Technische Universität Braunschweig

Suresoft
Sustainable Research Software

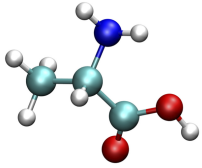
Institut für Physikalische und Theoretische Chemie

PyADF – A scripting framework for multiscale quantum chemistry

Mario Wolter, 16. Mai 2022

Theoretical Chemistry

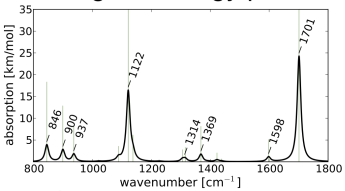
⇒ **Calculation of molecular properties**



$$\hat{H}\Psi = E\Psi$$

molecular structure

energies, energy profiles



spectra
molecular properites

***ab initio*: Based on quantum mechanics**

- Molecules consist of electrons and atomic nuclei

Quantum Chemistry Program Packages

Big, highly optimized, monolithic codes for single molecules

- e.g., ADF, Orca, Turbomole, NWChem, Quantum Espresso, …
- often developed since 30–40 years
- programmed in Fortran, more recently also C/C++

Quantum Chemistry Program Packages

Big, highly optimized, monolithic codes for single molecules

- e.g., ADF, Orca, Turbomole, NWChem, Quantum Espresso, …
- often developed since 30–40 years
- programmed in Fortran, more recently also C/C++

Problem:
Complicated workflows often require multiple individual calculations

Quantum Chemistry Program Packages

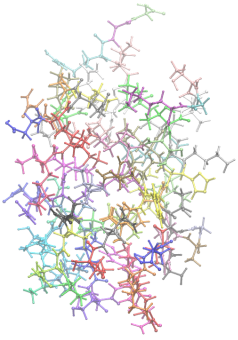
Big, highly optimized, monolithic codes for single molecules

- e.g., ADF, Orca, Turbomole, NWChem, Quantum Espresso, ...
- often developed since 30–40 years
- programmed in Fortran, more recently also C/C++

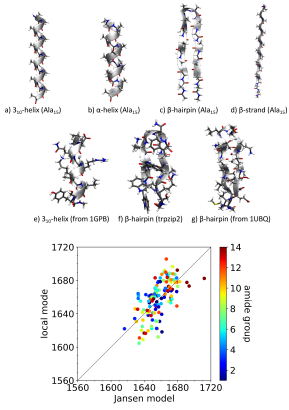
Problem:
Complicated workflows often require multiple individual calculations

$\Rightarrow$ can be addressed using a scripting framework

Examples of Complicated Workflows



Subsystem DFT



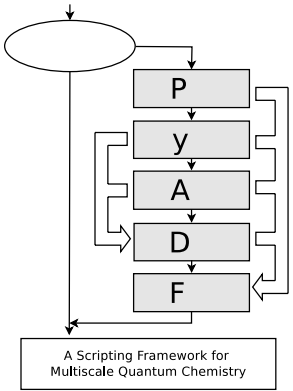
Vibrational Spectra

PyADF

Python scripting framework for multiscale quantum chemistry

- uniform framework for different quantum chemical tasks
- makes uses of object-oriented programming techniques

$\Rightarrow$ flexible, easy-to-use scripting interface



Ch. R. Jacob *et al.*, *J. Comput. Chem.* **32**, 2328 (2011), http://www.pyadf.org.

PyADF

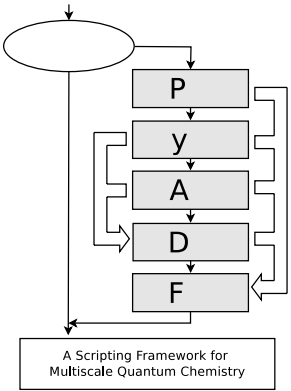
Python scripting framework for multiscale quantum chemistry

- uniform framework for different quantum chemical tasks
- makes uses of object-oriented programming techniques

⇒ flexible, easy-to-use scripting interface

- PyADF input files are simple Python scripts

⇒ full power of Python available



Ch. R. Jacob *et al.*, *J. Comput. Chem.* **32**, 2328 (2011), http://www.pyadf.org.

Current Status

PyADF Scripting Framework

- development and use for several research projects for almost 15 years
- publication (2011) has been cited >55 times
- developers and users in several theoretical chemistry groups, e.g.
  - Prof. Dr. Lucas Visscher (VU University Amsterdam, Netherlands)
  - Dr. André Gomes (CRNS and Université Lille, France)
  - Prof. Dr. Michele Pavanello (Rutgers University Newark, USA)
  - Prof. Dr. Johannes Neugebauer (WWU Münster)

Before Suresoft - Software Engineering Status

- about 28k lines of Python2 code
  $\rightarrow$ migration to Python3 was imminent

Before Suresoft - Software Engineering Status

- about 28k lines of Python2 code
  → migration to Python3 was imminent
- Gitlab repository on our own server
  - new functionality is added to master branch by merge requests from developers
  - maintainer checks for compatibility of merge requests manually

Before Suresoft - Software Engineering Status

- about 28k lines of Python2 code
  → migration to Python3 was imminent
- Gitlab repository on our own server
  - new functionality is added to master branch by merge requests from developers
  - maintainer checks for compatibility of merge requests manually
- 88 "integration"/functionality tests
  - functionality is tested in "two dimensions"
    - functionality of PyADF itself
    - functionality of the interface to QC software packages
- some unit tests, e.g. for "PlotGridFunctions"

Before Suresoft - Software Engineering Status

- about 28k lines of Python2 code
  → migration to Python3 was imminent
- Gitlab repository on our own server
  - new functionality is added to master branch by merge requests from developers
  - maintainer checks for compatibility of merge requests manually
- 88 "integration"/functionality tests
  - functionality is tested in "two dimensions"
    - functionality of PyADF itself
    - functionality of the interface to QC software packages
- some unit tests, e.g. for "PlotGridFunctions"
- functionality tests run every night (cron job) and report results via mail

Challenges

- continuous integration
  → unit and integration tests for all functionality

Challenges

- continuous integration
  → unit and integration tests for all functionality
- virtualization and deployment
  → Docker containers complete with (commercial) software packages
  → or interaction between containers running different packages

Challenges

- continuous integration
  → unit and integration tests for all functionality
- virtualization and deployment
  → Docker containers complete with (commercial) software packages
  → or interaction between containers running different packages
- archival of quantum-chemical research data
  → using PyADF as an infrastructure for archival of research data
  → develop suitable data and metadata formats

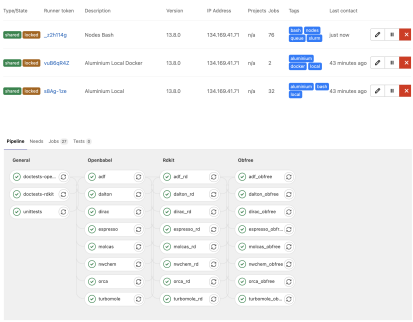
Challenges

- continuous integration
  → unit and integration tests for all functionality
- virtualization and deployment
  → Docker containers complete with (commercial) software packages
  → or interaction between containers running different packages
- archival of quantum-chemical research data
  → using PyADF as an infrastructure for archival of research data
  → develop suitable data and metadata formats
- adding new functionality using these key developments

Sustainable Software Development

Continuous Integration with Gitlab-CI

- different Gitlab runners (using tags)
- subdivided pipeline (based on test_pyadf script)
- automatic testing on merge requests or "on demand"
- error reporting through web interface and mail



Sustainable Software Development

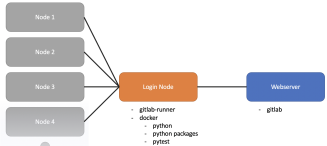
Using HPC nodes for Gitlab runner

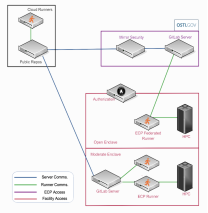
- custom Gitlab runner
- using Jacamar (Exascale Computing Project)
- load is distributed on cluster nodes
- access for users without cluster account

Additional approach: HPCRocket

- python scripts to send slurm commands
- developed within Suresoft project

https://github.com/SvenMarcus/hpc-rocket





Current Software Engineering Status

- **more than 40k** lines of Python code
  → migration to Python3 **successfully completed**
- Gitlab repository on our own server
  - new functionality is added to master branch by merge requests from developers
  - **automatic CI pipeline for all merge requests and on demand**
- currently **129** "integration"/functionality tests
  - functionality is tested in "two dimensions"
    - functionality of PyADF itself
    - functionality of the interface to QC software packages
- some unit tests, e.g. for "PlotGridFunctions"
- **automatic testing pipeline**
  - **with specified builds of QC software packages**
  - **tests are distributed to cluster nodes**

Future development

- virtualization and deployment
  - → improve packaging and installation of PyADF (pip, requirements.txt, etc.)
  - → containerization (Docker/Singularity)
- archival of quantum-chemical research data
  - → use PyADF as an infrastructure for archival of research data
  - → develop suitable data and metadata formats
- Spread PyADF and Suresoft to the world!