AiiDA Siesta

Emanuele Bosoni (ICMAB-CSIC, Barcelona)
Vladimir Dikan
Pol Febrer
Victor M. Garcia-Suarez
Alberto Garcia

21 June 2021

ICMAB INSTITUT DE CIÈNCIA DE MATERIALS DE BARCELONA
EXCELENCIA SEVERO OCHOA
MAX DRIVING THE EXASCALE TRANSITION

High-Throughput and Computational Materials Science

When high volumes of data can be produced without much effort, convenient to calculate/measure properties of an entire category of choice (screening) and to analyze the data afterwards.

Extraction of property correlations used to guide the search for systems with ad-hoc characteristics.

$\Downarrow$

Materials Discovery

High-Throughput and Computational Materials Science

When high volumes of data can be produced without much effort, convenient to calculate/measure properties of an entire category of choice (screening) and to analyze the data afterwards.

Extraction of property correlations used to guide the search for systems with ad-hoc characteristics.

$\Downarrow$

Materials Discovery

High-Throughput approach is the process of creating a database of calculated material properties

- standardize the calculation process (workflows)
- attention to data organization since most of the science is done during the data analysis

Dedicated infrastructure needed....

Overview

AiiDA

Automated Interactive Infrastructure and Database for Computational Science

https://www.aiida.net

Scientific Data **7**, 300 (2020)

ADES

Automation	Data	Environment	Sharing
Remote management Coupling to data High-throughput	Storage Provenance Database	High-level workspace Scientific workflows Data analytics	Social ecosystem Standardization Repository pipelines
Abstract away the low-level tasks to prepare, submit, retrieve and store automatically large numbers of calculations	Management and persistence of heterogeneous simulation data; database search and query; reproducibility	Natural, high-level environment to encode complex sequences of low-level codes into scientific workflows and turnkey solutions	Social ecosystem to foster interactions, share codes, data and scientific workflows in open repositories, and promote standardized formats

Basic concepts: data and processes

Tutorials `https://aiida-tutorials.readthedocs.io/en/latest/`

Basic concepts: data and processes

Tutorials `https://aiida-tutorials.readthedocs.io/en/latest/`

Data Types

Python classes hosting data, allow storing in the database and provenance.

Simple wrappers of python types.

```
from aiida.orm import Float
vols = Float(7)
```

Material science related objects.

```
from aiida.orm import StructureData
struct = StructureData(ase=ase_struct)
```

Effort to standardize description of physical quantities.

`https://aiida-common-workflows.readthedocs.io/en/latest/`

Basic concepts: data and processes

Tutorials https://aiida-tutorials.readthedocs.io/en/latest/

Data Types

Python classes hosting data, allow storing in the database and provenance.

Simple wrappers of python types.

```
from aiida.orm import Float
vols = Float(7)
```

Material science related objects.

```
from aiida.orm import StructureData
struct = StructureData(ase=ase_struct)
```

Effort to standardize description of physical quantities.

https://aiida-common-workflows.readthedocs.io/en/latest/

Calculations plugins

Interface with codes

```
from aiida.plugins import CalculationFactory
from aiida.engine import submit
my_calc = CalculationFactory("plugin.name")
submit(my_calc, **inputs)
```

Basic concepts: data and processes

Tutorials `https://aiida-tutorials.readthedocs.io/en/latest/`

Data Types

Python classes hosting data, allow storing in the database and provenance.

Simple wrappers of python types.

```
from aiida.orm import Float
vols = Float(7)
```

Material science related objects.

```
from aiida.orm import StructureData
struct = StructureData(ase=ase_struct)
```

Effort to standardize description of physical quantities.

`https://aiida-common-workflows.readthedocs.io/en/latest/`

Calculations plugins

Interface with codes

```
from aiida.plugins import CalculationFactory
from aiida.engine import submit
my_calc = CalculationFactory("plugin.name")
submit(my_calc, **inputs)
```

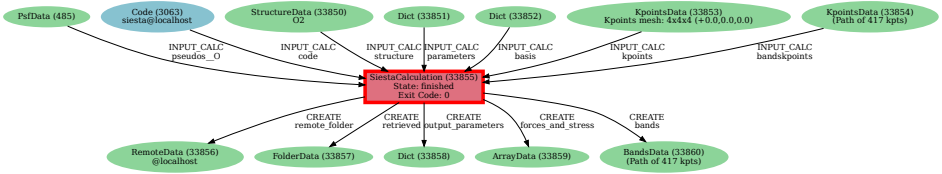
Workflows

Encode complex steps of scientific workflow

```
from aiida.plugins import WorkflowFactory
from aiida.engine import submit
my_wc = WorkflowFactory("workflow.name")
submit(my_wc, **inputs)
```

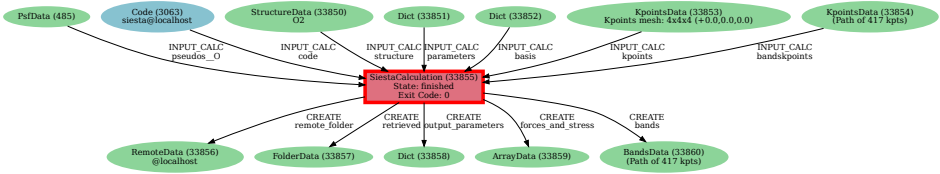
Basic concepts: provenance and database queries

All inputs and outputs are connected. Provenance is maintained.



Basic concepts: provenance and database queries

All inputs and outputs are connected. Provenance is maintained.



Very flexible way to query the database.

```
qb = QueryBuilder()
qb.append(SiestaCalculation, tag='calcjob')
qb.append(Dict, with_outgoing='calcjob', filters='attributes.mesh-cutoff': "800 Ry")
```

Ability to group objects and tag them ⇒ Export clean databases to share.

```
group = Group(label='big_meshcutoff')
group.add_nodes(qb.all())
```

Overview

Calculations 2 | Parsers 2 | Data 3 | Workflows 7

- Source code: `https://github.com/siesta-project/aiida_siesta_plugin`
- Docs: `https://aiida-siesta-plugin.readthedocs.io/en/latest/`
- Latest version requires aiida-core >= 1.3.0
- Installation: `pip install aiida-siesta` or git clone
- Paper: A. Garcia *et al.* J. Chem. Phys. **152**, 204108 (2020)
  `https://doi.org/10.1063/5.0005077`

Capabilities

1. Fully flexible in accepting Siesta inputs ⇒ able to run all the options of the Siesta code.

Code	StructureData	Dict	KpointsData	PsfData or PsmlData	IonData
code	structure	parameters basis	kpoints bandskpoints	pseudos	ions

```
siesta_calc = CalculationFactory("siesta.siesta")
submit(siesta_calc, **inputs)
```

Capabilities

1. Fully flexible in accepting Siesta inputs ⇒ able to run all the options of the Siesta code.

Code	StructureData	Dict	KpointsData	PsfData or PsmlData	IonData
code	structure	parameters basis	kpoints bandskpoints	pseudos	ions

```
siesta_calc = CalculationFactory("siesta.siesta")
submit(siesta_calc, **inputs)
```

2. Limited only by the quantity parsed and by the support of external post-processing tools. At the moment:

- Energies (Total, Efermi, ...)
- Total spin
- Relaxed structure
- Electronic bands

Support for ‘plstm‘ post-process code, generating STM images from LDOS

Coming soon: DOS, PDOS

Always possible to retrieve entire files and post process them locally.

Capabilities

1. Fully flexible in accepting Siesta inputs ⇒ able to run all the options of the Siesta code.

Code	StructureData	Dict	KpointsData	PsfData or PsmlData	IonData
code	structure	parameters basis	kpoints bandskpoints	pseudos	ions

```
siesta_calc = CalculationFactory("siesta.siesta")
submit(siesta_calc, **inputs)
```

2. Limited only by the quantity parsed and by the support of external post-processing tools. At the moment:

- Energies (Total, Efermi, ...)
- Total spin
- Relaxed structure
- Electronic bands

Support for ‘plstm‘ post-process code, generating STM images from LDOS

Coming soon: DOS, PDOS

Always possible to retrieve entire files and post process them locally.

Another way to run Siesta calculations, with plus of provenance and AiiDA data features.

Tools to help research

1. **The SiestaBaseWorkChain**
   Robustness in siesta calculations: deals automatically with some common errors (not converged scf/geometry, basis set specification problems).

```
from aiida.engine import submit
from aiida.plugins import WorkflowFactory
SiestaBaseWorkChain = WorkflowFactory("siesta.base")
submit(SiestaBaseWorkChain, **inputs)
```

Tools to help research

1. **The SiestaBaseWorkChain**
   Robustness in siesta calculations: deals automatically with some common errors (not converged scf/geometry, basis set specification problems).

```
from aiida.engine import submit
from aiida.plugins import WorkflowFactory
SiestaBaseWorkChain = WorkflowFactory("siesta.base")
submit(SiestaBaseWorkChain, **inputs)
```

2. **The Convergence Workflows**
   Helps convergence test on siesta parameters.

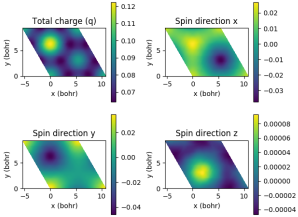
```
from aiida.engine import submit
from aiida.orm import (Str,Float)
from aiida.plugins import WorkflowFactory
SiestaConverger = WorkflowFactory("siesta.converger")
submit(SiestaConverger,
 iterate_over = {'meshcutoff':  [100, 200, 300, 400]},
 **inputs, target = Str('E_KS'), threshold = Float(0.01)
)
```

Soon basis optimization workflow.

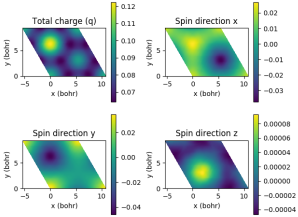
Tools to help research

3 **More “scientific” workflows**

- BandgapWorkChain.
- EqOfStateFixedCellShape
- SiestaSTMWorkChain



Tools to help research



3 More "scientific" workflows

- BandgapWorkChain.
- EqOfStateFixedCellShape
- SiestaSTMWorkChain

4 The protocol system

A system to get suggested inputs of workflows, available for all workflows.

```
from aiida.engine import submit
from aiida.plugins import WorkflowFactory
SiestaBaseWorkChain = WorkflowFactory("siesta.base")
inp_gen = SiestaBaseWorkChain.inputs_generator
filled_builder = inp_gen.get_filled_builder(structure, calc_engines, protocol)
# possibility to change inputs
submit(filled_builder)
```

"protocol" = string summarizing a choice of inputs. "calc_engines": computational resources.

Conclusions

- AiiDA helps researchers with all the aspects of high-throughput simulations with material science codes.
- Strong focus on preserving provenance and facilitate data access.
- In addition to the AiiDA capabilities, AiiDA Siesta offers some useful tools to help every day research and facilitate repetitive tasks.