ASE Manual

Amsterdam Modeling Suite 2023.1

www.scm.com

Mar 31, 2023

CONTENTS

CHAPTER

ONE

USAGE

The **ASE engine** (this manual) is the interface between any existing ASE calculator (https://wiki.fysik.dtu.dk/ase/ase/calculators/calculators.html#supported-calculators) and the AMS Driver.

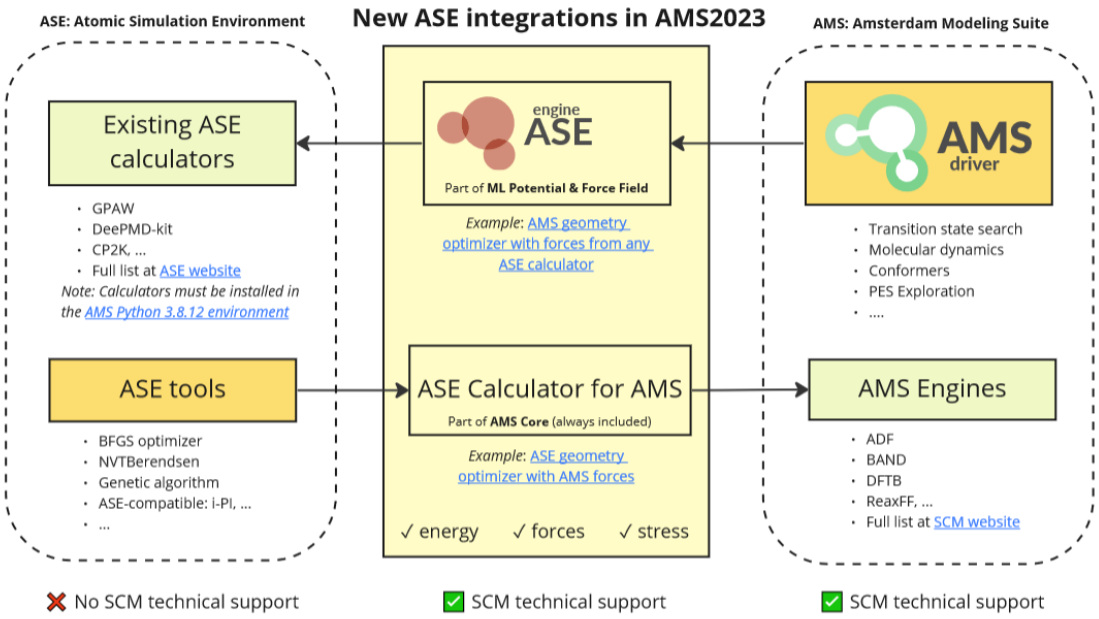
Important: The calculator must be installed in the AMS Python environment.

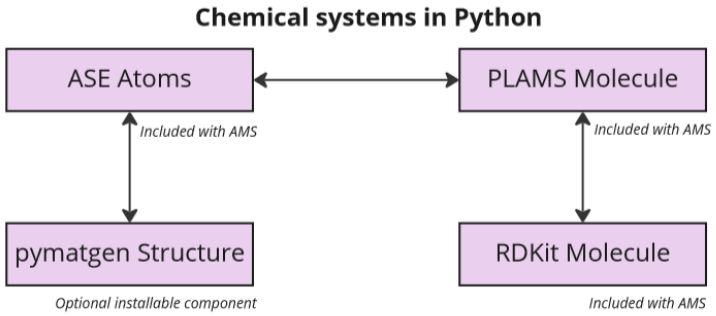
If you launch `$AMSBIN/amspython`, you need to be able to import all the needed packages. Otherwise, make sure to install them.

The ASE Calculator for AMS (AMSCalculator) is described in a different manual.

More information about ASE can be found in ref.[1] or on the ASE website (https://wiki.fysik.dtu.dk/ase/).

[1] A. H. Larsen, J. J. Mortensen, J. Blomqvist, I. E. Castelli, R. Christensen, M. Dułak, J. Friis, M. N. Groves, B. Hammer, C. Hargus, E. D. Hermes, P. C. Jennings, P. B. Jensen, J. Kermode, J. R. Kitchin, E. L. Kolsbjerg, J. Kubal, K. Kaasbjerg, S. Lysgaard, J. Bergmann Maronsson, T. Maxson, T. Olsen, L. Pastewka, A. Peterson, C. Rostgaard, J. Schiøtz, O. Schütt, M. Strange, K. S. Thygesen, T. Vegge, L. Vilhelmsen, M. Walter, Z. Zeng, K. W. Jacobsen. J. Phys.: Condens. Matter Vol. 29 (2007) 273002 https://iopscience.iop.org/article/10.1088/1361-648X/aa680e





1.1 What's new in AMS2023.1?

- The ASE engine is new in AMS2023.

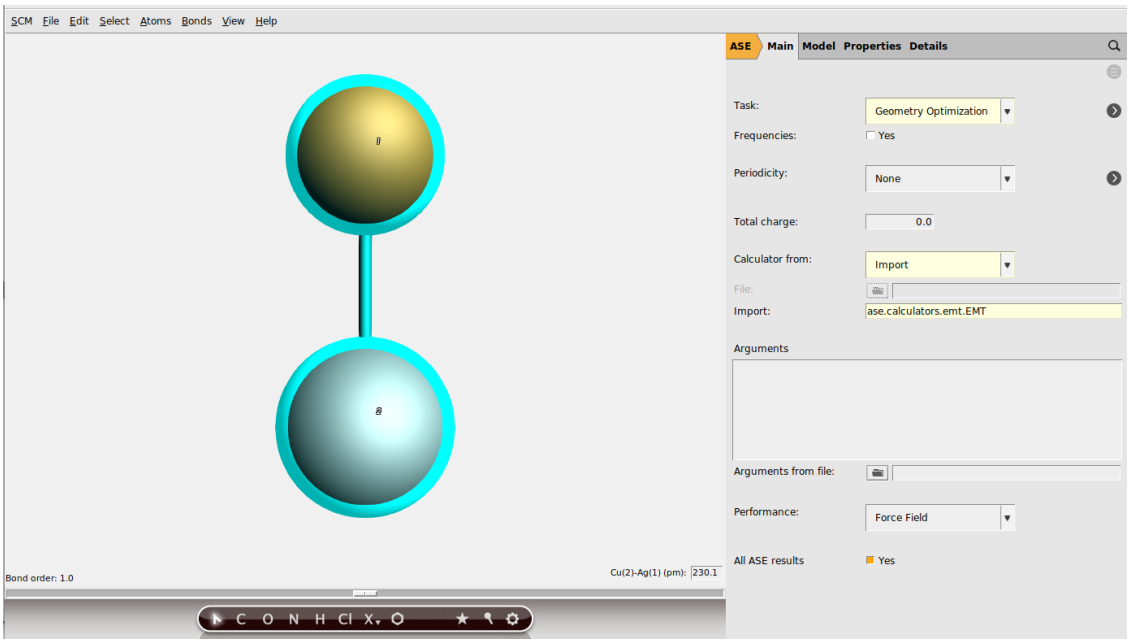
1.2 Quickstart guide

1.2.1 Quickstart with GUI

- Start AMSinput and switch to the ASE engine in the yellow dropdown.
- In the `Calculator from` drop-down, select `Import`
- In the `Import` field, type `ase.calculators.emt.EMT` (more information: ASE implementation of EMT (https://databases.fysik.dtu.dk/ase/ase/calculators/emt.html))
- Copy and paste the following atom coordinates into AMSinput

```
Ag 0. 0. 0.
Cu 0. 0. 2.0
```

- Run the calculation
- Open the trajectory in AMSmovie, or update the geometry in AMSinput



1.2.2 Quickstart with Python

See the PLAMS example: Engine ASE: AMS geometry optimizer with forces from any ASE Calculator

1.2.3 Quickstart with command-line

Run the below example:

```
$AMSBIN/ams <<eor
Task SinglePoint
Properties
    Gradients Yes
End

System
    Atoms
        Ag 0. 0. 0.
        Cu 0. 0. 2.0
    End
End

Engine ASE
   Type Import
   Import ase.calculators.emt.EMT
EndEngine
eor
```

1.3 Engine input

1.3.1 Calculator from Import

If a Calculator is installed in the AMS python stack, it can be selected by specifying the module and name of the Calculator in `Import`.

Example without arguments

The EMT calculator is a fast method applicable to many systems, especially metals.

```
Engine ASE
   Type Import
   Import ase.calculators.emt.EMT
EndEngine
```

Example with arguments

```
Engine ASE
    Type Import
    Import ase.calculators.harmonic.SpringCalculator
    Arguments
      ideal_positions=[[0.2,0.0,0.0],[0.8,0.0,0.0]]
      k=2.0
    End
EndEngine
```

1.3.2 Calculator from Python file

A Calculator is selected by providing a python file through `File` and should contain a function or class named *get_calculator* which returns an initialized ASE Calculator.

Example without arguments

The AMS input file contains:

```
Engine ASE
   Type File
   File /path/to/pythonfile.py
EndEngine
```

With *pythonfile.py* containing e.g.

```
from ase.calculators.harmonic import SpringCalculator
def get_calculator():
   return SpringCalculator(ideal_positions=[[0.2,0.0,0.0],[0.8,0.0,0.0]], k=2.0)
```

Example with arguments

```
Engine ASE
   Type File
   File /path/to/pythonfile.py
   Arguments
      ideal_positions=[[0.2,0.0,0.0],[0.8,0.0,0.0]]
      k=2.0
   End
EndEngine
```

With *pythonfile.py* containing e.g.

```
from ase.calculators.harmonic import SpringCalculator
def get_calculator(ideal_positions, k):
   return SpringCalculator(ideal_positions=ideal_positions, k=k)
```

1.3.3 Specifying arguments for the Calculator

Arguments can be specified either with the `Arguments` argument as above, or with `ArgumentsFromFile`.

`ArgumentsFromFile` is either a yaml file with the .yml extension or a python file with the .py extension.

1.3.4 Obtaining results from the Calculator

When available, the *energy*, *forces* and *stress* are always obtained and are fully integrated in AMS. If a Calculator holds any additional results in its results dictionary, then by default they are stored in the *Other* section of ase.rkf, but without any unit conversions. If additional results are undesired, they can be turned off through `AllASEResults` and specific results can be requested by setting `Results`.

Example. The AMS input file contains:

```
Engine ASE
   Type File
   File custom/calculator.py
   AllASEResults no
   Results specific_result
EndEngine
```

See also:

Tutorial: 10 Ways to Get the Energy and Other Properties

1.4 Troubleshooting

- If the required Calculator needs advanced setup or input arguments, it is usually more convenient to use the python input style and handle such details in there.
- The *Arguments* block is stripped of any indentation and comments ("!", "#" and "::"). If this is not desirable, use *ArgumentsFromFile* or python input style instead.
- If the Calculator accepts any arguments related to files, make sure to provide **absolute paths** and not relative paths. The Calculator does not run in the same directory as AMS to avoid conflicts.

1.5 Support

SCM does not provide support for any ASE calculators.

1.6 Licensing

In AMS2023, the ASE engine is part of the product “ML Potentials & Classical Force Fields”.

1.7 References

CHAPTER

TWO

AMS DRIVER'S TASKS AND PROPERTIES

The ASE engine is an engine used by the AMS driver. While the specific options for the ASE engine are described in this manual, the definition of the system, the selection of the task and certain (potential-energy-surface-related) properties are documented in the AMS driver's manual.

In this page you will find useful links to the relevant sections of the AMS driver's Manual.

2.1 Geometry, System definition

The definition of the system, i.e. the atom types and atomic coordinates (and optionally, the lattice vectors and atomic masses for isotopes) are part of the AMS driver input. See the System definition section of the AMS manual.

2.2 Tasks: exploring the PES

The job of the AMS driver is to handle all changes in the simulated system's geometry, e.g. during a geometry optimization or molecular dynamics calculation, using energy and forces calculated by the engine.

These are the tasks available in the AMS driver:

- GCMC (Grand Canonical Monte Carlo)
- Geometry Optimization
- IRC (Intrinsic Reaction Coordinate)
- Molecular Dynamics
- NEB (Nudged Elastic Band)
- PESScan (Potential Energy Surface Scan, including linear transit)
- Single Point
- Transition State Search
- Vibrational Analysis

2.3 Properties in the AMS driver

The following properties can be requested to the ASE engine in the AMS driver's input:

- Elastic tensor
- Hessian
- Nuclear gradients (forces)
- Normal modes
- PES point character
- Phonons
- Stress tensor
- Thermodynamic properties

CHAPTER

THREE

ASE KEYWORDS

`AllASEResults`

Type Bool

Default value Yes

Recurring False

GUI name All ASE results

Description Return all ASE results that are not also part of AMSResults. These values can be found in ase.rkf without any unit conversions.

`Arguments`

Type Non-standard block

Description Python style arguments given to the function or class defined in *Calculator*. This is case sensitive.

`ArgumentsFromFile`

Type String

Default value

Description Specify the path to a yaml or python file defining the arguments to the function or class defined in *Calculator* or *Callable*.

`File`

Type String

Default value

Description Specify the path to a python file if it is not installed in the ams python environment. This file should define a callable (e.g. function or class) named *get_calculator* that returns an ASE Calculator and uses the arguments defined in *Arguments* and *ArgumentsFromFile*.

`Import`

Type String

Default value

Description Specify the module and name of a Calculator installed in the AMS python stack, e.g. 'module.submodule.DFTCalculator'. This is case sensitive.

`Performance`

Type Multiple Choice

Default value ForceField

Options [Fast, ForceField, DFTB, DFT, Slow]

Description Choose which option most accurately corresponds to how long a calculation with the calculator takes.

`Results`

Type String

Recurring True

Description The data of this key in the results dictionary of the Calculator is stored in the engine rkf. Multiple results keys can be specified. This is case sensitive.

`Type`

Type Multiple Choice

Default value File

Options [File, Import]

GUI name Calculator from

Description Select how to specify which calculator to use.

INDEX