run_jdata[fp_style=vasp]/fp_pp_files (Argument)
    fp_pp_files:, 32
run_jdata[fp_style=vasp]/fp_pp_path (Argument)
    fp_pp_path:, 32
run_jdata[fp_style=vasp]/fp_skip_bad_box (Argument)
    fp_skip_bad_box:, 32
run_jdata[model_devi_engine=amber]/cutoff (Argument)
    cutoff:, 30
run_jdata[model_devi_engine=amber]/disang (Argument)
    disang:, 31
run_jdata[model_devi_engine=amber]/disang_prefix (Argument)
    disang_prefix:, 31
run_jdata[model_devi_engine=amber]/low_level (Argument)
    low_level:, 30
run_jdata[model_devi_engine=amber]/mdin (Argument)
    mdin:, 30
run_jdata[model_devi_engine=amber]/mdin_prefix (Argument)
    mdin_prefix:, 30
run_jdata[model_devi_engine=amber]/model_devi_f_trust_hi (Argument)
    model_devi_f_trust_hi:, 31
run_jdata[model_devi_engine=amber]/model_devi_f_trust_lo (Argument)
    model_devi_f_trust_lo:, 31
run_jdata[model_devi_engine=amber]/model_devi_jobs (Argument)
    model_devi_jobs:, 29
run_jdata[model_devi_engine=amber]/model_devi_restart_from_iter (Argument)
    restart_from_iter:, 29
run_jdata[model_devi_engine=amber]/model_devi_jobs/sys_idx (Argument)
    sys_idx:, 29
run_jdata[model_devi_engine=amber]/model_devi_jobs/trj_freq (Argument)
    trj_freq:, 29
run_jdata[model_devi_engine=amber]/nsteps (Argument)
    nsteps:, 31
run_jdata[model_devi_engine=amber]/parm7 (Argument)
    parm7:, 30
run_jdata[model_devi_engine=amber]/parm7_prefix (Argument)
    parm7_prefix:, 30
run_jdata[model_devi_engine=amber]/qm_charge (Argument)
    qm_charge:, 31
run_jdata[model_devi_engine=amber]/qm_region (Argument)
    qm_region:, 30
run_jdata[model_devi_engine=amber]/r (Argument)
    r:, 31
run_jdata[model_devi_engine=lammps]/epsilon (Argument)
    epsilon:, 29
run_jdata[model_devi_engine=lammps]/epsilon_v (Argument)
    epsilon_v:, 29
run_jdata[model_devi_engine=lammps]/model_devi_adapt_trust_lo (Argument)
    model_devi_adapt_trust_lo:, 27
run_jdata[model_devi_engine=lammps]/model_devi_clean_traj (Argument)
    model_devi_clean_traj:, 28
run_jdata[model_devi_engine=lammps]/model_devi_dt (Argument)
    model_devi_dt:, 26
run_jdata[model_devi_engine=lammps]/model_devi_f_avg_relative (Argument)
    model_devi_f_avg_relative:, 28
run_jdata[model_devi_engine=lammps]/model_devi_f_trust_hi (Argument)
    model_devi_f_trust_hi:, 26
run_jdata[model_devi_engine=lammps]/model_devi_f_trust_lo (Argument)
    model_devi_f_trust_lo:, 26
run_jdata[model_devi_engine=lammps]/model_devi_jobs (Argument)
    model_devi_jobs:, 23
run_jdata[model_devi_engine=lammps]/model_devi_jobs/ensemble (Argument)
    ensemble:, 23

```



```

ensemble:, 25
run_jdata[model_devi_engine=lammps]/model_devi_jobs/ensemble/
  (Argument)
model_devi_f_trust_hi:, 25
run_jdata[model_devi_engine=lammps]/model_devi_jobs/ensemble/
  (Argument)
model_devi_f_trust_lo:, 25
run_jdata[model_devi_engine=lammps]/model_devi_jobs/ensemble/
  (Argument)
model_devi_v_trust_hi:, 26
run_jdata[model_devi_engine=lammps]/model_devi_jobs/ensemble/
  (Argument)
model_devi_v_trust_lo:, 26
run_jdata[model_devi_engine=lammps]/model_devi_jobs/ensemble/
  (Argument)
nbeads:, 25
run_jdata[model_devi_engine=lammps]/model_devi_jobs/ensemble/
  (Argument)
neidelay:, 25
run_jdata[model_devi_engine=lammps]/model_devi_jobs/ensemble/
  (Argument)
nsteps:, 25
run_jdata[model_devi_engine=lammps]/model_devi_jobs/ensemble/
  (Argument)
press:, 24
run_jdata[model_devi_engine=lammps]/model_devi_jobs/ensemble/
  (Argument)
rev_mat:, 23
run_jdata[model_devi_engine=lammps]/model_devi_jobs/ensemble/
  (Argument)
lmp:, 23
run_jdata[model_devi_engine=lammps]/model_devi_jobs/ensemble/
  (Argument)
plm:, 24
run_jdata[model_devi_engine=lammps]/model_devi_jobs/ensemble/
  (Argument)
sys_idx:, 24
run_jdata[model_devi_engine=lammps]/model_devi_jobs/ensemble/
  (Argument)
sys_rev_mat:, 24
run_jdata[model_devi_engine=lammps]/model_devi_jobs/ensemble/
  (Argument)
taup:, 25
run_jdata[model_devi_engine=lammps]/model_devi_jobs/ensemble/
  (Argument)
taut:, 25
run_jdata[model_devi_engine=lammps]/model_devi_jobs/ensemble/
  (Argument)
template:, 23
run_jdata[model_devi_engine=lammps]/model_devi_jobs/ensemble/
  (Argument)
lmp:, 23
run_jdata[model_devi_engine=lammps]/model_devi_jobs/ensemble/
  (Argument)

plm:, 23
run_jdata[model_devi_engine=lammps]/model_devi_jobs/ensemble/
  (Argument)
temps:, 24
run_jdata[model_devi_engine=lammps]/model_devi_jobs/ensemble/
  (Argument)
trj_freq:, 24
run_jdata[model_devi_engine=lammps]/model_devi_jobs/ensemble/
  (Argument)
model_devi_merge_traj:, 28
run_jdata[model_devi_engine=lammps]/model_devi_jobs/ensemble/
  (Argument)
model_devi_nopbc:, 28
run_jdata[model_devi_engine=lammps]/model_devi_jobs/ensemble/
  (Argument)
model_devi_nopbc:, 28
run_jdata[model_devi_engine=lammps]/model_devi_jobs/ensemble/
  (Argument)
model_devi_numb_candi_f:, 27
run_jdata[model_devi_engine=lammps]/model_devi_jobs/ensemble/
  (Argument)
model_devi_numb_candi_v:, 27
run_jdata[model_devi_engine=lammps]/model_devi_jobs/ensemble/
  (Argument)
model_devi_perc_candi_f:, 27
run_jdata[model_devi_engine=lammps]/model_devi_jobs/ensemble/
  (Argument)
model_devi_perc_candi_v:, 28
run_jdata[model_devi_engine=lammps]/model_devi_jobs/ensemble/
  (Argument)
model_devi_plumed:, 28
run_jdata[model_devi_engine=lammps]/model_devi_jobs/ensemble/
  (Argument)
model_devi_plumed_path:, 28
run_jdata[model_devi_engine=lammps]/model_devi_jobs/ensemble/
  (Argument)
model_devi_skip:, 26
run_jdata[model_devi_engine=lammps]/model_devi_jobs/ensemble/
  (Argument)
model_devi_v_trust_hi:, 27
run_jdata[model_devi_engine=lammps]/model_devi_jobs/ensemble/
  (Argument)
model_devi_v_trust_lo:, 26
run_jdata[model_devi_engine=lammps]/model_devi_jobs/ensemble/
  (Argument)
shuffle_poscar:, 28
run_jdata[model_devi_engine=lammps]/model_devi_jobs/ensemble/
  (Argument)
use_relative:, 29
run_jdata[model_devi_engine=lammps]/model_devi_jobs/ensemble/
  (Argument)
use_relative_v:, 29
run_jdata[model_devi_engine=lammps]/model_devi_jobs/ensemble/
  (Argument)
run_mdata (Argument)
run_mdata/api_version (Argument)

```

```

    api_version:, 40
run_mdata/deepmd_version (Argument)
    deepmd_version:, 40
run_mdata/fp (Argument)
    fp:, 64
run_mdata/fp/command (Argument)
    command:, 64
run_mdata/fp/machine (Argument)
    machine:, 64
run_mdata/fp/machine/batch_type (Argument)
    batch_type:, 64
run_mdata/fp/machine/clean_asynchronously
    (Argument)
    clean_asynchronously:, 65
run_mdata/fp/machine/context_type (Argument)
    context_type:, 65
run_mdata/fp/machine/local_root (Argument)
    local_root:, 65
run_mdata/fp/machine/remote_root (Argument)
    remote_root:, 65
run_mdata/fp/machine[BohriumContext]/remote_profile
    (Argument)
    remote_profile:, 68
run_mdata/fp/machine[BohriumContext]/remote_profile/remote_email
    (Argument)
    email:, 68
run_mdata/fp/machine[BohriumContext]/remote_profile/remote_ignore_exit_code
    (Argument)
    ignore_exit_code:, 69
run_mdata/fp/machine[BohriumContext]/remote_profile/remote_input_data
    (Argument)
    input_data:, 69
run_mdata/fp/machine[BohriumContext]/remote_profile/remote_keep_backup
    (Argument)
    keep_backup:, 69
run_mdata/fp/machine[BohriumContext]/remote_profile/remote_password
    (Argument)
    password:, 68
run_mdata/fp/machine[BohriumContext]/remote_profile/remote_program_id
    (Argument)
    program_id:, 68
run_mdata/fp/machine[BohriumContext]/remote_profile/remote_retry_count
    (Argument)
    retry_count:, 68
run_mdata/fp/machine[HDFSContext]/remote_profile
    (Argument)
    remote_profile:, 68
run_mdata/fp/machine[LazyLocalContext]/remote_profile
    (Argument)
    remote_profile:, 67
run_mdata/fp/machine[LocalContext]/remote_profile
    (Argument)
    remote_profile:, 69
run_mdata/fp/machine[OpenAPIContext]/remote_profile
    (Argument)
    remote_profile:, 69
run_mdata/fp/machine[SSHContext]/remote_profile
    (Argument)
    remote_profile:, 65
run_mdata/fp/machine[SSHContext]/remote_profile/hostname
    (Argument)
    hostname:, 66
run_mdata/fp/machine[SSHContext]/remote_profile/key_filename
    (Argument)
    key_filename:, 66
run_mdata/fp/machine[SSHContext]/remote_profile/look_for_keys
    (Argument)
    look_for_keys:, 67
run_mdata/fp/machine[SSHContext]/remote_profile/passphrase
    (Argument)
    passphrase:, 66
run_mdata/fp/machine[SSHContext]/remote_profile/password
    (Argument)
    password:, 66
run_mdata/fp/machine[SSHContext]/remote_profile/port
    (Argument)
    port:, 66
run_mdata/fp/machine[SSHContext]/remote_profile/tar_compression
    (Argument)
    tar_compression:, 66
run_mdata/fp/machine[SSHContext]/remote_profile/timeout
    (Argument)
    timeout:, 66
run_mdata/fp/machine[SSHContext]/remote_profile/total_cpu_quota
    (Argument)
    total_cpu_quota:, 67
run_mdata/fp/machine[SSHContext]/remote_profile/total_memory_quota
    (Argument)
    total_memory_quota:, 67
run_mdata/fp/machine[SSHContext]/remote_profile/total_swap_quota
    (Argument)
    total_swap_quota:, 67
run_mdata/fp/machine[SSHContext]/remote_profile/username
    (Argument)
    username:, 66
run_mdata/fp/resources (Argument)
    resources:, 69
run_mdata/fp/resources/append_script (Argument)
    append_script:, 72
run_mdata/fp/resources/batch_type (Argument)
    batch_type:, 72
run_mdata/fp/resources/cpu_per_node (Argument)
    cpu_per_node:, 70
run_mdata/fp/resources/custom_flags (Argument)
    custom_flags:, 70
run_mdata/fp/resources/envs (Argument)
    envs:, 72
run_mdata/fp/resources/gpu_per_node (Argument)
    gpu_per_node:, 70

```

`run_mdata/fp/resources/group_size` (*Argument*)
`group_size`:, 70
`run_mdata/fp/resources/module_list` (*Argument*)
`module_list`:, 72
`run_mdata/fp/resources/module_purge` (*Argument*)
`module_purge`:, 71
`run_mdata/fp/resources/module_unload_list` (*Argument*)
`module_unload_list`:, 72
`run_mdata/fp/resources/number_node` (*Argument*)
`number_node`:, 70
`run_mdata/fp/resources/para_deg` (*Argument*)
`para_deg`:, 71
`run_mdata/fp/resources/prepend_script` (*Argument*)
`prepend_script`:, 72
`run_mdata/fp/resources/queue_name` (*Argument*)
`queue_name`:, 70
`run_mdata/fp/resources/source_list` (*Argument*)
`source_list`:, 71
`run_mdata/fp/resources/strategy` (*Argument*)
`strategy`:, 70
`run_mdata/fp/resources/strategy/customized_script_header_template_file` (*Argument*)
`customized_script_header_template_file`:, 71
`run_mdata/fp/resources/strategy/if_cuda_multi_devices` (*Argument*)
`if_cuda_multi_devices`:, 70
`run_mdata/fp/resources/strategy/ratio_unfinished` (*Argument*)
`ratio_unfinished`:, 71
`run_mdata/fp/resources/wait_time` (*Argument*)
`wait_time`:, 72
`run_mdata/fp/resources[Bohrium]/kwargs` (*Argument*)
`kwargs`:, 74
`run_mdata/fp/resources[DistributedShell]/kwargs` (*Argument*)
`kwargs`:, 73
`run_mdata/fp/resources[Fugaku]/kwargs` (*Argument*)
`kwargs`:, 75
`run_mdata/fp/resources[LSF]/kwargs` (*Argument*)
`kwargs`:, 75
`run_mdata/fp/resources[LSF]/kwargs/custom_gpu_line` (*Argument*)
`custom_gpu_line`:, 75
`run_mdata/fp/resources[LSF]/kwargs/gpu_exclusive` (*Argument*)
`gpu_exclusive`:, 75
`run_mdata/fp/resources[LSF]/kwargs/gpu_new_syntax` (*Argument*)
`gpu_new_syntax`:, 75
`run_mdata/fp/resources[LSF]/kwargs/gpu_usage` (*Argument*)
`gpu_usage`:, 75
`run_mdata/fp/resources[OpenAPI]/kwargs` (*Argument*)
`kwargs`:, 74
`run_mdata/fp/resources[PBS]/kwargs` (*Argument*)
`kwargs`:, 73
`run_mdata/fp/resources[SGE]/kwargs` (*Argument*)
`kwargs`:, 74
`run_mdata/fp/resources[Shell]/kwargs` (*Argument*)
`kwargs`:, 73
`run_mdata/fp/resources[SlurmJobArray]/kwargs` (*Argument*)
`kwargs`:, 73
`run_mdata/fp/resources[SlurmJobArray]/kwargs/custom_gpu_line` (*Argument*)
`custom_gpu_line`:, 73
`run_mdata/fp/resources[SlurmJobArray]/kwargs/slurm_job_size` (*Argument*)
`slurm_job_size`:, 74
`run_mdata/fp/resources[Slurm]/kwargs` (*Argument*)
`kwargs`:, 74
`run_mdata/fp/resources[Slurm]/kwargs/custom_gpu_line`
`custom_gpu_line`:, 74
`run_mdata/fp/resources[Torque]/kwargs` (*Argument*)
`kwargs`:, 75
`run_mdata/fp/user_backward_files` (*Argument*)
`user_backward_files`:, 76
`run_mdata/fp/user_forward_files` (*Argument*)
`user_forward_files`:, 76
`run_mdata/model_devi` (*Argument*)
`model_devi`:, 52
`run_mdata/model_devi/command` (*Argument*)
`command`:, 52
`run_mdata/model_devi/machine` (*Argument*)
`machine`:, 52
`run_mdata/model_devi/machine/batch_type` (*Argument*)
`batch_type`:, 52
`run_mdata/model_devi/machine/clean_asynchronously` (*Argument*)
`clean_asynchronously`:, 53
`run_mdata/model_devi/machine/context_type` (*Argument*)
`context_type`:, 53
`run_mdata/model_devi/machine/local_root`
`local_root`:, 52

```

run_mdata/model_devi/machine/remote_root (Argument)
remote_root:, 52
run_mdata/model_devi/machine[BohriumContext]/remote_profile/
(Argument)
remote_profile:, 56
run_mdata/model_devi/machine[BohriumContext]/remote_profile/email:
(Argument)
email:, 56
run_mdata/model_devi/machine[BohriumContext]/remote_profile/ignore_exit_code:
(Argument)
ignore_exit_code:, 57
run_mdata/model_devi/machine[BohriumContext]/remote_profile/input_data:
(Argument)
input_data:, 57
run_mdata/model_devi/machine[BohriumContext]/remote_profile/keep_backup:
(Argument)
keep_backup:, 57
run_mdata/model_devi/machine[BohriumContext]/remote_profile/password:
(Argument)
password:, 56
run_mdata/model_devi/machine[BohriumContext]/remote_profile/program_id:
(Argument)
program_id:, 56
run_mdata/model_devi/machine[BohriumContext]/remote_profile/retry_count:
(Argument)
retry_count:, 56
run_mdata/model_devi/machine[HDFSContext]/remote_profile:
(Argument)
remote_profile:, 55
run_mdata/model_devi/machine[LazyLocalContext]/remote_profile:
(Argument)
remote_profile:, 55
run_mdata/model_devi/machine[LocalContext]/remote_profile:
(Argument)
remote_profile:, 57
run_mdata/model_devi/machine[OpenAPIContext]/remote_profile:
(Argument)
remote_profile:, 57
run_mdata/model_devi/machine[SSHContext]/remote_profile:
(Argument)
remote_profile:, 53
run_mdata/model_devi/machine[SSHContext]/remote_profile/hostname:
(Argument)
hostname:, 53
run_mdata/model_devi/machine[SSHContext]/remote_profile/key_filename:
(Argument)
key_filename:, 54
run_mdata/model_devi/machine[SSHContext]/remote_profile/look_for_keys:
(Argument)
look_for_keys:, 55
run_mdata/model_devi/machine[SSHContext]/remote_profile/passphrase:
(Argument)
passphrase:, 54

run_mdata/model_devi/machine[SSHContext]/remote_profile/password:
(Argument)
password:, 54
run_mdata/model_devi/machine[SSHContext]/remote_profile/port:
(Argument)
port:, 54
run_mdata/model_devi/machine[SSHContext]/remote_profile/tar_compress:
(Argument)
tar_compress:, 55
run_mdata/model_devi/machine[SSHContext]/remote_profile/timeout:
(Argument)
timeout:, 54
run_mdata/model_devi/machine[SSHContext]/remote_profile/totp_secret:
(Argument)
totp_secret:, 55
run_mdata/model_devi/machine[SSHContext]/remote_profile/username:
(Argument)
username:, 53
run_mdata/model_devi/resources (Argument)
resources:, 58
run_mdata/model_devi/resources/append_script
program_id
append_script:, 60
run_mdata/model_devi/resources/batch_type
batch_type:, 61
run_mdata/model_devi/resources/cpu_per_node
cpu_per_node:, 58
run_mdata/model_devi/resources/custom_flags
custom_flags:, 58
run_mdata/model_devi/resources/envs (Argument)
envs:, 60
run_mdata/model_devi/resources/gpu_per_node
gpu_per_node:, 58
run_mdata/model_devi/resources/group_size
group_size:, 58
run_mdata/model_devi/resources/module_list
module_list:, 60
run_mdata/model_devi/resources/module_purge
module_purge:, 60
run_mdata/model_devi/resources/module_unload_list
module_unload_list:, 60
run_mdata/model_devi/resources/number_node
number_node:, 58
run_mdata/model_devi/resources/para_deg

```

```

        (Argument)
    para_deg:, 59
run_mdata/model_devi/resources/prepend_script
    (Argument)
    prepend_script:, 60
run_mdata/model_devi/resources/queue_name
    (Argument)
    queue_name:, 58
run_mdata/model_devi/resources/source_list
    (Argument)
    source_list:, 59
run_mdata/model_devi/resources/strategy
    (Argument)
    strategy:, 58
run_mdata/model_devi/resources/strategy/customized_script_header_template_file
    (Argument)
    customized_script_header_template_file:,
    59
run_mdata/model_devi/resources/strategy/if_cuda_multi_devices
    (Argument)
    if_cuda_multi_devices:, 59
run_mdata/model_devi/resources/strategy/ratio_unfinished
    (Argument)
    ratio_unfinished:, 59
run_mdata/model_devi/resources/wait_time (Argument)
    wait_time:, 60
run_mdata/model_devi/resources[Bohrium]/kwargs
    (Argument)
    kwargs:, 62
run_mdata/model_devi/resources[DistributedShell]/kwargs
    (Argument)
    kwargs:, 61
run_mdata/model_devi/resources[Fugaku]/kwargs
    (Argument)
    kwargs:, 63
run_mdata/model_devi/resources[LSF]/kwargs
    (Argument)
    kwargs:, 63
run_mdata/model_devi/resources[LSF]/kwargs/custom_gpu_line
    (Argument)
    custom_gpu_line:, 64
run_mdata/model_devi/resources[LSF]/kwargs/gpu_exclusive
    (Argument)
    gpu_exclusive:, 64
run_mdata/model_devi/resources[LSF]/kwargs/gpu_new_syntax
    (Argument)
    gpu_new_syntax:, 63
run_mdata/model_devi/resources[LSF]/kwargs/gpu_usage
    (Argument)
    gpu_usage:, 63
run_mdata/model_devi/resources[OpenAPI]/kwargs
    (Argument)
    kwargs:, 62
run_mdata/model_devi/resources[PBS]/kwargs
    (Argument)
    kwargs:, 61
run_mdata/model_devi/resources[SGE]/kwargs
    (Argument)
    kwargs:, 63
run_mdata/model_devi/resources[Shell]/kwargs
    (Argument)
    kwargs:, 61
run_mdata/model_devi/resources[SlurmJobArray]/kwargs
    (Argument)
    kwargs:, 61
run_mdata/model_devi/resources[SlurmJobArray]/kwargs/custom_gpu_line
    (Argument)
    custom_gpu_line:, 62
run_mdata/model_devi/resources[Slurm]/kwargs
    (Argument)
    kwargs:, 62
run_mdata/model_devi/resources[Slurm]/kwargs/custom_gpu_line
    (Argument)
    custom_gpu_line:, 62
run_mdata/model_devi/resources[Torque]/kwargs
    (Argument)
    kwargs:, 63
run_mdata/model_devi/user_backward_files (Argument)
    user_backward_files:, 64
run_mdata/model_devi/user_forward_files
    (Argument)
    user_forward_files:, 64
run_mdata/train (Argument)
    train:, 40
run_mdata/train/command (Argument)
    command:, 40
run_mdata/train/machine (Argument)
    machine:, 40
run_mdata/train/machine/batch_type (Argument)
    batch_type:, 41
run_mdata/train/machine/clean_asynchronously
    (Argument)
    clean_asynchronously:, 41
run_mdata/train/machine/context_type (Argument)
    context_type:, 41
run_mdata/train/machine/local_root (Argument)
    local_root:, 41
run_mdata/train/machine/remote_root (Argument)
    remote_root:, 41
run_mdata/train/machine[BohriumContext]/remote_profile
    (Argument)

```

```

remote_profile:, 44
run_mdata/train/machine[BohriumContext]/remote_profile/email (Argument)
email:, 44
run_mdata/train/machine[BohriumContext]/remote_profile/tar_compress:, 43
tar_compress:, 43
run_mdata/train/machine[BohriumContext]/remote_profile/timeout:, 42
timeout:, 42
run_mdata/train/machine[BohriumContext]/remote_profile/totp_secret:, 43
totp_secret:, 43
run_mdata/train/machine[BohriumContext]/remote_profile/keep_backup (Argument)
keep_backup:, 45
run_mdata/train/machine[BohriumContext]/remote_profile/passwd (Argument)
password:, 44
run_mdata/train/machine[BohriumContext]/remote_profile/program_id (Argument)
program_id:, 44
run_mdata/train/machine[BohriumContext]/remote_profile/retry_count (Argument)
retry_count:, 44
run_mdata/train/machine[HDFSContext]/remote_profile (Argument)
remote_profile:, 43
run_mdata/train/machine[LazyLocalContext]/remote_profile (Argument)
remote_profile:, 43
run_mdata/train/machine[LocalContext]/remote_profile (Argument)
remote_profile:, 45
run_mdata/train/machine[OpenAPIContext]/remote_profile (Argument)
remote_profile:, 45
run_mdata/train/machine[SSHContext]/remote_profile (Argument)
remote_profile:, 41
run_mdata/train/machine[SSHContext]/remote_profile/hostname:, 42
hostname:, 42
run_mdata/train/machine[SSHContext]/remote_profile/key_filename:, 42
key_filename:, 42
run_mdata/train/machine[SSHContext]/remote_profile/look_for_keys:, 43
look_for_keys:, 43
run_mdata/train/machine[SSHContext]/remote_profile/passphrase:, 42
passphrase:, 42
run_mdata/train/machine[SSHContext]/remote_profile/password:, 42
password:, 42
run_mdata/train/machine[SSHContext]/remote_profile/port:, 42
port:, 42
run_mdata/train/machine[SSHContext]/remote_profile/tar_compress:, 43
tar_compress:, 43
run_mdata/train/machine[SSHContext]/remote_profile/timeout:, 42
timeout:, 42
run_mdata/train/machine[SSHContext]/remote_profile/totp_secret:, 43
totp_secret:, 43
run_mdata/train/machine[SSHContext]/remote_profile/username:, 42
username:, 42
run_mdata/train/resources (Argument)
resources:, 45
run_mdata/train/resources/append_script (Argument)
append_script:, 48
run_mdata/train/resources/batch_type (Argument)
batch_type:, 48
run_mdata/train/resources/cpu_per_node (Argument)
cpu_per_node:, 46
run_mdata/train/resources/custom_flags (Argument)
custom_flags:, 46
run_mdata/train/resources/envs (Argument)
envs:, 48
run_mdata/train/resources/gpu_per_node (Argument)
gpu_per_node:, 46
run_mdata/train/resources/group_size (Argument)
group_size:, 46
run_mdata/train/resources/module_list (Argument)
module_list:, 48
run_mdata/train/resources/module_purge (Argument)
module_purge:, 47
run_mdata/train/resources/module_unload_list (Argument)
module_unload_list:, 47
run_mdata/train/resources/number_node (Argument)
number_node:, 45
run_mdata/train/resources/para_deg (Argument)
para_deg:, 47
run_mdata/train/resources/prepend_script (Argument)
prepend_script:, 48
run_mdata/train/resources/queue_name (Argument)
queue_name (Argument)

```


queue_name:, 46
 run_mdata/train/resources/source_list (Argument)
 source_list:, 47
 run_mdata/train/resources/strategy (Argument)
 strategy:, 46
 run_mdata/train/resources/strategy/customized_script_header_template_file
 (Argument)
 customized_script_header_template_file:, 47
 run_mdata/train/resources/strategy/if_cuda_multi_devices (Argument)
 if_cuda_multi_devices:, 46
 run_mdata/train/resources/strategy/ratio_unfinished (Argument)
 ratio_unfinished:, 47
 run_mdata/train/resources/wait_time (Argument)
 wait_time:, 48
 run_mdata/train/resources[Bohrium]/kwargs (Argument)
 kwargs:, 50
 run_mdata/train/resources[DistributedShell]/kwargs (Argument)
 kwargs:, 49
 run_mdata/train/resources[Fugaku]/kwargs (Argument)
 kwargs:, 51
 run_mdata/train/resources[LSF]/kwargs (Argument)
 kwargs:, 51
 run_mdata/train/resources[LSF]/kwargs/custom_gpu_line
 (Argument)
 custom_gpu_line:, 51
 run_mdata/train/resources[LSF]/kwargs/gpu_exclusive (Argument)
 gpu_exclusive:, 51
 run_mdata/train/resources[LSF]/kwargs/gpu_new_syntax (Argument)
 gpu_new_syntax:, 51
 run_mdata/train/resources[LSF]/kwargs/gpu_usage (Argument)
 gpu_usage:, 51
 run_mdata/train/resources[OpenAPI]/kwargs (Argument)
 kwargs:, 50
 run_mdata/train/resources[PBS]/kwargs (Argument)
 kwargs:, 49
 run_mdata/train/resources[SGE]/kwargs (Argument)
 kwargs:, 50
 run_mdata/train/resources[Shell]/kwargs (Argument)
 kwargs:, 49
 run_mdata/train/resources[SlurmJobArray]/kwargs (Argument)
 kwargs:, 49
 run_mdata/train/resources[SlurmJobArray]/kwargs/custom_gpu_line
 (Argument)
 slurm_job_size:, 49
 run_mdata/train/resources[Slurm]/kwargs (Argument)
 kwargs:, 50
 run_mdata/train/resources[Slurm]/kwargs/custom_gpu_line
 (Argument)
 custom_gpu_line:, 50
 run_mdata/train/resources[Torque]/kwargs (Argument)
 kwargs:, 50
 run_mdata/train/user_backward_files (Argument)
 user_backward_files:, 52
 run_mdata/train/user_forward_files (Argument)
 user_forward_files:, 52
 run_mdata:
 run_mdata (Argument), 40
 run_mdata_arginfo() (in module dp-gen.generator.arginfo), 293
 run_model_devi() (in module dp-gen.generator.run), 297
 run_model_devi() (in module dp-gen.simplify.simplify), 299
 run_property() (in module dp-gen.auto_test.common_prop), 272
 run_reaxff() (in module dp-gen.data.reaction), 278
 run_report() (in module dp-gen.tools.run_report), 302
 run_report() (in module dp-gen.tools.stat_sys), 302
 run_task() (in module dp-gen.auto_test.run), 273
 run_train() (in module dp-gen.generator.run), 297
 run_vasp_md() (in module dp-gen.data.gen), 278
 run_vasp_relax() (in module dp-gen.data.gen), 278
 run_vasp_relax() (in module dp-gen.data.surf), 279
 runvasp() (in module dp-gen.generator.lib.cvasp), 289

S

save() (dp-gen.generator.lib.ele_temp.NBandsEsti method), 290
 sc() (in module dp-gen.auto_test.lib.crys), 252
 scale:
 init_bulk_jdata/scale (Argument), 79
 init_surf_jdata/scale (Argument), 98
 scan_files() (in module dp-gen.tools.auto_gen_param), 301

`scan_outcar_file()` (in module `dp-gen.data.tools.cessp2force_lin`), 274
`sepline()` (in module `dpgen.util`), 304
`set_atoms_typeids()` (in module `dp-gen.data.tools.io_lammps`), 275
`set_atoms_typeids_with_atomic_numbers()` (in module `dpgen.data.tools.io_lammps`), 275
`set_directory()` (in module `dpgen.util`), 304
`set_inter_type_func()` (`dp-gen.auto_test.Lammps.Lammps` method), 266
`set_model_param()` (`dpgen.auto_test.Lammps.Lammps` method), 266
`set_version()` (in module `dpgen.generator.run`), 297
`setup_ele_temp()` (in module `dpgen.util`), 304
`shuffle_poscar:`
 `run_jdata[model_devi_engine=lammps]/shuffle_poscar` (Argument), 28
`shuffle_stru_data()` (in module `dpgen.data.gen`), 278
`sigma:`
 `run_jdata[fp_style=pwscf]/fp_params/sigma` (Argument), 39
 `simplify_jdata[pwscf]/fp_params/sigma` (Argument), 167
`simplify_jdata` (Argument)
 `simplify_jdata:`, 155
`simplify_jdata/default_training_param` (Argument)
 `default_training_param:`, 158
`simplify_jdata/dp_compress` (Argument)
 `dp_compress:`, 159
`simplify_jdata/dp_train_skip_neighbor_stat` (Argument)
 `dp_train_skip_neighbor_stat:`, 158
`simplify_jdata/fp_accurate_soft_threshold` (Argument)
 `fp_accurate_soft_threshold:`, 161
`simplify_jdata/fp_accurate_threshold` (Argument)
 `fp_accurate_threshold:`, 160
`simplify_jdata/fp_style` (Argument)
 `fp_style:`, 161
`simplify_jdata/fp_task_max` (Argument)
 `fp_task_max:`, 160
`simplify_jdata/fp_task_min` (Argument)
 `fp_task_min:`, 160
`simplify_jdata/init_batch_size` (Argument)
 `init_batch_size:`, 156
`simplify_jdata/init_data_prefix` (Argument)
 `init_data_prefix:`, 156
`simplify_jdata/init_data_sys` (Argument)
 `init_data_sys:`, 156
`simplify_jdata/init_pick_number` (Argument)
 `init_pick_number:`, 157
`simplify_jdata/iter_pick_number` (Argument)
 `iter_pick_number:`, 157
`simplify_jdata/labeled` (Argument)
 `labeled:`, 157
`simplify_jdata/mass_map` (Argument)
 `mass_map:`, 156
`simplify_jdata/model_devi_activation_func` (Argument)
 `model_devi_activation_func:`, 160
`simplify_jdata/model_devi_e_trust_hi` (Argument)
 `model_devi_e_trust_hi:`, 157
`simplify_jdata/model_devi_e_trust_lo` (Argument)
 `model_devi_e_trust_lo:`, 157
`simplify_jdata/model_devi_f_trust_hi` (Argument)
 `model_devi_f_trust_hi:`, 157
`simplify_jdata/model_devi_f_trust_lo` (Argument)
 `model_devi_f_trust_lo:`, 157
`simplify_jdata/numb_models` (Argument)
 `numb_models:`, 158
`simplify_jdata/one_h5` (Argument)
 `one_h5:`, 160
`simplify_jdata/pick_data` (Argument)
 `pick_data:`, 157
`simplify_jdata/ratio_failed` (Argument)
 `ratio_failed:`, 161
`simplify_jdata/srtab_file_path` (Argument)
 `srtab_file_path:`, 160
`simplify_jdata/sys_batch_size` (Argument)
 `sys_batch_size:`, 157
`simplify_jdata/sys_configs` (Argument)
 `sys_configs:`, 156
`simplify_jdata/sys_configs_prefix` (Argument)
 `sys_configs_prefix:`, 156
`simplify_jdata/sys_format` (Argument)
 `sys_format:`, 156
`simplify_jdata/training_finetune_model` (Argument)
 `training_finetune_model:`, 160
`simplify_jdata/training_init_frozen_model` (Argument)
 `training_init_frozen_model:`, 160
`simplify_jdata/training_init_model` (Argument)
 `training_init_model:`, 158
`simplify_jdata/training_iter0_model_path` (Argument)
 `training_iter0_model_path:`, 158
`simplify_jdata/training_reuse_iter` (Argument)
 `training_reuse_iter:`, 159
`simplify_jdata/training_reuse_numb_steps` (Argument)

training_reuse_num_steps:, 159
 simplify_jdata/training_reuse_old_ratio
 (Argument)
 training_reuse_old_ratio:, 159
 simplify_jdata/training_reuse_start_lr (Argument)
 training_reuse_start_lr:, 159
 simplify_jdata/training_reuse_start_pref_e
 (Argument)
 training_reuse_start_pref_e:, 159
 simplify_jdata/training_reuse_start_pref_f
 (Argument)
 training_reuse_start_pref_f:, 159
 simplify_jdata/true_error_e_trust_hi (Argument)
 true_error_e_trust_hi:, 158
 simplify_jdata/true_error_e_trust_lo (Argument)
 true_error_e_trust_lo:, 158
 simplify_jdata/true_error_f_trust_hi (Argument)
 true_error_f_trust_hi:, 158
 simplify_jdata/true_error_f_trust_lo (Argument)
 true_error_f_trust_lo:, 158
 simplify_jdata/type_map (Argument)
 type_map:, 155
 simplify_jdata/use_ele_temp (Argument)
 use_ele_temp:, 156
 simplify_jdata:
 simplify_jdata (Argument), 155
 simplify_jdata_arginfo() (in module *dp-gen.simplify_arginfo*), 298
 simplify_jdata[abacus]/fp_dpks_descriptor
 (Argument)
 fp_dpks_descriptor:, 166
 simplify_jdata[abacus]/fp_incar (Argument)
 fp_incar:, 166
 simplify_jdata[abacus]/fp_kpt_file (Argument)
 fp_kpt_file:, 166
 simplify_jdata[abacus]/fp_orb_files (Argument)
 fp_orb_files:, 165
 simplify_jdata[abacus]/fp_pp_files (Argument)
 fp_pp_files:, 165
 simplify_jdata[abacus]/fp_pp_path (Argument)
 fp_pp_path:, 165
 simplify_jdata[abacus]/k_points (Argument)
 k_points:, 166
 simplify_jdata[abacus]/user_fp_params (Argument)
 user_fp_params:, 166
 simplify_jdata[cp2k]/external_input_path (Argument)

external_input_path:, 165
 simplify_jdata[cp2k]/user_fp_params (Argument)
 user_fp_params:, 165
 simplify_jdata[custom]/fp_params (Argument)
 fp_params:, 168
 simplify_jdata[custom]/fp_params/input_fmt
 (Argument)
 input_fmt:, 168
 simplify_jdata[custom]/fp_params/input_fn
 (Argument)
 input_fn:, 168
 simplify_jdata[custom]/fp_params/output_fmt
 (Argument)
 output_fmt:, 168
 simplify_jdata[custom]/fp_params/output_fn
 (Argument)
 output_fn:, 168
 simplify_jdata[gaussian]/cluster_cutoff
 (Argument)
 cluster_cutoff:, 162
 simplify_jdata[gaussian]/cluster_cutoff_hard
 (Argument)
 cluster_cutoff_hard:, 162
 simplify_jdata[gaussian]/cluster_minify
 (Argument)
 cluster_minify:, 162
 simplify_jdata[gaussian]/fp_params (Argument)
 fp_params:, 162
 simplify_jdata[gaussian]/fp_params/basis_set
 (Argument)
 basis_set:, 163
 simplify_jdata[gaussian]/fp_params/charge
 (Argument)
 charge:, 163
 simplify_jdata[gaussian]/fp_params/fragment_guesses
 (Argument)
 fragment_guesses:, 163
 simplify_jdata[gaussian]/fp_params/keywords
 (Argument)
 keywords:, 162
 simplify_jdata[gaussian]/fp_params/keywords_high_multiplicity
 (Argument)
 keywords_high_multiplicity:, 163
 simplify_jdata[gaussian]/fp_params/multiplicity
 (Argument)
 multiplicity:, 163
 simplify_jdata[gaussian]/fp_params/nproc (Argument)
 nproc:, 163
 simplify_jdata[gaussian]/use_clusters (Argument)
 use_clusters:, 162
 simplify_jdata[pwscf]/fp_params (Argument)

```

    fp_params:, 167
simplify_jdata[pwscf]/fp_params/ecut (Argument)
    ecut:, 167
simplify_jdata[pwscf]/fp_params/ediff (Argument)
    ediff:, 167
simplify_jdata[pwscf]/fp_params/kspacing (Argument)
    kspacing:, 167
simplify_jdata[pwscf]/fp_params/sigma (Argument)
    sigma:, 167
simplify_jdata[pwscf]/fp_params/smearing (Argument)
    smearing:, 167
simplify_jdata[pwscf]/fp_pp_files (Argument)
    fp_pp_files:, 166
simplify_jdata[pwscf]/fp_pp_path (Argument)
    fp_pp_path:, 166
simplify_jdata[pwscf]/user_fp_params (Argument)
    user_fp_params:, 167
simplify_jdata[siesta]/cluster_cutoff (Argument)
    cluster_cutoff:, 164
simplify_jdata[siesta]/fp_params (Argument)
    fp_params:, 164
simplify_jdata[siesta]/fp_params/ecut (Argument)
    ecut:, 164
simplify_jdata[siesta]/fp_params/ediff (Argument)
    ediff:, 164
simplify_jdata[siesta]/fp_params/kspacing (Argument)
    kspacing:, 164
simplify_jdata[siesta]/fp_params/mixingWeight (Argument)
    mixingWeight:, 164
simplify_jdata[siesta]/fp_params/NumberPulay (Argument)
    NumberPulay:, 164
simplify_jdata[siesta]/fp_pp_files (Argument)
    fp_pp_files:, 165
simplify_jdata[siesta]/fp_pp_path (Argument)
    fp_pp_path:, 164
simplify_jdata[siesta]/use_clusters (Argument)
    use_clusters:, 163
simplify_jdata[vasp]/cvasp (Argument)
    cvasp:, 161
simplify_jdata[vasp]/fp_aniso_kspacing (Argument)
    fp_aniso_kspacing:, 161
simplify_jdata[vasp]/fp_incar (Argument)
    fp_incar:, 161
simplify_jdata[vasp]/fp_pp_files (Argument)
    fp_pp_files:, 161
simplify_jdata[vasp]/fp_pp_path (Argument)
    fp_pp_path:, 161
simplify_jdata[vasp]/fp_skip_bad_box (Argument)
    fp_skip_bad_box:, 162
simplify_mdata (Argument)
    simplify_mdata:, 168
simplify_mdata/api_version (Argument)
    api_version:, 168
simplify_mdata/deepmd_version (Argument)
    deepmd_version:, 169
simplify_mdata/fp (Argument)
    fp:, 193
simplify_mdata/fp/command (Argument)
    command:, 193
simplify_mdata/fp/machine (Argument)
    machine:, 193
simplify_mdata/fp/machine/batch_type (Argument)
    batch_type:, 194
simplify_mdata/fp/machine/clean_asynchronously (Argument)
    clean_asynchronously:, 194
simplify_mdata/fp/machine/context_type (Argument)
    context_type:, 194
simplify_mdata/fp/machine/local_root (Argument)
    local_root:, 194
simplify_mdata/fp/machine/remote_root (Argument)
    remote_root:, 194
simplify_mdata/fp/machine[BohriumContext]/remote_profile (Argument)
    remote_profile:, 197
simplify_mdata/fp/machine[BohriumContext]/remote_profile/email (Argument)
    email:, 197
simplify_mdata/fp/machine[BohriumContext]/remote_profile/ignore_exit_code (Argument)
    ignore_exit_code:, 198
simplify_mdata/fp/machine[BohriumContext]/remote_profile/input_data (Argument)
    input_data:, 198
simplify_mdata/fp/machine[BohriumContext]/remote_profile/keep_backup (Argument)
    keep_backup:, 198
simplify_mdata/fp/machine[BohriumContext]/remote_profile/p

```

password:, 197	simplify_mdata/fp/resources/append_script
simplify_mdata/fp/machine[BohriumContext]/remote_profile/program_id	append_script:, 201
(Argument)	simplify_mdata/fp/resources/batch_type (Argument)
program_id:, 197	batch_type:, 201
simplify_mdata/fp/machine[BohriumContext]/remote_profile/retry_count	simplify_mdata/fp/resources/cpu_per_node (Argument)
(Argument)	cpu_per_node:, 199
retry_count:, 197	simplify_mdata/fp/resources/custom_flags (Argument)
simplify_mdata/fp/machine[HDFSContext]/remote_profile	custom_flags:, 199
(Argument)	simplify_mdata/fp/resources/envs (Argument)
remote_profile:, 197	simplify_mdata/fp/resources/gpu_per_node (Argument)
simplify_mdata/fp/machine[LazyLocalContext]/remote_profile	gpu_per_node:, 199
(Argument)	simplify_mdata/fp/resources/group_size (Argument)
remote_profile:, 196	group_size:, 199
simplify_mdata/fp/machine[LocalContext]/remote_profile	simplify_mdata/fp/resources/module_list
(Argument)	(Argument)
remote_profile:, 198	simplify_mdata/fp/resources/module_purge (Argument)
simplify_mdata/fp/machine[OpenAPIContext]/remote_profile	module_list:, 201
(Argument)	simplify_mdata/fp/resources/module_unload_list
remote_profile:, 198	(Argument)
simplify_mdata/fp/machine[SSHContext]/remote_profile/hostname	simplify_mdata/fp/resources/number_node
(Argument)	(Argument)
hostname:, 195	simplify_mdata/fp/resources/para_deg (Argument)
simplify_mdata/fp/machine[SSHContext]/remote_profile/key_filename	para_deg:, 200
(Argument)	simplify_mdata/fp/resources/prepend_script
key_filename:, 195	(Argument)
simplify_mdata/fp/machine[SSHContext]/remote_profile/look_for_keys	simplify_mdata/fp/resources/queue_name (Argument)
(Argument)	queue_name:, 199
look_for_keys:, 196	simplify_mdata/fp/resources/source_list
simplify_mdata/fp/machine[SSHContext]/remote_profile/passphrase	(Argument)
(Argument)	simplify_mdata/fp/resources/strategy (Argument)
passphrase:, 195	strategy:, 200
simplify_mdata/fp/machine[SSHContext]/remote_profile/password	simplify_mdata/fp/resources/strategy/customized_script_header_template_file:, 200
(Argument)	simplify_mdata/fp/resources/strategy/if_cuda_multi_devices
password:, 195	(Argument)
simplify_mdata/fp/machine[SSHContext]/remote_profile/port	if_cuda_multi_devices:, 199
(Argument)	
port:, 195	
simplify_mdata/fp/machine[SSHContext]/remote_profile/tar_compress	
(Argument)	
tar_compress:, 196	
simplify_mdata/fp/machine[SSHContext]/remote_profile/timeout	
(Argument)	
timeout:, 196	
simplify_mdata/fp/machine[SSHContext]/remote_profile/totp_secret	
(Argument)	
totp_secret:, 196	
simplify_mdata/fp/machine[SSHContext]/remote_profile/username	
(Argument)	
username:, 195	
simplify_mdata/fp/resources (Argument)	
resources:, 198	

<code>simplify_mdata/fp/resources/strategy/ratio_unfinished</code> (Argument) ratio_unfinished:, 200	<code>simplify_mdata/fp/resources[Slurm]/kwargs/custom_gpu_line</code> (Argument) custom_gpu_line:, 203
<code>simplify_mdata/fp/resources/wait_time</code> (Argument) wait_time:, 201	<code>simplify_mdata/fp/resources[Torque]/kwargs</code> (Argument) kwargs:, 204
<code>simplify_mdata/fp/resources[Bohrium]/kwargs</code> (Argument) kwargs:, 203	<code>simplify_mdata/fp/user_backward_files</code> (Argument) user_backward_files:, 205
<code>simplify_mdata/fp/resources[DistributedShell]/kwargs</code> (Argument) kwargs:, 202	<code>simplify_mdata/fp/user_forward_files</code> (Argument) user_forward_files:, 205
<code>simplify_mdata/fp/resources[Fugaku]/kwargs</code> (Argument) kwargs:, 204	<code>simplify_mdata/model_devi</code> (Argument) model_devi:, 181
<code>simplify_mdata/fp/resources[LSF]/kwargs</code> (Argument) kwargs:, 204	<code>simplify_mdata/model_devi/command</code> (Argument) command:, 181
<code>simplify_mdata/fp/resources[LSF]/kwargs/custom_gpu_line</code> (Argument) custom_gpu_line:, 204	<code>simplify_mdata/model_devi/machine</code> (Argument) machine:, 181
<code>simplify_mdata/fp/resources[LSF]/kwargs/gpu_exclusive</code> (Argument) gpu_exclusive:, 204	<code>simplify_mdata/model_devi/machine/batch_type</code> (Argument) batch_type:, 181
<code>simplify_mdata/fp/resources[LSF]/kwargs/gpu_new_syntax</code> (Argument) gpu_new_syntax:, 204	<code>simplify_mdata/model_devi/machine/clean_asynchronously</code> (Argument) clean_asynchronously:, 181
<code>simplify_mdata/fp/resources[LSF]/kwargs/gpu_usage</code> (Argument) gpu_usage:, 204	<code>simplify_mdata/model_devi/machine/context_type</code> (Argument) context_type:, 182
<code>simplify_mdata/fp/resources[LSF]/kwargs/gpu_usage</code> (Argument) gpu_usage:, 204	<code>simplify_mdata/model_devi/machine/local_root</code> (Argument) local_root:, 181
<code>simplify_mdata/fp/resources[OpenAPI]/kwargs</code> (Argument) kwargs:, 203	<code>simplify_mdata/model_devi/machine/remote_root</code> (Argument) remote_root:, 181
<code>simplify_mdata/fp/resources[PBS]/kwargs</code> (Argument) kwargs:, 202	<code>simplify_mdata/model_devi/machine[BohriumContext]/remote_profile</code> (Argument) remote_profile:, 185
<code>simplify_mdata/fp/resources[SGE]/kwargs</code> (Argument) kwargs:, 203	<code>simplify_mdata/model_devi/machine[BohriumContext]/remote_profile</code> (Argument) email:, 185
<code>simplify_mdata/fp/resources[Shell]/kwargs</code> (Argument) kwargs:, 202	<code>simplify_mdata/model_devi/machine[BohriumContext]/remote_profile</code> (Argument) ignore_exit_code:, 185
<code>simplify_mdata/fp/resources[SlurmJobArray]/kwargs</code> (Argument) kwargs:, 202	<code>simplify_mdata/model_devi/machine[BohriumContext]/remote_profile</code> (Argument) input_data:, 186
<code>simplify_mdata/fp/resources[SlurmJobArray]/kwargs/custom_gpu_line</code> (Argument) custom_gpu_line:, 202	<code>simplify_mdata/model_devi/machine[BohriumContext]/remote_profile</code> (Argument) keep_backup:, 186
<code>simplify_mdata/fp/resources[SlurmJobArray]/kwargs/slurm_job_size</code> (Argument) slurm_job_size:, 203	<code>simplify_mdata/model_devi/machine[BohriumContext]/remote_profile</code> (Argument) password:, 185
<code>simplify_mdata/fp/resources[Slurm]/kwargs</code> (Argument) kwargs:, 203	<code>simplify_mdata/model_devi/machine[BohriumContext]/remote_profile</code> (Argument) program_id:, 185

simplify_mdata/model_devi/machine[BohriumContext]/simplify_mdata/model_devi/resources/batch_type (Argument)	simplify_mdata/model_devi/resources/batch_type (Argument)
retry_count:, 185	batch_type:, 190
simplify_mdata/model_devi/machine[HDFSContext]/simplify_mdata/model_devi/resources/cpu_per_node (Argument)	simplify_mdata/model_devi/resources/cpu_per_node (Argument)
remote_profile:, 184	cpu_per_node:, 187
simplify_mdata/model_devi/machine[LazyLocalContext]/simplify_mdata/model_devi/resources/custom_flags (Argument)	simplify_mdata/model_devi/resources/custom_flags (Argument)
remote_profile:, 184	custom_flags:, 187
simplify_mdata/model_devi/machine[LocalContext]/simplify_mdata/model_devi/resources/envs (Argument)	simplify_mdata/model_devi/resources/envs (Argument)
remote_profile:, 186	envs:, 189
simplify_mdata/model_devi/machine[OpenAPIContext]/simplify_mdata/model_devi/resources/gpu_per_node (Argument)	simplify_mdata/model_devi/resources/gpu_per_node (Argument)
remote_profile:, 186	gpu_per_node:, 187
simplify_mdata/model_devi/machine[SSHContext]/simplify_mdata/model_devi/resources/group_size (Argument)	simplify_mdata/model_devi/resources/group_size (Argument)
remote_profile:, 182	group_size:, 187
simplify_mdata/model_devi/machine[SSHContext]/simplify_mdata/model_devi/resources/module_list (Argument)	simplify_mdata/model_devi/resources/module_list (Argument)
hostname:, 182	module_list:, 189
simplify_mdata/model_devi/machine[SSHContext]/simplify_mdata/model_devi/resources/module_purge (Argument)	simplify_mdata/model_devi/resources/module_purge (Argument)
key_filename:, 183	module_purge:, 188
simplify_mdata/model_devi/machine[SSHContext]/simplify_mdata/model_devi/resources/module_unload_list (Argument)	simplify_mdata/model_devi/resources/module_unload_list (Argument)
look_for_keys:, 184	module_unload_list:, 189
simplify_mdata/model_devi/machine[SSHContext]/simplify_mdata/model_devi/resources/number_node (Argument)	simplify_mdata/model_devi/resources/number_node (Argument)
passphrase:, 183	number_node:, 187
simplify_mdata/model_devi/machine[SSHContext]/simplify_mdata/model_devi/resources/para_deg (Argument)	simplify_mdata/model_devi/resources/para_deg (Argument)
password:, 182	para_deg:, 188
simplify_mdata/model_devi/machine[SSHContext]/simplify_mdata/model_devi/resources/prepend_script (Argument)	simplify_mdata/model_devi/resources/prepend_script (Argument)
port:, 183	prepend_script:, 189
simplify_mdata/model_devi/machine[SSHContext]/simplify_mdata/model_devi/resources/queue_name (Argument)	simplify_mdata/model_devi/resources/queue_name (Argument)
tar_compress:, 184	queue_name:, 187
simplify_mdata/model_devi/machine[SSHContext]/simplify_mdata/model_devi/resources/source_list (Argument)	simplify_mdata/model_devi/resources/source_list (Argument)
timeout:, 183	source_list:, 188
simplify_mdata/model_devi/machine[SSHContext]/simplify_mdata/model_devi/resources/strategy (Argument)	simplify_mdata/model_devi/resources/strategy (Argument)
totp_secret:, 183	strategy:, 187
simplify_mdata/model_devi/machine[SSHContext]/simplify_mdata/model_devi/resources/strategy/customized_script_header_template_file: (Argument)	simplify_mdata/model_devi/resources/strategy/customized_script_header_template_file: (Argument)
username:, 182	188
simplify_mdata/model_devi/resources (Argument)	simplify_mdata/model_devi/resources/strategy/if_cuda_multi_devices: (Argument)
resources:, 186	188
simplify_mdata/model_devi/resources/append_script (Argument)	simplify_mdata/model_devi/resources/strategy/ratio_unfinished: (Argument)
append_script:, 189	

ratio_unfinished:, 188	custom_gpu_line:, 191
simplify_mdata/model_devi/resources/wait_time (Argument)	simplify_mdata/model_devi/resources[Torque]/kwargs (Argument)
wait_time:, 189	kwargs:, 192
simplify_mdata/model_devi/resources[Bohrium]/kwargs (Argument)	simplify_mdata/model_devi/user_backward_files (Argument)
kwargs:, 191	user_backward_files:, 193
simplify_mdata/model_devi/resources[DistributedShell]/kwargs (Argument)	simplify_mdata/model_devi/user_forward_files (Argument)
kwargs:, 190	user_forward_files:, 193
simplify_mdata/model_devi/resources[Fugaku]/kwargs (Argument)	simplify_mdata/train (Argument)
kwargs:, 192	train:, 169
simplify_mdata/model_devi/resources[LSF]/kwargs (Argument)	simplify_mdata/train/command (Argument)
kwargs:, 192	command:, 169
simplify_mdata/model_devi/resources[LSF]/kwargs (Argument)	simplify_mdata/train/machine (Argument)
custom_gpu_line:, 193	machine:, 169
simplify_mdata/model_devi/resources[LSF]/kwargs (Argument)	simplify_mdata/train/machine/batch_type (Argument)
gpu_exclusive:, 193	batch_type:, 169
simplify_mdata/model_devi/resources[LSF]/kwargs (Argument)	simplify_mdata/train/machine/clean_asynchronously (Argument)
gpu_new_syntax:, 192	clean_asynchronously:, 169
simplify_mdata/model_devi/resources[LSF]/kwargs (Argument)	simplify_mdata/train/machine/context_type (Argument)
gpu_usage:, 192	context_type:, 170
simplify_mdata/model_devi/resources[LSF]/kwargs (Argument)	simplify_mdata/train/machine/local_root (Argument)
simplify_mdata/model_devi/resources[OpenAPI]/kwargs (Argument)	local_root:, 169
kwargs:, 191	simplify_mdata/train/machine/remote_root (Argument)
simplify_mdata/model_devi/resources[PBS]/kwargs (Argument)	remote_root:, 169
kwargs:, 190	simplify_mdata/train/machine[BohriumContext]/remote_profile (Argument)
simplify_mdata/model_devi/resources[SGE]/kwargs (Argument)	remote_profile:, 172
kwargs:, 192	simplify_mdata/train/machine[BohriumContext]/remote_profile (Argument)
simplify_mdata/model_devi/resources[Shell]/kwargs (Argument)	email:, 172
kwargs:, 190	simplify_mdata/train/machine[BohriumContext]/remote_profile (Argument)
simplify_mdata/model_devi/resources[SlurmJobArns]/kwargs (Argument)	ignore_exit_code:, 173
kwargs:, 190	simplify_mdata/train/machine[BohriumContext]/remote_profile (Argument)
simplify_mdata/model_devi/resources[SlurmJobArns]/kwargs (Argument)	input_data:, 174
custom_gpu_line:, 191	simplify_mdata/train/machine[BohriumContext]/remote_profile (Argument)
simplify_mdata/model_devi/resources[SlurmJobArns]/kwargs (Argument)	keep_backup:, 173
slurm_job_size:, 191	simplify_mdata/train/machine[BohriumContext]/remote_profile (Argument)
simplify_mdata/model_devi/resources[Slurm]/kwargs (Argument)	password:, 173
kwargs:, 191	simplify_mdata/train/machine[BohriumContext]/remote_profile (Argument)
simplify_mdata/model_devi/resources[Slurm]/kwargs (Argument)	program_id:, 173
	simplify_mdata/train/machine[BohriumContext]/remote_profile (Argument)

retry_count:, 173	simplify_mdata/train/resources/cpu_per_node
simplify_mdata/train/machine[HDFSContext]/remote_profile(<i>Argument</i>)	cpu_per_node:, 174
remote_profile:, 172	simplify_mdata/train/resources/custom_flags
simplify_mdata/train/machine[LazyLocalContext]/remote_profile(<i>Argument</i>)	custom_flags:, 175
remote_profile:, 172	simplify_mdata/train/resources/envs (<i>Argument</i>)
simplify_mdata/train/machine[LocalContext]/remote_profile(<i>Argument</i>)	envs:, 177
remote_profile:, 174	simplify_mdata/train/resources/gpu_per_node
simplify_mdata/train/machine[OpenAPIContext]/remote_profile(<i>Argument</i>)	gpu_per_node:, 175
remote_profile:, 174	simplify_mdata/train/resources/group_size
simplify_mdata/train/machine[SSHContext]/remote_profile(<i>Argument</i>)	group_size:, 175
remote_profile:, 170	simplify_mdata/train/resources/module_list
simplify_mdata/train/machine[SSHContext]/remote_profile(<i>Argument</i>)	module_list:, 176
hostname:, 170	simplify_mdata/train/resources/module_purge
simplify_mdata/train/machine[SSHContext]/remote_profile(<i>Argument</i>)	module_purge:, 176
key_filename:, 171	simplify_mdata/train/resources/module_unload_list
simplify_mdata/train/machine[SSHContext]/remote_profile(<i>Argument</i>)	module_unload_list:, 176
look_for_keys:, 172	simplify_mdata/train/resources/number_node
simplify_mdata/train/machine[SSHContext]/remote_profile(<i>Argument</i>)	number_node:, 174
passphrase:, 171	simplify_mdata/train/resources/para_deg
simplify_mdata/train/machine[SSHContext]/remote_profile(<i>Argument</i>)	para_deg:, 176
password:, 170	simplify_mdata/train/resources/prepend_script
simplify_mdata/train/machine[SSHContext]/remote_profile(<i>Argument</i>)	prepend_script:, 177
port:, 171	simplify_mdata/train/resources/queue_name
simplify_mdata/train/machine[SSHContext]/remote_profile(<i>Argument</i>)	queue_name:, 175
tar_compress:, 171	simplify_mdata/train/resources/source_list
simplify_mdata/train/machine[SSHContext]/remote_profile(<i>Argument</i>)	source_list:, 176
timeout:, 171	simplify_mdata/train/resources/strategy
simplify_mdata/train/machine[SSHContext]/remote_profile(<i>Argument</i>)	strategy:, 175
totp_secret:, 171	simplify_mdata/train/resources/strategy/customized_script
simplify_mdata/train/machine[SSHContext]/remote_profile(<i>Argument</i>)	customized_script_header_template_file:, 176
username:, 170	simplify_mdata/train/resources/strategy/if_cuda_multi_devices
simplify_mdata/train/resources (<i>Argument</i>)	if_cuda_multi_devices:, 175
resources:, 174	simplify_mdata/train/resources/strategy/ratio_unfinished
simplify_mdata/train/resources/append_script (<i>Argument</i>)	ratio_unfinished:, 175
append_script:, 177	simplify_mdata/train/resources/wait_time (<i>Argument</i>)
simplify_mdata/train/resources/batch_type (<i>Argument</i>)	
batch_type:, 177	

```

    wait_time:, 177
simplify_mdata/train/resources[Bohrium]/kwargs:
    (Argument)
    kwargs:, 179
simplify_mdata/train/resources[DistributedShell]/kwargs:
    (Argument)
    kwargs:, 178
simplify_mdata/train/resources[Fugaku]/kwargs:
    (Argument)
    kwargs:, 179
simplify_mdata/train/resources[LSF]/kwargs:
    (Argument)
    kwargs:, 180
simplify_mdata/train/resources[LSF]/kwargs/custom_gpu_line:
    (Argument)
    custom_gpu_line:, 180
simplify_mdata/train/resources[LSF]/kwargs/gpu_exclusive:
    (Argument)
    gpu_exclusive:, 180
simplify_mdata/train/resources[LSF]/kwargs/gpu_new_syntax:
    (Argument)
    gpu_new_syntax:, 180
simplify_mdata/train/resources[LSF]/kwargs/gpu_usage:
    (Argument)
    gpu_usage:, 180
simplify_mdata/train/resources[OpenAPI]/kwargs:
    (Argument)
    kwargs:, 179
simplify_mdata/train/resources[PBS]/kwargs:
    (Argument)
    kwargs:, 178
simplify_mdata/train/resources[SGE]/kwargs:
    (Argument)
    kwargs:, 179
simplify_mdata/train/resources[Shell]/kwargs:
    (Argument)
    kwargs:, 177
simplify_mdata/train/resources[SlurmJobArray]/kwargs:
    (Argument)
    kwargs:, 178
simplify_mdata/train/resources[SlurmJobArray]/kwargs/custom_gpu_line:
    (Argument)
    custom_gpu_line:, 178
simplify_mdata/train/resources[SlurmJobArray]/kwargs/slurm_job_size:
    (Argument)
    slurm_job_size:, 178
simplify_mdata/train/resources[Slurm]/kwargs:
    (Argument)
    kwargs:, 178
simplify_mdata/train/resources[Slurm]/kwargs/custom_gpu_line:
    (Argument)
    custom_gpu_line:, 179
simplify_mdata/train/resources[Torque]/kwargs:
    (Argument)
    kwargs:, 179
simplify_mdata/train/user_backward_files (Argument)
    user_backward_files:, 181
simplify_mdata/train/user_forward_files (Argument)
    user_forward_files:, 180
simplify_mdata:
    simplify_mdata (Argument), 168
simplify_mdata_arginfo() (in module dp-gen.simplify_arginfo), 298
SJX_5p() (in module dpgen.auto_test.lib.mfp_eosfit), 254
skip_relax:
    skip_relax (Argument), 79
    init_surf_jdata/skip_relax (Argument), 98
slurm_job_size:
    simplify_mdata/fp/resources[SlurmJobArray]/kwargs/slurm_job_size (Argument), 91
    init_reaction_mdata/build/resources[SlurmJobArray]/kwargs/slurm_job_size (Argument), 137
    init_reaction_mdata/fp/resources[SlurmJobArray]/kwargs/slurm_job_size (Argument), 149
    init_reaction_mdata/reaxff/resources[SlurmJobArray]/kwargs/slurm_job_size (Argument), 124
    init_surf_mdata/fp/resources[SlurmJobArray]/kwargs/slurm_job_size (Argument), 109
    run_mdata/fp/resources[SlurmJobArray]/kwargs/slurm_job_size (Argument), 74
    run_mdata/model_devi/resources[SlurmJobArray]/kwargs/slurm_job_size (Argument), 62
    run_mdata/train/resources[SlurmJobArray]/kwargs/slurm_job_size (Argument), 49
    simplify_mdata/fp/resources[SlurmJobArray]/kwargs/slurm_job_size (Argument), 203
    simplify_mdata/model_devi/resources[SlurmJobArray]/kwargs/slurm_job_size (Argument), 191
    simplify_mdata/train/resources[SlurmJobArray]/kwargs/slurm_job_size (Argument), 178
    smearing:
        run_jdata[fp_style=pwscf]/fp_params/smearing (Argument), 38
        simplify_jdata[pwscf]/fp_params/smearing (Argument), 167
    sort_poscar() (in module dpgen.auto_test.lib.vasp), 259
    source_list:
        simplify_mdata/fp/resources/source_list (Argument), 88
        init_reaction_mdata/build/resources/source_list (Argument), 134
        init_reaction_mdata/fp/resources/source_list (Argument), 134

```


(Argument), 147
 init_reaction_mdata/reaxff/resources/source_list (Argument), 122
 init_surf_mdata/fp/resources/source_list (Argument), 106
 run_mdata/fp/resources/source_list (Argument), 71
 run_mdata/model_devi/resources/source_list (Argument), 59
 run_mdata/train/resources/source_list (Argument), 47
 simplify_mdata/fp/resources/source_list (Argument), 200
 simplify_mdata/model_devi/resources/source_list (Argument), 188
 simplify_mdata/train/resources/source_list (Argument), 176
 srtab_file_path:
 run_jdata/srtab_file_path (Argument), 21
 simplify_jdata/srtab_file_path (Argument), 160
 stages:
 init_bulk_jdata/stages (Argument), 78
 init_surf_jdata/stages (Argument), 96
 start_dpGUI() (in module dpGen.gui), 303
 stat_iter() (in module dpGen.tools.stat_iter), 302
 stat_sys() (in module dpGen.tools.stat_sys), 302
 stat_time() (in module dpGen.tools.stat_time), 302
 strategy:
 init_bulk_mdata/fp/resources/strategy (Argument), 87
 init_reaction_mdata/build/resources/strategy (Argument), 133
 init_reaction_mdata/fp/resources/strategy (Argument), 146
 init_reaction_mdata/reaxff/resources/strategy (Argument), 121
 init_surf_mdata/fp/resources/strategy (Argument), 105
 run_mdata/fp/resources/strategy (Argument), 70
 run_mdata/model_devi/resources/strategy (Argument), 58
 run_mdata/train/resources/strategy (Argument), 46
 simplify_mdata/fp/resources/strategy (Argument), 199
 simplify_mdata/model_devi/resources/strategy (Argument), 187
 simplify_mdata/train/resources/strategy (Argument), 175
 stress6_to_stress9() (in module dpGen.data.tools.io_lammps), 275
 stress9_to_stress6() (in module dpGen.data.tools.io_lammps), 275
 stru_fix_atom() (in module dpGen.auto_test.lib.abacus), 251
 stru_scale() (in module dpGen.auto_test.lib.abacus), 252
 super_cell:
 init_bulk_jdata/super_cell (Argument), 78
 init_surf_jdata/super_cell (Argument), 96
 Surface (class in dpGen.auto_test.Surface), 267
 symlink_user_forward_files() (in module dpGen.generator.lib.utils), 292
 sys_batch_size:
 run_jdata/sys_batch_size (Argument), 19
 simplify_jdata/sys_batch_size (Argument), 157
 sys_configs:
 run_jdata/sys_configs (Argument), 19
 simplify_jdata/sys_configs (Argument), 156
 sys_configs_prefix:
 run_jdata/sys_configs_prefix (Argument), 19
 simplify_jdata/sys_configs_prefix (Argument), 156
 sys_format:
 run_jdata/sys_format (Argument), 19
 simplify_jdata/sys_format (Argument), 156
 sys_idx:
 run_jdata[model_devi_engine=amber]/model_devi_jobs/sys_idx (Argument), 29
 run_jdata[model_devi_engine=lammps]/model_devi_jobs/sys_idx (Argument), 24
 sys_link_fp_vasp_pp() (in module dpGen.generator.run), 297
 sys_rev_mat:
 run_jdata[model_devi_engine=lammps]/model_devi_jobs/sys_rev_mat (Argument), 24
 System (class in dpGen.tools.auto_gen_param), 300
 system_data() (in module dpGen.auto_test.lib.lmp), 253
T
 take_cluster() (in module dpGen.generator.lib.gaussian), 290
 tar_compress:
 init_bulk_mdata/fp/machine[SSHContext]/remote_profile (Argument), 84
 init_reaction_mdata/build/machine[SSHContext]/remote_profile (Argument), 130
 init_reaction_mdata/fp/machine[SSHContext]/remote_profile (Argument), 142
 init_reaction_mdata/reaxff/machine[SSHContext]/remote_profile (Argument), 117

`init_surf_mdata/fp/machine[SSHContext]/remote_profile/tar_compress`
 (Argument), 102

`run_mdata/fp/machine[SSHContext]/remote_profile/tar_compress`
 (Argument), 67

`run_mdata/model_devi/machine[SSHContext]/remote_profile/tar_compress`
 (Argument), 55

`run_mdata/train/machine[SSHContext]/remote_profile/tar_compress`
 (Argument), 43

`simplify_mdata/fp/machine[SSHContext]/remote_profile/tar_compress`
 (Argument), 196

`simplify_mdata/model_devi/machine[SSHContext]/remote_profile/tar_compress`
 (Argument), 184

`simplify_mdata/train/machine[SSHContext]/remote_profile/tar_compress`
 (Argument), 171

Task (class in `dpgen.auto_test.Task`), 268

`task_param` (`dpgen.auto_test.Property.Property` property), 267

`task_param()` (`dpgen.auto_test.Elastic.Elastic` method), 262

`task_param()` (`dpgen.auto_test.EOS.EOS` method), 261

`task_param()` (`dpgen.auto_test.Gamma.Gamma` method), 263

`task_param()` (`dpgen.auto_test.Interstitial.Interstitial` method), 264

`task_param()` (`dpgen.auto_test.Surface.Surface` method), 268

`task_param()` (`dpgen.auto_test.Vacancy.Vacancy` method), 271

`task_type` (`dpgen.auto_test.Property.Property` property), 267

`task_type()` (`dpgen.auto_test.Elastic.Elastic` method), 262

`task_type()` (`dpgen.auto_test.EOS.EOS` method), 261

`task_type()` (`dpgen.auto_test.Gamma.Gamma` method), 263

`task_type()` (`dpgen.auto_test.Interstitial.Interstitial` method), 264

`task_type()` (`dpgen.auto_test.Surface.Surface` method), 268

`task_type()` (`dpgen.auto_test.Vacancy.Vacancy` method), 271

`tau_t:`
`init_reaction_jdata/reaxff/tau_t` (Argument), 113

`taup:`
`run_jdata[model_devi_engine=lammps]/model_devi_jobs/taup`
 (Argument), 25

`taut:`
`run_jdata[model_devi_engine=lammps]/model_devi_jobs/taut`
 (Argument), 25

`temp:`
`init_reaction_jdata/reaxff/temp` (Argument), 113

`template:`

`run_jdata[model_devi_engine=lammps]/model_devi_jobs/taup`
 (Argument), 23

`run_jdata[model_devi_engine=lammps]/model_devi_jobs/taut`
 (Argument), 24

`run_jdata[model_devi_engine=lammps]/model_devi_jobs/taut`
 (Argument), 254

`run_jdata[model_devi_engine=lammps]/model_devi_jobs/taut`
 (Argument), 272

`timeout:`

`init_reaction_mdata/reaxff/machine[SSHContext]/remote_profile/tar_compress`
 (Argument), 84

`init_surf_mdata/fp/machine[SSHContext]/remote_profile/tar_compress`
 (Argument), 129

`run_mdata/fp/machine[SSHContext]/remote_profile/tar_compress`
 (Argument), 142

`init_reaction_mdata/reaxff/machine[SSHContext]/remote_profile/tar_compress`
 (Argument), 117

`init_surf_mdata/fp/machine[SSHContext]/remote_profile/tar_compress`
 (Argument), 102

`run_mdata/fp/machine[SSHContext]/remote_profile/tar_compress`
 (Argument), 67

`run_mdata/model_devi/machine[SSHContext]/remote_profile/tar_compress`
 (Argument), 54

`run_mdata/train/machine[SSHContext]/remote_profile/tar_compress`
 (Argument), 42

`simplify_mdata/fp/machine[SSHContext]/remote_profile/tar_compress`
 (Argument), 196

`simplify_mdata/model_devi/machine[SSHContext]/remote_profile/tar_compress`
 (Argument), 183

`simplify_mdata/train/machine[SSHContext]/remote_profile/tar_compress`
 (Argument), 171

`to_system_data()` (in module `dpgen.auto_test.lib.lmp`), 253

`totp_secret:`

`init_bulk_mdata/fp/machine[SSHContext]/remote_profile/tar_compress`
 (Argument), 84

`init_reaction_mdata/build/machine[SSHContext]/remote_profile/tar_compress`
 (Argument), 129

`init_reaction_mdata/fp/machine[SSHContext]/remote_profile/tar_compress`
 (Argument), 142

`init_reaction_mdata/reaxff/machine[SSHContext]/remote_profile/tar_compress`
 (Argument), 117

`init_surf_mdata/fp/machine[SSHContext]/remote_profile/tar_compress`
 (Argument), 102

`run_mdata/fp/machine[SSHContext]/remote_profile/tar_compress`
 (Argument), 67

`run_mdata/model_devi/machine[SSHContext]/remote_profile/tar_compress`
 (Argument), 55

`run_mdata/train/machine[SSHContext]/remote_profile/tar_compress`
 (Argument), 43

`simplify_mdata/fp/machine[SSHContext]/remote_profile/tar_compress`
 (Argument), 196

`simplify_mdata/model_devi/machine[SSHContext]/remote_profile/tar_compress`
 (Argument), 183

`simplify_mdata/train/machine[SSHContext]/remote_profile/tar_compress`

(Argument), 171
 train:
 run_mdata/train (Argument), 40
 simplify_mdata/train (Argument), 169
 training_args() (in module *dpgen.generator.arginfo*), 293
 training_finetune_model:
 run_jdata/training_finetune_model (Argument), 22
 simplify_jdata/training_finetune_model (Argument), 160
 training_init_frozen_model:
 run_jdata/training_init_frozen_model (Argument), 21
 simplify_jdata/training_init_frozen_model (Argument), 160
 training_init_model:
 run_jdata/training_init_model (Argument), 20
 simplify_jdata/training_init_model (Argument), 158
 training_iter0_model_path:
 run_jdata/training_iter0_model_path (Argument), 20
 simplify_jdata/training_iter0_model_path (Argument), 158
 training_reuse_iter:
 run_jdata/training_reuse_iter (Argument), 20
 simplify_jdata/training_reuse_iter (Argument), 159
 training_reuse_numb_steps:
 run_jdata/training_reuse_numb_steps (Argument), 21
 simplify_jdata/training_reuse_numb_steps (Argument), 159
 training_reuse_old_ratio:
 run_jdata/training_reuse_old_ratio (Argument), 20
 simplify_jdata/training_reuse_old_ratio (Argument), 159
 training_reuse_start_lr:
 run_jdata/training_reuse_start_lr (Argument), 21
 simplify_jdata/training_reuse_start_lr (Argument), 159
 training_reuse_start_pref_e:
 run_jdata/training_reuse_start_pref_e (Argument), 21
 simplify_jdata/training_reuse_start_pref_e (Argument), 159
 training_reuse_start_pref_f:
 run_jdata/training_reuse_start_pref_f (Argument), 21
 simplify_jdata/training_reuse_start_pref_f (Argument), 159
 trj_freq:
 run_jdata[model_devi_engine=amber]/model_devi_jobs/trj (Argument), 29
 run_jdata[model_devi_engine=lammps]/model_devi_jobs/trj (Argument), 24
 true_error_e_trust_hi:
 simplify_jdata/true_error_e_trust_hi (Argument), 158
 true_error_e_trust_lo:
 simplify_jdata/true_error_e_trust_lo (Argument), 158
 true_error_f_trust_hi:
 simplify_jdata/true_error_f_trust_hi (Argument), 158
 true_error_f_trust_lo:
 simplify_jdata/true_error_f_trust_lo (Argument), 158
 type_map:
 init_bulk_jdata/type_map (Argument), 80
 init_reaction_jdata/type_map (Argument), 112
 run_jdata/type_map (Argument), 18
 simplify_jdata/type_map (Argument), 155

U

uniq() (in module *dpgen.data.tools.cessp2force_lin*), 274
 universal() (in module *dpgen.auto_test.lib.mfp_eosfit*), 257
 update_dict() (in module *dpgen.generator.lib.cp2k*), 289
 update_mass_map() (in module *dpgen.generator.run*), 297
 use_clusters:
 run_jdata[fp_style=gaussian]/use_clusters (Argument), 32
 run_jdata[fp_style=siesta]/use_clusters (Argument), 34
 simplify_jdata[gaussian]/use_clusters (Argument), 162
 simplify_jdata[siesta]/use_clusters (Argument), 163
 use_ele_temp:
 run_jdata/use_ele_temp (Argument), 19
 simplify_jdata/use_ele_temp (Argument), 156
 use_relative:
 run_jdata[model_devi_engine=lammps]/use_relative (Argument), 29
 use_relative_v:
 run_jdata[model_devi_engine=lammps]/use_relative_v (Argument), 29
 user_backward_files:

init_bulk_mdata/fp/user_backward_files
 (Argument), 93
 init_reaction_mdata/build/user_backward_files
 (Argument), 139
 init_reaction_mdata/fp/user_backward_files
 (Argument), 151
 init_reaction_mdata/reaxff/user_backward_files
 (Argument), 127
 init_surf_mdata/fp/user_backward_files
 (Argument), 111
 run_mdata/fp/user_backward_files (Argument), 76
 run_mdata/model_devi/user_backward_files
 (Argument), 64
 run_mdata/train/user_backward_files
 (Argument), 52
 simplify_mdata/fp/user_backward_files
 (Argument), 205
 simplify_mdata/model_devi/user_backward_files
 (Argument), 193
 simplify_mdata/train/user_backward_files
 (Argument), 181
 user_forward_files:
 init_bulk_mdata/fp/user_forward_files
 (Argument), 93
 init_reaction_mdata/build/user_forward_files
 (Argument), 139
 init_reaction_mdata/fp/user_forward_files
 (Argument), 151
 init_reaction_mdata/reaxff/user_forward_files
 (Argument), 126
 init_surf_mdata/fp/user_forward_files
 (Argument), 111
 run_mdata/fp/user_forward_files (Argument), 76
 run_mdata/model_devi/user_forward_files
 (Argument), 64
 run_mdata/train/user_forward_files (Argument), 52
 simplify_mdata/fp/user_forward_files (Argument), 205
 simplify_mdata/model_devi/user_forward_files
 (Argument), 193
 simplify_mdata/train/user_forward_files
 (Argument), 180
 user_fp_params:
 run_jdata[fp_style=abacus]/user_fp_params
 (Argument), 37
 run_jdata[fp_style=cp2k]/user_fp_params
 (Argument), 36
 run_jdata[fp_style=pwscf]/user_fp_params
 (Argument), 39
 simplify_jdata[abacus]/user_fp_params
 (Argument), 166
 simplify_jdata[cp2k]/user_fp_params
 (Argument), 165
 simplify_jdata[pwscf]/user_fp_params (Argument), 167
 init_bulk_mdata/fp/machine[SSHContext]/remote_profile/
 (Argument), 83
 init_reaction_mdata/build/machine[SSHContext]/remote_p
 (Argument), 128
 init_reaction_mdata/fp/machine[SSHContext]/remote_prof
 (Argument), 141
 init_reaction_mdata/reaxff/machine[SSHContext]/remote_
 (Argument), 116
 init_surf_mdata/fp/machine[SSHContext]/remote_profile/
 (Argument), 101
 run_mdata/fp/machine[SSHContext]/remote_profile/userna
 (Argument), 66
 run_mdata/model_devi/machine[SSHContext]/remote_profil
 (Argument), 53
 run_mdata/train/machine[SSHContext]/remote_profile/use
 (Argument), 42
 simplify_mdata/fp/machine[SSHContext]/remote_profile/u
 (Argument), 195
 simplify_mdata/model_devi/machine[SSHContext]/remote_p
 (Argument), 182
 simplify_mdata/train/machine[SSHContext]/remote_profil
 (Argument), 170
V
 Vacancy (class in dpgen.auto_test.Vacancy), 271
 vacuum_max:
 init_surf_jdata/vacuum_max (Argument), 97
 vacuum_min:
 init_surf_jdata/vacuum_min (Argument), 97
 vacuum_numb:
 init_surf_jdata/vacuum_numb (Argument), 97
 vacuum_resol:
 init_surf_jdata/vacuum_resol (Argument), 97
 VASP (class in dpgen.auto_test.VASP), 269
 VaspInput (class in dpgen.database), 281
 VaspInput (class in dpgen.database.vasp), 285
 vfnnet() (in module dpgen.auto_test.lib.mfp_eosfit), 257
 vinet_pv() (in module dpgen.auto_test.lib.mfp_eosfit),
 257
 voigt_to_stress() (in module dp-
 gen.auto_test.lib.util), 258
W
 wait_time:
 init_bulk_mdata/fp/resources/wait_time
 (Argument), 89
 init_reaction_mdata/build/resources/wait_time
 (Argument), 135

`init_reaction_mdata/fp/resources/wait_time`
 (*Argument*), 148
`init_reaction_mdata/reaxff/resources/wait_time`
 (*Argument*), 123
`init_surf_mdata/fp/resources/wait_time`
 (*Argument*), 107
`run_mdata/fp/resources/wait_time` (*Argument*), 72
`run_mdata/model_devi/resources/wait_time`
 (*Argument*), 60
`run_mdata/train/resources/wait_time`
 (*Argument*), 48
`simplify_mdata/fp/resources/wait_time`
 (*Argument*), 201
`simplify_mdata/model_devi/resources/wait_time`
 (*Argument*), 189
`simplify_mdata/train/resources/wait_time`
 (*Argument*), 177
`worker()` (*in module dpgen.auto_test.common_prop*),
 272
`write_file()` (*dpgen.database.DPPotcar method*), 280
`write_file()` (*dpgen.database.vasp.DPPotcar*
method), 285
`write_incar_dict()` (*in module dp-*
gen.generator.lib.vasp), 292
`write_input()` (*dpgen.database.vasp.VaspInput*
method), 287
`write_input()` (*dpgen.database.VaspInput method*),
 283
`write_input()` (*in module dpgen.auto_test.lib.abacus*),
 252
`write_input_dict()` (*in module dp-*
gen.generator.lib.pwmat), 291
`write_kpt()` (*in module dpgen.auto_test.lib.abacus*),
 252
`write_model_devi_out()` (*in module dp-*
gen.generator.lib.make_calypso), 290

Z

`z_min:`
`init_surf_jdata/z_min` (*Argument*), 97