



Introduction to scripting

NSCCS ADF/ReaxFF Workshop

Ole Carstensen
Fedor Goumans
Anna Shchygol

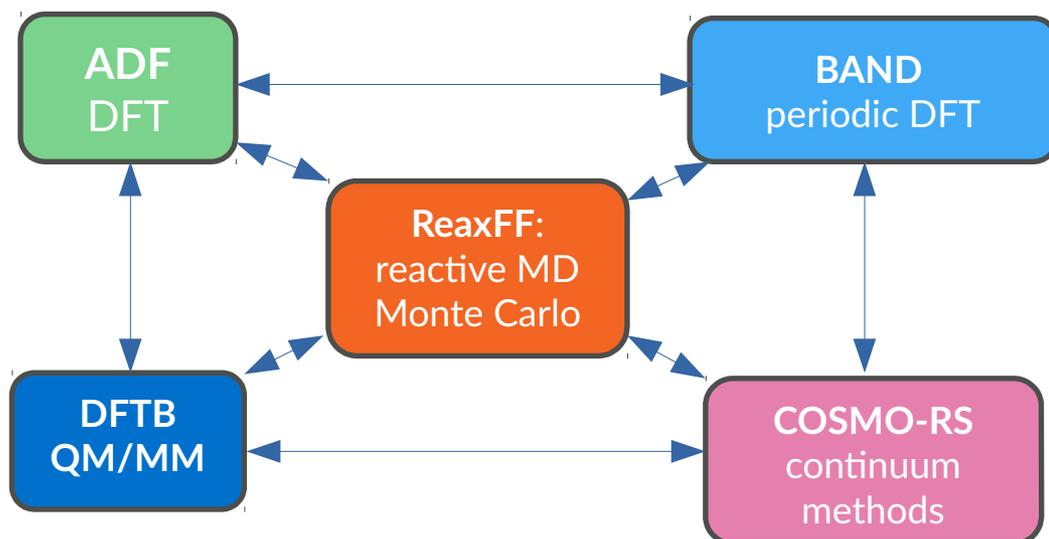
Imperial College London
27-28 September 2016

Scripting



The ADF modelling suite comes with its own Python:

```
$ADFBIN/startpython
```



Links all program packages

→ workflows

→ custom post-processing/analysis

→ **rapid prototyping in development**

Available modules



fundamental package
for scientific computing



Cheminformatics
and Machine Learning

ASE (+ calculators)

Atomic simulation environment

PLAMS

Python Library for Automating
Molecular Simulations developed @SCM

more...

<https://www.scm.com/doc/Scripting/index.html>

ADF/ReaxFF: RXKF files

The RXKF format

Info:

The RXKF file is the general result file of an ADF/ReaxFF calculation.

It is a KF file: Direct-Access, binary, and keyword driven (see [here](#)). It contains your trajectory as well as information about the calculation.

Use the KFBrowser of the GUI to inspect RXKF files:

KFBrowser 2016.205+ (r53730): defected_graphene.rxkf

File **Edit** **View** **Graph**

General

- Name: defected_graphene
- uid
- account: Dr. Ole Carstensen / SCM / Amsterdam / NL
- Energy terms
- Scalars
- Scalars units: 1920 chars
- Time step s: 2e-16
- Step numbers: 8i##10, 100 ints
- Geometry: 0 bytes
- History: 2,872,508 bytes
- Molecules: 107,536 bytes

Q

Make sure to always use the KFBrowser in “Expert Mode”:

CTRL + E

or select

“File” → “Expert Mode”

ADF/ReaxFF: RXKF files

How to look up Sections and Variables...

The data on the RXKF is organized in Sections which group together related data. Each section contains a number of variables. Each variable may be an array or a scalar and may be integer, real, logical or character type.

The screenshot shows the KFBrowser interface with the 'History' section expanded. A red box highlights the 'History' section name, and a red arrow points to it with the label 'Section'. Another red box highlights the 'ConnTab num neighb 0' variable, and a red arrow points to it with the label 'Variable'. The variable is an array of 648 integers, with the first 10 values shown in the table below.

Section	Variable	Value	Type
History	ConnTab num neighb 0	5	ints
	ConnTab num neighb 0	5	ints
	ConnTab num neighb 0	4	ints
	ConnTab num neighb 0	3	ints
	ConnTab num neighb 0	3	ints
	ConnTab num neighb 0	3	ints
	ConnTab num neighb 0	3	ints
	ConnTab num neighb 0	3	ints
	ConnTab num neighb 0	3	ints
	ConnTab num neighb 0	3	ints

Here:

Number of neighbors for each atom in the first frame: Section = "History", Variable = "ConnTab num neighb 0" (in this case the variable is an array with 648 integer entries, but PYTHON will take care of this for us...)

Python Scripting

using PLAMS to read from rxkf



Data from RXKF files can easily be read and processed using PYTHON and the PLAMS library. Just modify the following template according to your own needs:



File: `plams_template.py`

Execute: `startpython plams_template.py`

Reading General Data, Examples:

```
#
# Read the timestep
#
timestep = KFReader.read('General', 'Time step s')
print timestep

[...]

#
# Read the stepnumbers
#
steps = KFReader.read('General', 'Step numbers')
print steps
```

Documentation is found [here](#)

Python Scripting

using PLAMS to read from rxkf



The most common situation will be to process some entries of an RXKF file per frame. The following code snippet demonstrates how to read the Number of Molecules / frame:



File: plams_template.py

Execute: startpython plams_template.py

Reading Data per Frame

```
#
# Read the steps
# remove -1 from the returned list
#
Steps = KFReader.read('General', 'Step numbers')
Steps = filter(lambda a: a != -1, Steps)
#
# Loop through all frames and read
# current number of molecules
#
for step in Steps:
    NumMols = KFReader.read('History', 'ConnTab num mols ' + str(step))
    print NumMols
```

Documentation is found [here](#)



contact us:

Licenses

General information

User support

license@scm.com

info@scm.com

support@scm.com