The AiiDA Ecosystem for Computational Materials Science

Leopold Talirz[1,2,3], Aliaksandr V. Yakutovich[1,2,3], Sebastiaan P. Huber[1,2], Martin Uhrin[2], Spyros Zoupanos[1,2], Leonid Kahle[1,2], Conrad Johnston[1,2], Nicolas Mounet[2], Rico Häuselmann[2], Dominik Gresch[6], Tiziano Müller[1,7], Andrea Cepellotti[2], Fernando Gargiulo[2], Snehal Kumbhar[1,2], Elsa Passaro[1,2], Marco Borelli[2], Andrius Merkys[2], Ole Schütt[4], Berend Smit[1,3], Daniele Passerone[1,4], Carlo A. Pignedoli[1,4], Boris Kozinsky[8], Joost VandeVondele[1,5,,6], Thomas Schulthess[1,5,6], Nicola Marzari[1,2], Giovanni Pizzi[1,2]

[1]National Centre for Computational Design and Discovery of Novel Materials (MARVEL), École Polytechnique Fédérale de Lausanne, CH-1015 Lausanne, Switzerland
[2]Theory and Simulation of Materials (THEOS), Faculté des Sciences et Techniques de l'Ingénieur, École Polytechnique Fédérale de Lausanne, CH-1015 Lausanne, Switzerland
[3]Laboratory of Molecular Simulation (LSMO), Institut des Sciences et Ingénierie Chimiques, École Polytechnique Fédérale de Lausanne, CH-1951 Sion, Switzerland
[4]nanotech@surfaces laboratory, Swiss Federal Laboratories for Materials Science and Technology (Empa), CH-8600 Dübendorf, Switzerland
[5]Swiss National Supercomputing Centre, CH-6900 Lugano, Switzerland
[6]ETH Zürich, Switzerland
[7]University of Zurich, Switzerland
[8]Research and Technology Center, Robert Bosch LLC, Cambridge, MA 02139, USA

leopold.talirz@epfl.ch, aliaksandr.yakutovich@epfl.ch

Abstract. AiiDA (aiida.net) is a workflow manager for computational science with a strong focus on provenance, performance and extensibility. When executing a workflow, AiiDA records the provenance – calculations performed, codes used and data generated – in a directed acyclic graph tailored to provide full reproducibility of any given result. The AiiDA engine relies on a message queue in order to support high-throughput use cases of up to 50k calculations per hour, and the relational database backend enables performant queries on graphs of millions of nodes. AiiDA *plugins* can extend the core python framework in numerous ways, adding not only new workflows and connections to new simulation codes but also support for new types of job schedulers, transport protocols and extensions of the AiiDA command line interface.

While domain experts can install AiiDA on their own hardware, the AiiDA lab web platform gives novice users access to their personal AiiDA environment in the cloud, where they can run and manage workflows through tailored and lightweight web applications in the browser. The ecosystem is completed by the Materials Cloud dissemination portal, where researchers can publish their AiiDA graphs, thus providing access not only to the results of calculations, but to every

step along the way. Peers can browse the database interactively, download individual files or the whole database, and start their research right from where the original author left off.

Keywords: Computational Materials Science, Provenance, Workflows, SaaS

Today, many open questions in computational science call for more than individual computations using a single code. As the demand for integration and throughput increases, the design of robust and reproducible workflows is becoming ever more important. In this context, the move towards open science [1–3] raises the level of scrutiny and demands that workflows and data be recorded in a way that can be inspected and reused by scientific peers.

AiiDA. AiiDA is a python framework designed around the four pillars of computational science: Automation, Data, Environment and Sharing (ADES) [4]. In the default usage model, AiiDA is installed on the workstation of a researcher and connects to remote compute resources through the secure shell protocol (SSH). In order to support high-throughput use cases of 50k calculations per hour, the AiiDA daemon relies on the RabbitMQ message broker, while the PostgreSQL database backend enables performant queries on data sets of millions of nodes.

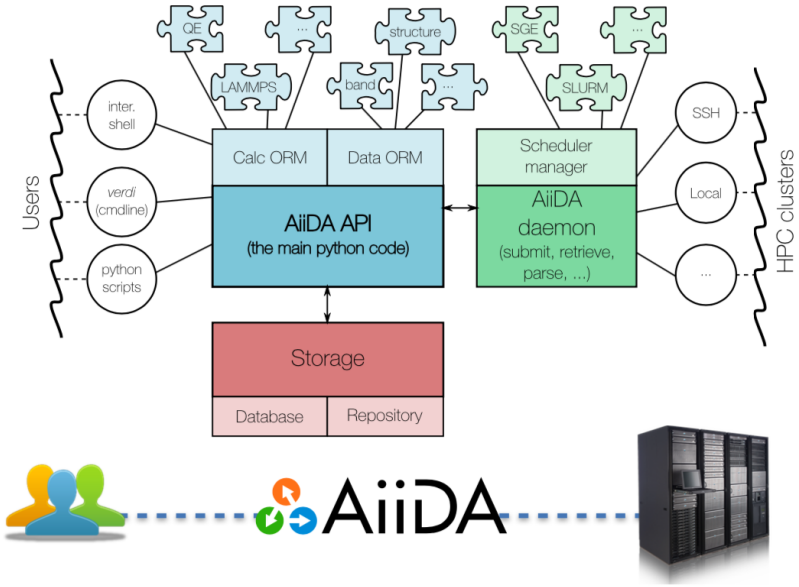

Fig. 1. AiiDA infrastructure. Users interact with AiiDA through the `verdi` command line, through interactive shell or via python scripts. AiiDA records the provenance of calculations in a database and file repository, while the AiiDA daemon automates workflows and interacts with remote compute resources. Figure reproduced with permission [4].

The focus on provenance and extensibility is a design choice that differentiates AiiDA from other workflow managers in the field of computational materials science, such as Aflow [5], atomate [6], MAST [7] or OQMD [8]. AiiDA plugins leverage python entry points to extend both the AiiDA command line interface and the python API - for example, AiiDA plugins can provide new workflows, connect to new simulation codes, provide support for new types of schedulers, transport protocols and seamlessly extend the existing command line interface. A template helps getting started with plugin development [9], and a plugin registry [10] provides a central point for registering the plugin. For example, a 2019 survey on the AiiDA mailing list points to more than 30 AiiDA-powered research projects using >25 different AiiDA plugins.

Instead of defining a new workflow markup language based on XML derivatives (Karajan [11], Askalon [12]) or JSON/yaml (Fireworks [13], Common Workflow Language [14]), AiiDA aims to make it easy for users to write workflows directly in python, providing full access to the AiiDA API, including queries of the entire provenance of previous calculations.

Since the release of the first paper [4], the provenance model of AiiDA has been extended to include a representation of workflows. While the basic building blocks of data and calculation nodes are sufficient for recording "data provenance" in a directed acyclic graph, workflow nodes provide logical abstraction by bundling several calculations (Figure 2).

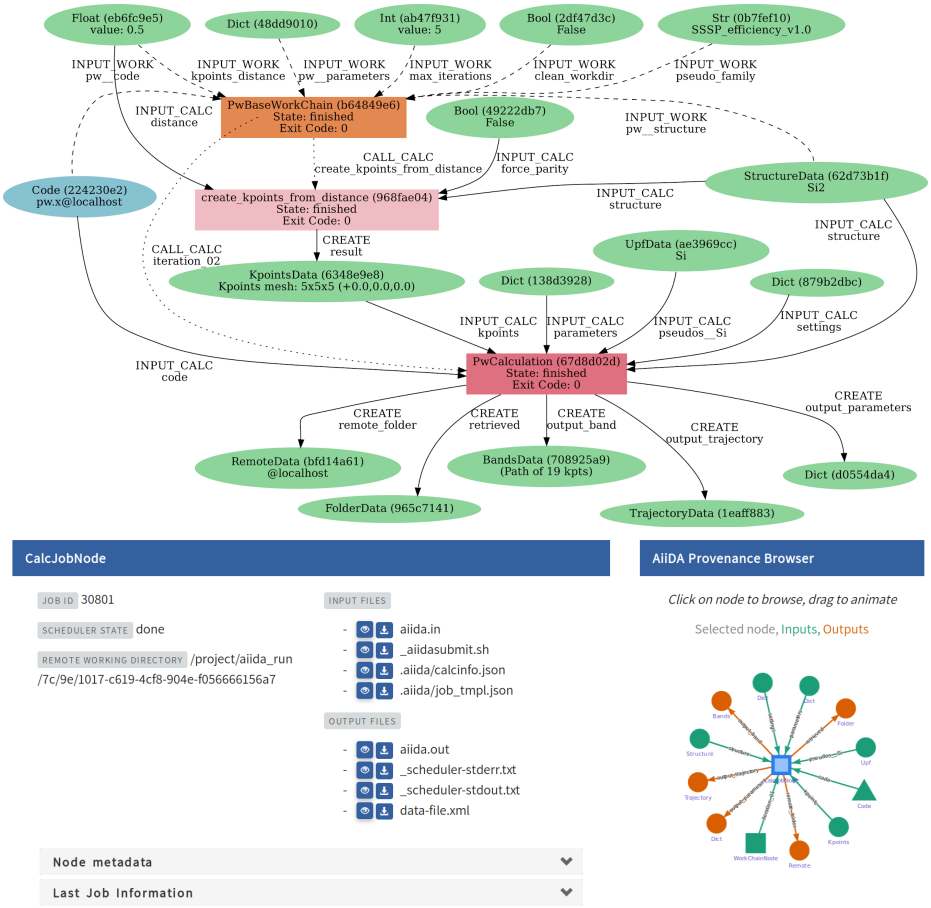

Fig. 2. AiiDA provenance graph. AiiDA tracks the provenance of data, calculations and workflows, allowing share it with other AiiDA users, to query and visualize it on the fly. (top) Autogenerated provenance graph of a calculation (red box) that takes four input data and produces five output data (green ellipses). In addition to the "data provenance layer" (solid lines), the graph includes the "logical provenance layer" with the workflow (orange box, dashed lines). While this workflow simply wraps the calculation, this might just represent the first of many workflow steps (not shown).
(bottom) Same graph viewed through the interactive provenance browser driven by the AiiDA REST API. The selected calculation node is shown in the center, with arrows to/from connected nodes.

Further updates include switching the workflow engine from a polling mechanism to a message queue, which reduces overhead for quick calculations by orders of magnitude and makes it possible to run 50k calculations per hour. Reusing one workflow in another has become easier, and workflows now are auto documenting, telling users what inputs they expect and what outputs they produce without the need to read code. AiiDA now includes measures to deal with stability issues when connecting to remote clusters (network issues, cluster down). The command line interface has been

overhauled, providing a uniform feel across all commands, dramatically increasing code reuse as well as test coverage. Writing AiiDA plugins requires significantly less boilerplate code, and AiiDA 1.0 is python3 compatible.

AiiDA was developed with the computational scientist in mind - a demographic familiar with UNIX operating systems, the terminal and python, interested in designing and tweaking complex workflows. The availability of robust materials science workflows, however, makes AiiDA interesting for a new user base: non-specialists, such as experimentalists or researchers at companies, who would like to run well-defined turnkey solutions using an intuitive graphical user interface.

AiiDA lab. The AiiDA lab leverages state-of-the-art technologies (JupyterHub, Jupyter widgets and kubernetes) to provide AiiDA-powered "apps" that run in the web browser. After logging in to the platform, an AiiDA lab user has access to a personal Docker container through a Jupyter-based graphical user interface. The container comes preinstalled with AiiDA as well as a selection of apps for common tasks, such as connecting AiiDA to a remote compute resource or performing a geometry optimization or band structure calculation using the Quantum ESPRESSO [15] and CP2K [16] density functional theory (DFT) codes.

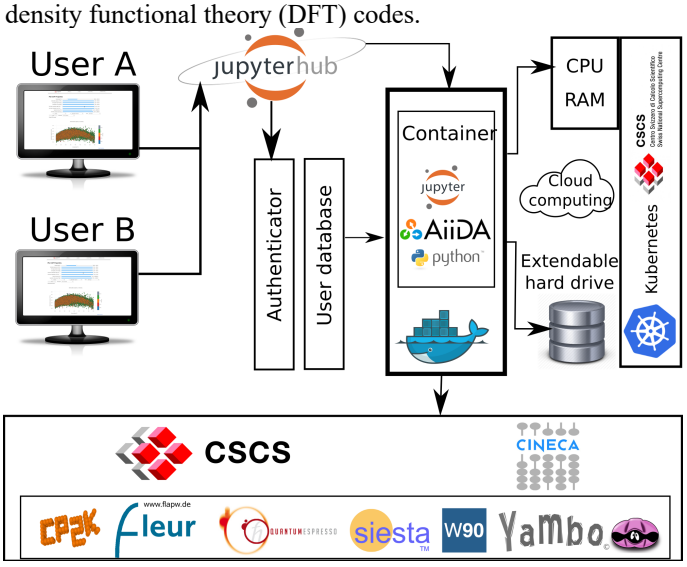

Fig. 3. AiiDA lab infrastructure. The AiiDA lab login page is provided by the JupyterHub that manages user authentication. Once user is logged in, JupyterHub will launch a Docker container and will expose access to Jupyter notebooks running inside the container. To balance the server load we deployed AiiDA lab on Kubernetes platform provided by CSCS. Every AiiDA lab container comes with AiiDA pre-installed and pre-configured.

AiiDA lab apps are nothing but Jupyter notebooks rendered in "app mode" [17]. Developers can therefore write powerful apps directly in python (no JavaScript required), minimizing the entry barrier for existing AiiDA users to writing such apps. A library of AiiDA-specific, reusable widgets further simplifies the task of creating

apps, making e.g. the upload of a structure just one line of code. One context, in which this model has already been taken up, are mixed experimental/theoretical groups, where it frees computational scientists from repetitive tasks by letting the experimentalists run the corresponding workflows themselves.

The AiiDA ecosystem is completed by the Materials Cloud Archive, a moderated research data repository for computational materials science registered on re3data [18], FAIRsharing [19] and recommended by Nature Scientific Data [20]. Besides welcoming relevant data from computational materials science in general, the Materials Cloud Archive accepts AiiDA databases. By uploading an AiiDA database, researchers provide access to the full provenance of their calculations, enabling peers to browse the database interactively, download individual files or the whole database, and start their research right from where the original author left off.

AiiDA is free and open source (MIT license), and deployment scripts for the AiiDA lab are scheduled to be released under the same license later this year.

Demo. Jupyter notebooks will be used to demonstrate how to solve a range of common tasks using the AiiDA python & command line interfaces (not shown). The demo will also include an AiiDA lab application that allows to perform electronic structure calculations with Quantum ESPRESSO [15] and CP2K [16]. Figure 4 provides a glimpse of the interface for preparing the inputs of a Quantum ESPRESSO calculation and displaying its results. The use of Jupyter notebooks enables a smooth transition from regular use to development, with Jupyter widgets providing interactive JavaScript components while programming in python.

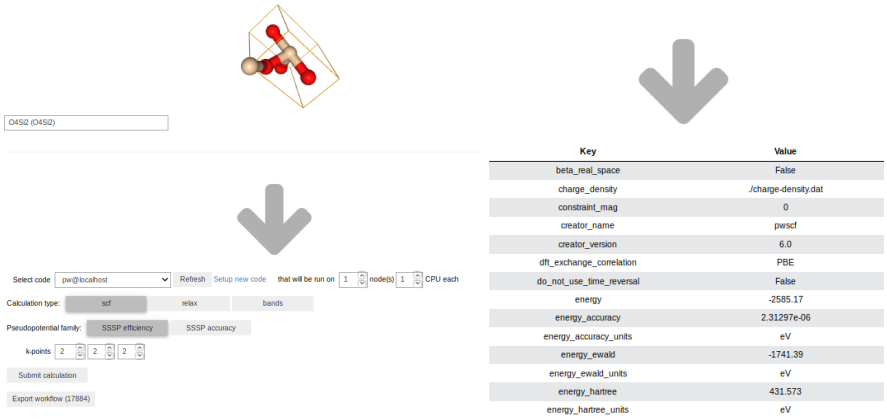

Fig. 4. Electronic structure application. After providing minimal inputs via a Jupyter notebook in AppMode, a dedicated AiiDA WorkChain generates the full inputs required by the DFT code and submits the calculation. From then on AiiDA takes over, managing the preparation of the input files, sending them to the supercomputer, waiting until the calculation is finished, retrieving the results back and parsing the output.

Since the calculation is managed by AiiDA all the data are stored in the AiiDA graph. To access it one can employ the QueryBuilder a tool that allows to query the AiiDA database. An example of a query is provided below.

```
In [1]: %aiida
        from aiida.orm import StructureData
        PwCalc = CalculationFactory('quantumespresso.pw')

In [2]: q = QueryBuilder()
        q.append(StructureData, tag='structure', project='*')
        q.append(PwCalc, with_incoming='structure', tag='calc')
        q.append(Dict, with_incoming='calc', filters={'attributes.total_force': {'<':1e-5}})
        q.all()

Out[2]: [[<StructureData: uuid: 62d73b1f-1d7b-414a-999c-5d4070487d40 (pk: 99)>],
         [<StructureData: uuid: 62d73b1f-1d7b-414a-999c-5d4070487d40 (pk: 99)>]]
```

Fig. 5. Querying via the AiiDA python API. This example query searches for atomic structures that were used as inputs to a Quantum ESPRESSO PwCalculation, filtering only those structures for which the total computed force has converged to less than 1e-5 eV/Å..

Acknowledgements. The AiiDA ecosystem is supported by the MARVEL National Centre of Competence in Research, funded by the Swiss National Science Foundation, the European Centre of Excellence MaX (grant no. 824143), the INTERSECT project (grant no. 814487), the swissuniversities P-5 “Materials Cloud” project (ID: 182-008), and the OSSCAR project of the EPFL Open Science Fund.
The Swiss National Supercomputing Centre is acknowledged for virtual hardware, storage and high-performance computing resources.

References

1. Concordat on Open Research Data. https://www.ukri.org/files/legacy/documents/concordatonopenresearchdata-pdf/
2. Research Data Alliance (2014) The Data Harvest Report – sharing data for knowledge, jobs and growth. https://rd-alliance.org/data-harvest-report-sharing-data-knowledge-jobs-and-growth.html. Accessed 28 Oct 2018
3. Ministerie van Onderwijs C en W (2016) Amsterdam Call for Action on Open Science - Report. https://www.government.nl/documents/reports/2016/04/04/amsterdam-call-for-action-on-open-science. Accessed 19 Aug 2019
4. Pizzi G, Cepellotti A, Sabatini R, et al (2016) AiiDA: automated interactive infrastructure and database for computational science. Comput Mater Sci 111:218–230. https://doi.org/10.1016/j.commatsci.2015.09.013
5. Curtarolo S, Setyawan W, Hart GLW, et al (2012) AFLOW: An automatic framework for high-throughput materials discovery. Comput Mater Sci 58:218–226. https://doi.org/10.1016/j.commatsci.2012.02.005
6. Mathew K, Montoya JH, Faghaninia A, et al (2017) Atomate: A high-level interface to generate, execute, and analyze computational materials science workflows. Comput Mater Sci 139:140–152. https://doi.org/10.1016/j.commatsci.2017.07.030
7. Mayeshiba T, Wu H, Angsten T, et al (2017) The MAterials Simulation Toolkit (MAST) for atomistic modeling of defects and diffusion. Comput Mater Sci

126:90–102. https://doi.org/10.1016/j.commatsci.2016.09.018
8. Saal JE, Kirklin S, Aykol M, et al (2013) Materials Design and Discovery with High-Throughput Density Functional Theory: The Open Quantum Materials Database (OQMD). JOM 65:1501–1509. https://doi.org/10.1007/s11837-013-0755-4
9. Cookie cutter recipe for AiiDA plugins. AiiDA team. https://github.com/aiidateam/aiida-plugin-cutter. Accessed 28 May 2019
10. AiiDA Team AiiDA registry of plugins. https://aiidateam.github.io/aiida-registry/. Accessed 28 May 2019
11. von Laszewski G, Hategan M, Kodeboyina D (2007) Java CoG Kit Workflow. In: Taylor IJ, Deelman E, Gannon DB, Shields M (eds) Workflows for e-Science: Scientific Workflows for Grids. Springer London, London, pp 340–356
12. Fahringer T, Prodan R, Rubing Duan, et al (2005) ASKALON: a Grid application development and computing environment. In: The 6th IEEE/ACM International Workshop on Grid Computing, 2005. pp 10 pp.-
13. Jain A, Ong SP, Chen W, et al (2015) FireWorks: a dynamic workflow system designed for high-throughput applications. Concurr Comput Pract Exp 27:5037–5059. https://doi.org/10.1002/cpe.3505
14. Amstutz P, Crusoe MR, Tijanić N, et al (2016) Common Workflow Language, v1.0
15. Giannozzi P, Baroni S, Bonini N, et al (2009) QUANTUM ESPRESSO: a modular and open-source software project for quantum simulations of materials. J Phys Condens Matter 21:395502. https://doi.org/10.1088/0953-8984/21/39/395502
16. Hutter J, Iannuzzi M, Schiffmann F, Vandevondele J (2014) Cp2k: Atomistic simulations of condensed matter systems. Wiley Interdiscip Rev Comput Mol Sci 4:15–25. https://doi.org/10.1002/wcms.1159
17. Schütt O. (2019) Appmode: a Jupyter extension that turns notebooks into web applications. https://github.com/oschuett/appmode. Accessed 28 May 2019
18. Re3data.Org (2018) Materials Cloud Archive. https://doi.org/10.17616/r3zj5w
19. FAIRsharing Team (2018) Materials Cloud
20. Recommended Data Repositories | Scientific Data. https://www.nature.com/sdata/policies/repositories. Accessed 29 Nov 2018