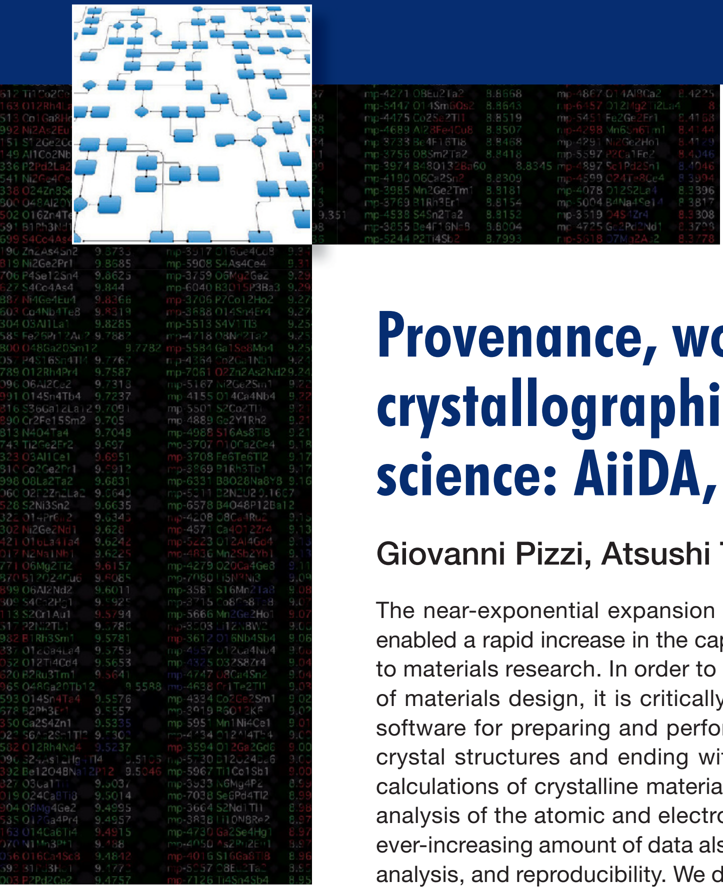


Provenance, workflows, and crystallographic tools in materials science: AiiDA, spglib, and seekpath

Giovanni Pizzi, Atsushi Togo, and Boris Kozinsky

The near-exponential expansion in computing resources over the last few decades has enabled a rapid increase in the capabilities of computational science, including applications to materials research. In order to harness the available resources and accelerate the field of materials design, it is critically important to develop robust and reusable automation software for preparing and performing multistep computational workflows, starting with crystal structures and ending with material properties. In the domain of first-principles calculations of crystalline materials, we highlight emerging tools for automated symmetry analysis of the atomic and electronic structure. With automation capabilities in hand, the ever-increasing amount of data also becomes a serious bottleneck in terms of organization, analysis, and reproducibility. We describe some of the progress and strategic challenges in the development of a general infrastructure for coupling computational automation with data management, emphasizing data reproducibility and provenance capture.

Introduction

The growth of available storage, memory, and central processing unit (CPU) speed has arguably outpaced advances in the development of algorithms capable of taking advantage of an entire supercomputer for single computations, at least in common electronic structure and atomistic simulations in computational materials science. As a result, the trend in the field is shifting toward exploiting supercomputers to run large numbers of simulations, each taking only a relatively small amount of time and resources (e.g., plane-wave-basis density functional theory [DFT] codes typically running for a few days on a hundred cores). It has become increasingly possible, using thousands of parallel individual calculations, to rapidly scan wide parameter spaces, such as atomic structure and composition. This "screening" approach allows for the examination of many materials variations by computation of their properties, selection of promising areas to explore with more accurate methods and experiments, and the ultimate discovery of new materials with optimal properties.[1] Similar trends, sometimes termed high-throughput computing, have emerged in diverse areas of computation and information science, and many parallels exist in challenges and solutions across disciplines.

The first ingredient required for large-scale screening efforts is automation, eliminating the time-consuming task of manually managing the life cycle of each calculation, from input generation and deployment to output retrieval. In the solid-state electronic-structure domain, various tools have appeared early on in the last decade to address the needs to automate, among others, DFT energy computations (e.g., AFLOW,[2] Materials Project,[3] OQMD,[4]) and crystal structure manipulation, such as the Python codes ASE[5]/pymatgen,[6] which perform many useful operations such as creating supercells, surfaces, systems with defects, and many more advanced features. Despite increasing capabilities, however, many available tools are not interoperable and mostly focus on either specific computational codes or a narrow set of computation types.

The second required ingredient is the ability to maintain data quality, accessibility, and reproducibility. These challenges call for the development of new software infrastructures to couple automatic materials computations to database storage solutions. Fortunately, the availability of mature tools and concepts in the areas of databases and automation brings tremendous opportunities to data-intensive computational investigations of materials.

Giovanni Pizzi, École Polytechnique Fédérale de Lausanne, Switzerland; giovanni.pizzi@epfl.ch
Atsushi Togo, Department of Materials Science and Engineering, Kyoto University, Japan; togo.atsushi@gmail.com
Boris Kozinsky, Harvard University, USA; bkoz@seas.harvard.edu
doi:10.1557/mrs.2018.203

Here, we first discuss the relevant aspects needed for an automation platform for computational materials science, focusing on AiiDA,[7,8] a Python-based platform implementing the ADES pillars[7] (automation, data provenance and reproducibility, research environment with powerful workflows, and sharing). We then focus on crystallography tools (in particular spglib[9] and seekpath[10,11]) that are essential within a high-throughput computation environment.

ADES concepts and implementation in AiiDA

Automation of reusable dynamic workflows

Given a specific computation type and a simulation code, the quickest automation approach is to implement simple scripts to prepare inputs and run the calculations. These may be a direct solution for projects involving only a small set of properties (e.g., phase stability or bandgaps) computed with one code and changing only a few input parameters. However, these scripts tend to be hardcoded for the immediate problem and the available computer and batch queue scheduler (i.e., the system accepting and queuing executions to be run as soon as resources are available on a computer cluster). As a result, they are often difficult to apply to other problems, computers, or codes without major modification, and tend to be used only by their creators. In turn, this leads to sparse documentation and testing, increased chances of bugs and errors, and more generally to duplication of effort, with similar functionality repeatedly reinvented. At the same time, computational discovery of technologically relevant materials often involves computation of multiple properties—functional performance, stability, and mechanical properties. Each of these may require a different method and code and varying amounts of computational time. In the solid-state atomistic domain, there are close to 40 different codes implementing various approximations and property computations, with a comparable number in the quantum-chemistry field.

In order to be able to combine the strengths and advantages of all of these codes, it is necessary to implement a platform for developing reusable and interoperable workflows, achieving a delicate balance between standardization and flexibility. Instead of developing increasingly complicated custom scripts, it is more efficient to adopt ideas, methods, and tools from the field of computer science. In the approach implemented in AiiDA, each task is an independent self-contained building block described in a uniform way as a calculation that takes data items (e.g., crystal structure, parameters) as input and produces data (e.g., electron or phonon spectra, energies) as output. Once each datatype is represented in a standard way that each workflow step is designed to accept, arbitrarily complex workflows involving diverse computational engines and data analysis tools can be composed by connecting the building blocks. The main advantage of using standard datatypes is the ability to reuse workflows as substeps within other workflows without modification.

Figure 1 illustrates an example sequence of steps and data types involved in thermoelectric materials discovery, starting from crystal structure and first-principles electronic-structure calculations.[1] Hierarchical in nature, a calculation step may be composed of lower-level operations, each also represented as a workflow step (e.g., converting data formats or writing input files). This example workflow, run for each material, involves the use of several codes (DFT total energy, phonons, electron–phonon coupling, Boltzmann transport) and a variety of datatypes. Nevertheless, both on the high-level scientific logic and on the low-level data management, each step has the same abstract representation—a calculation operating on data.

Driven by similar automation needs in different data-intensive disciplines, many workflow management systems have evolved over the last two decades. The vast majority of these require the entire workflow to be encoded as a predefined sequence of steps. However, the challenge in scientific computations, for properties of materials in particular, is the often unpredictable and dynamic nature of calculations, which depends on the application. On the algorithmic level, for instance, workflow "width" is only determined at runtime (e.g., for thermodynamic phase-diagram computations involving multiple competing phases). Likewise, workflow "depth" (e.g., in iterative convergence) is often not known *a priori.* In addition, many codes in the community had not been originally designed with automation in mind. Therefore, workflows often need to implement error-recovery features that must adapt to the actual output of intermediate calculations. It is thus essential to be able to easily construct fully dynamic workflows, where decisions on which steps to perform are made programmatically. Moreover, the workflow system must be

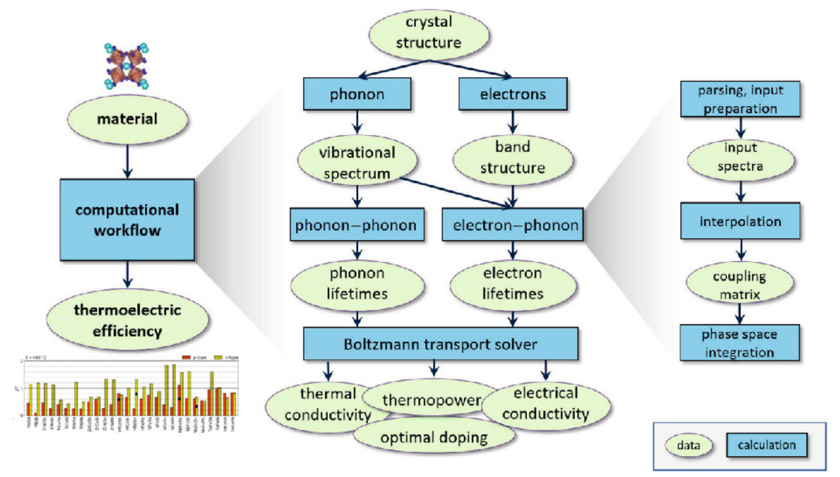


Figure 1. Example of workflow for computational discovery of thermoelectric materials used in Reference 1, shown at several resolution levels of control abstraction. Note: Green ovals, data objects; blue rectangles, calculations.

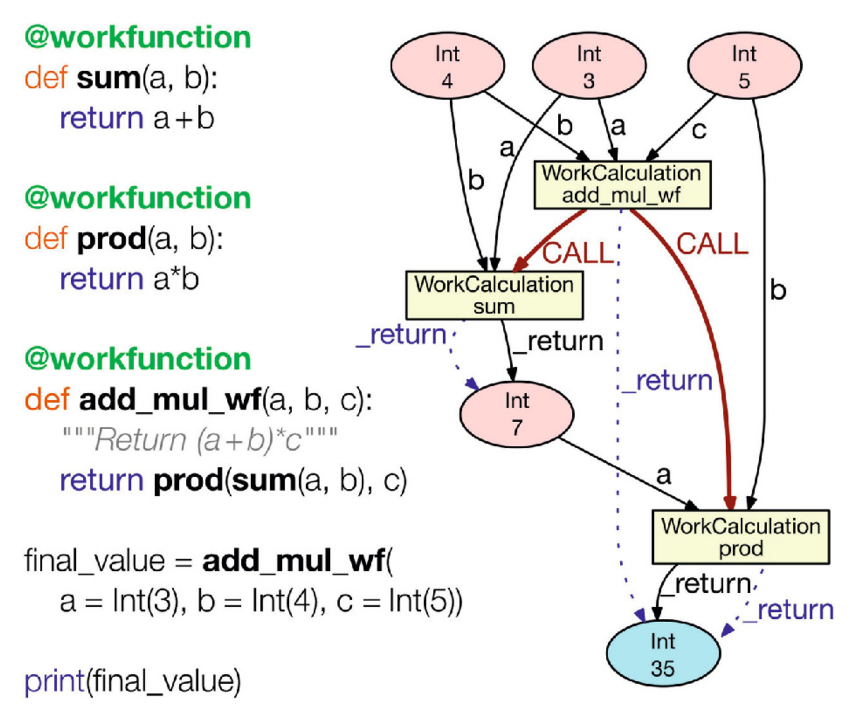
able to manage thousands of workflows, each of them potentially running for days or weeks (e.g., in molecular dynamics runs). For this reason, AiiDA implements workflows as subclasses of the WorkChain class, where steps must be defined as well as the logic flow (e.g., using "if" and "while" statements) that controls them. In each step, any Python logic can be executed, sub-WorkChains and long-running calculations can be launched, and at the end of each step, AiiDA waits for these to complete before continuing execution.

Importantly, the execution of WorkChains can be paused and restarted (and the computer on which AiiDA is running can even be rebooted) without losing the workflow state, which is essential when managing long-running workflows. An additional challenge of workflow systems, compared to scripts, is the learning curve for scientists to master new workflow constructs. Even for users familiar with Python, arguably today's leading open-source language for data science, composing workflows using object-oriented features can be an obstacle. Therefore, development of high-level, easy-to-use ways to standardize workflow implementation is a critical long-term strategic direction to facilitate automated materials design. To address this, AiiDA introduces a workflow architecture based on workfunctions, where workflows can be written as "wrapped" Python functions that accept and return immutable data objects (see **Figure 2**). The goal is to reduce the barrier of implementing workflows and track data provenance (see next section) by exposing the familiar functional interface already used in scripts, with minimal changes required in the code structure.

Data reproducibility and provenance

With the ability to automate computations and generate large data sets, a challenge arises in managing and organizing the data in a way that makes them accessible, reproducible, and searchable to other researchers. Simulations, unlike experiments, have the advantage of being, in principle, more reproducible, since codes and input data are digital and trivially replicated. Reproducibility, paramount in any scientific field, therefore needs a stronger emphasis in computational materials science. To this aim, it is necessary to record data provenance, including a detailed description of how the data were obtained, input parameters used, and with which method. Even if bit-level reproducibility is not necessary and often not achievable, approximate reproducibility and even partial provenance has tremendous importance for several reasons. A typical example is the common case where computational researchers store simulation results in custom folder structures with minimal documentation. This results in the data becoming difficult to understand and reuse by others, especially when the original author leaves the group. Another important use case is exemplified by verification and validation studies, where knowledge of the exact inputs and settings is essential for comparing results of different codes or numerical methods. In general, all data and results should be published (along with the scientific paper) with complete provenance. This is not only scientifically necessary for reproducing results, but it can accelerate materials discovery by making the data immediately available for use by other researchers. Fully documented reproducible computations do not need to be repeated and can be used to perform additional analysis to uncover correlations or compute different properties.

The two main obstacles for data provenance tracking are the additional effort required to record the full reproducibility information and the absence of immediate benefit of the repetitive and often error-prone manual annotation process, as individual citation metrics still do not directly reflect the additional effort of making published data reproducible. Our experience with multiple materials discovery efforts in diverse technology areas led to the realization that a convenient approach to record provenance is to couple calculation automation with on-the-fly metadata capture. Therefore, one of the main design goals of AiiDA is to enable workflow automation with automatic provenance tracking. Thus, researchers are no longer required to manually organize or curate inputs and outputs of their calculations and describe data relationships (e.g., the parameters



```
@workfunction
def sum(a, b):
    return a+b

@workfunction
def prod(a, b):
    return a*b

@workfunction
def add_mul_wf(a, b, c):
    """Return (a+b)*c"""
    return prod(sum(a, b), c)

final_value = add_mul_wf(
    a = Int(3), b = Int(4), c = Int(5))

print(final_value)
```

Figure 2. Automatic tracking of provenance using workfunctions in AiiDA. (Left) Simple workfunctions that compute the sum or the product of two numbers, *a* and *b*, or (function add_mul_wf) the quantity (*a* + *b*)*c*. (Right) Provenance graph automatically recorded by AiiDA via the workfunction decorators. The two types of _return links indicate if data were generated (solid arrow) or returned (dotted arrow) by the workfunction.

of the numerical approximations, the input crystal structure or the sequence of manipulating it from structures available in an existing database). Instead, data objects are automatically generated and stored on the fly in a database by the workflow engine as the computational workflow progresses, recording the entire data history and calculation sequence without user intervention and enabling provenance inspection and queries. Importantly, since the data recorded are the same as that used by computations, correctness and complete computational reproducibility are guaranteed. Besides storing raw input and output files, data and calculation objects are represented in AiiDA as nodes in a graph. Relationships between data and calculations are encoded by edges, or links, in the dataflow graph representing the provenance of operations (e.g., input links connect data to calculations that used them, and call links connect calculations to other subcalculations that they launched). As outputs can be used as inputs of further calculations, the AiiDA graph structure captures the full sequence of steps used to generate the final results. Moreover, every time the user invokes a "wrapped" work function (see Figure 2), AiiDA transparently tracks the operation, together with inputs, outputs, and any substeps called. Therefore, automated tracking of workflow execution is achieved with minimal cost for the user.

Sharing and reuse of data

A particular emphasis in AiiDA is on creating an ecosystem of tools, data, and workflows that encourages sharing and reuse of data and codes. Each database instance is local to a user or a group (and therefore fully private). AiiDA, however, ensures that data subsets, with their provenance, can be easily shared between instances and uploaded to centralized public databases. To facilitate sharing, additional user-defined metadata can be added to computed results *a posteriori* (after having run the simulations). This metadata, whose formats are often defined by domain-specific ontologies, enables standardization of properties facilitating the reuse of data even between different computational codes and domains, as well as the possibility to perform searches and queries on data generated with different tools. As standardized ontologies are still being developed[12–14] and are not fully stabilized yet, *a posteriori* metadata tagging is a future-proof solution to enable conversion of data to any format. Most importantly, if provenance is automatically tracked, *a posteriori* metadata tagging can also occur with no user intervention. This has been shown in Reference 15, where methods and plugin-based tools are presented to convert the AiiDA provenance to any external ontology. Moreover, plugins are already available to convert calculations performed using Quantum ESPRESSO[16] to the metadata format defined by the Theoretical Crystallography Open Database.[17]

Similarly, the data generated with AiiDA can be seamlessly exported and visualized in the Materials Cloud,[18] a web portal based on AiiDA that focuses on (curated and raw) data dissemination, while also providing automation tools for generating data both on the cloud or on local resources. An example of how the coupling of AiiDA with the Materials Cloud allows accessible and discoverable sharing of data is provided by the computational exfoliation study of 2D materials by Mounet et al.[19] The full data accompanying the scientific publication are available on the Materials Cloud Archive, is versioned with a DOI,[20] and is linked to a curated section that presents interactive views of the data provided in the paper (main text and supplementary material) linked to their browsable AiiDA provenance graph.

Crystallographic tools

Crystallography and materials science

The essential datatype in atomistic materials science is the atomic structure. Crystalline solids can be efficiently studied by considering their periodic structure, where distortions or defects can be included as corrections. Periodic crystals can be categorized according to their symmetries in one of 230 space groups,[21] and symmetries can facilitate the understanding of physical and chemical properties (selection rules for optical transitions, degeneracy of electronic states, occurrence of electric polarization). An essential ingredient of most workflows in the field is thus a tool to detect the space-group type and symmetry operations of an input structure.

It is often essential to ensure that atomic coordinates and cell vectors given as input to a quantum code are "refined" to numerically satisfy the crystal symmetries. This helps in guaranteeing that codes can exploit symmetries to reduce the computational cost or enforce them, facilitating the interpretation of results. Deviations from symmetry might be due to low precision in the reported coordinates in a database or from experiments, or can originate from the output of numerical minimization algorithms. To clarify the latter point, we consider the search of phase-transition pathways in metals,[22] where structures were relaxed after distorting them according to the phonon eigenvectors associated with imaginary frequencies. On-the-fly refinement was essential for a fully automated search, because distorted structures often relax toward a higher-symmetry configuration, but the minimization algorithm stops before the exact minimum is reached.

In addition, tools for standardization of crystal structures are convenient to enforce crystallographic conventions (principal crystal axes direction, axes order in orthorhombic structures) and facilitate comparison of tensorial properties computed for different materials or in different numerical settings.

Symmetry detection: spglib

A library that can perform the crystallographic operations previously described is spglib.[9] It detects the symmetry using the algorithms by Grosse-Kunstleve,[23,24] searching exhaustively all point-group and space-group operations, obtaining their matrix representations, identifying primitive and conventional cells, and performing database matching with the Hall symbols data set.[25,26] Additionally, for slightly distorted structures, lattice parameters are refined and reoriented in Cartesian coordinates, and atomic positions are relocated to their nearest site-symmetry positions (within numerical thresholds).[24] The algorithm of spglib is designed to be robust against these distortions, ensuring that the detected symmetry operations correspond to a space group as coset representatives. Moreover, the chosen conventional unit cell always adheres to crystallographic conventions: spglib follows an algorithm[27] that first fixes all basis vectors as required by the space-group type[21] and chooses the first setting appearing in the Hall symbols list.[25,26] The remaining vectors are ordered to be either |a| < |b|, |a| < |c|, or |b| < |c|, as prescribed by Parthé and Gelato.[28] In the special case of triclinic crystals, the Niggli cell[29] is employed.

High-symmetry k-points and band-structure paths: seekpath

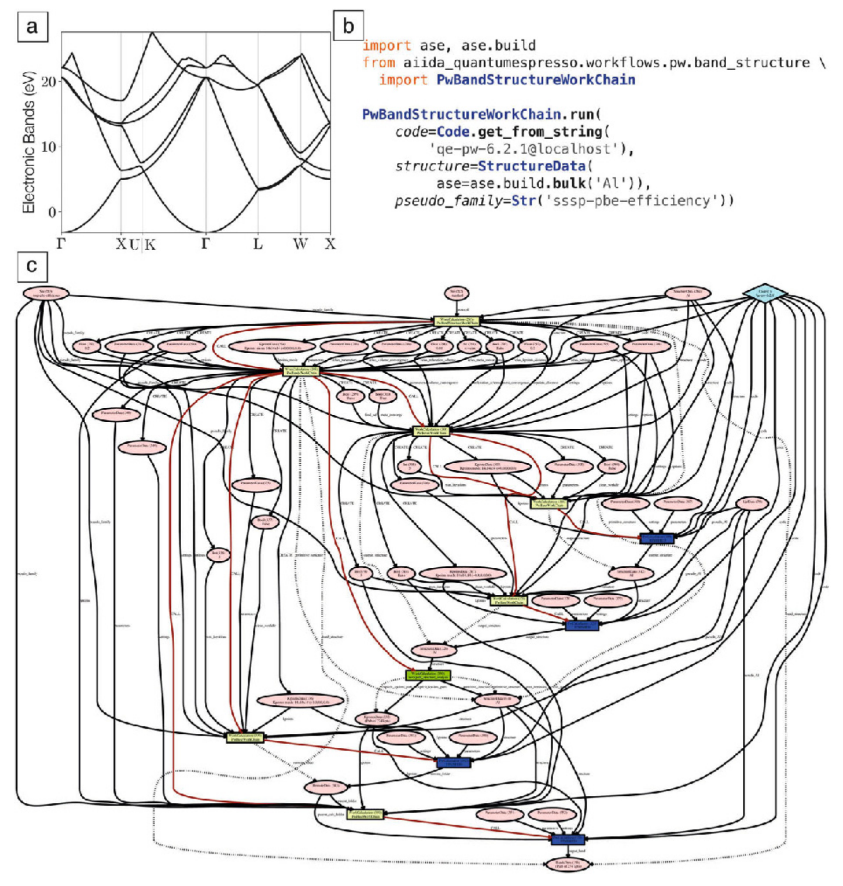
Band-structure plots are useful tools to study the electronic structure of materials, as they reflect the crystal symmetries and can provide both an intuitive and quantitative understanding (e.g., the Fermi-energy position relative to the bands indicates metallic or insulating character, or the band curvature is directly related to the effective electron mass). Typically, energy levels are plotted along paths (often chosen arbitrarily) connecting reciprocal-space high-symmetry points in the Brillouin Zone (BZ) (reciprocal-space points are often called "*k*-points"). High-symmetry paths and points have been classified and labeled by the crystallography community.[26,30] However, often there is also an interest in low-symmetry points at the vertices and face centers on the BZ surface, where energies of bands coming from neighboring BZs coincide by reciprocal-lattice translational symmetry.

In the high-throughput era, the need has been recognized[31] to provide coordinates of relevant *k*-path end points to automate band-structure computations. Moreover, using standard paths can ease comparisons. For this reason, seekpath[10] has been developed, also overcoming limitations of existing tools in the literature. seekpath relies on the standardization performed by spglib to guarantee that the crystal structure complies with crystallographic conventions. Moreover, labels of high-symmetry *k*-points are the same as in crystallography,[26] and labels for additional points do not conflict with existing letters. Additionally, seekpath provides default band paths, also for systems with no inversion and Hamiltonians without time-reversal symmetry, to cover all relevant lines in the BZ in a nonredundant way (note that the Bravais lattice might not be enough to determine this path, and the space-group symmetry must be taken into account[10]).

A Python implementation[32] and a web interface are both provided with seekpath. The latter[11] is valuable for educational purposes to visualize crystal structures and BZ using interactive three-dimensional plots, and to perform symmetry analysis without the need to install software. Conversely, the Python interface is ideal for fully automated computational projects.

Combining all tools

AiiDA implements interfaces to spglib and seekpath to streamline their use directly with AiiDA data nodes and automatically store their execution via work functions. These interfaces enable provenance tracking of computed band structures and are already implemented in the workflows for Quantum ESPRESSO.[33] Similarly, the combination of spglib with prototype AiiDA automation and database tools was key in enabling a systematic high-throughput study of the symmetry-controlled frustrated ionic transport and discovery of new solid electrolytes.[34] As an example, in **Figure 3**a, the aluminum band structure computed by the aforementioned



```
import ase, ase.build
from aiida_quantumespresso.workflows.pw.band_structure \
  import PwBandStructureWorkChain

PwBandStructureWorkChain.run(
    code=Code.get_from_string(
        'qe-pw-6.2.1@localhost'),
    structure=StructureData(
        ase=ase.build.bulk('Al')),
    pseudo_family=Str('sssp-pbe-efficiency'))
```

Figure 3. (a) Aluminum band structure computed with an automated AiiDA workflow. (b) Code to submit the "turnkey" workflow, requiring only minimal inputs: code, starting crystal structure, and set of pseudopotentials. (c) Provenance for the aluminum band structure data node (bottom node in the graph) as automatically tracked by AiiDA. Data nodes: red ovals, calculations: rectangles (light-green: WorkChains, dark-green: work functions, dark blue: Quantum ESPRESSO calculations), Quantum ESPRESSO code: light blue diamond. Starting from an initial structure (top of the graph), the workflow relaxes it, determines the suggested band path using spglib and seekpath (dark-green rectangle workfunction), and computes the band structure with Quantum ESPRESSO.

AiiDA workflows using Quantum ESPRESSO and exploiting spglib and seekpath was obtained by running the launching script of Figure 3b, where all tools described here were combined. The provenance graph, automatically tracked by AiiDA, is reported in Figure 3c. The input script is minimal and only requires specifying pseudopotential family, the code to use, and the initial structure that the workflow first relaxes. This "turnkey" workflow proves that the tools described enable full automation of calculations requiring many steps of different complexity and computational cost, while preserving the provenance of computed data. Leveraging these tools, the scientist's knowledge on how to perform the simulations and recover from possible errors is encoded in the workflows source code. Sharing the AiiDA workflows as plugins makes them easily reusable and ensures that complex calculations are fully reproducible.

Summary

We have presented the challenges of high-throughput simulations in computational materials science. First is the need for automated tools, such as AiiDA, to manage the execution of dynamic workflows while ensuring that these are implemented in a format reusable in different projects and by different researchers. We have emphasized the importance of guaranteeing reproducibility of results while tracking the provenance of all data, proving how AiiDA makes this possible without requiring additional effort by users. Furthermore, we have shown how, by leveraging automatic provenance tracking, it is possible to add metadata (in standardized formats) without additional user input to facilitate seamless sharing of computed data. We then discussed the tools (focusing on spglib and seekpath) to manage periodic crystal structures and their symmetries, essential in any atomistic materials science project, and how they can be integrated in dynamic workflows to automate the computation of advanced materials properties.

Acknowledgments

The authors would like to acknowledge the team of AiiDA developers,[35] as well as the contributors to AiiDA, spglib, and seekpath. G.P. acknowledges support from the MARVEL NCCR of the Swiss National Science Foundation and from the EU MaX Centre of Excellence (Grant 676598); A.T. from MEXT Japan through ESISM (Elements Strategy Initiative for Structural Materials) of Kyoto University; B.K. from Robert Bosch LLC.

References

1. G. Samsonidze, B. Kozinsky, *Adv. Energy Mater.* **8** (20), 1800246R1 (2018).
2. S. Curtarolo, W. Setyawan, G.L.W. Hart, M. Jahnatek, R.V. Chepulskii, R.H. Taylor, S. Wang, J. Xue, K. Yang, O. Levy, M.J. Mehl, H.T. Stokes, D.O. Demchenko, D. Morgan, *Comput. Mater. Sci.* **58**, 218 (2012).
3. A. Jain, S.P. Ong, G. Hautier, W. Chen, W.D. Richards, S. Dacek, S. Cholia, D. Gunter, D. Skinner, G. Ceder, K.A. Persson, *APL Mater.* **1**, 011002 (2013).
4. J.E. Saal, S. Kirklin, M. Aykol, B. Meredig, C. Wolverton, *JOM* **65**, 1501 (2013).
5. A.H. Larsen, J.J. Mortensen, J. Blomqvist, I.E. Castelli, R. Christensen, M. Dulak, 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.B. 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* **29**, 273002 (2017).
6. S.P. Ong, W.D. Richards, A. Jain, G. Hautier, M. Kocher, S. Cholia, D. Gunter, V.L. Chevrier, K.A. Persson, G. Ceder, *Comput. Mater. Sci.* **68**, 314 (2013).
7. G. Pizzi, A. Cepellotti, R. Sabatini, N. Marzari, B. Kozinsky, *Comput. Mater. Sci.* **111**, 218 (2016).
8. AiiDA, http://www.aiida.net.
9. spglib, https://atztogo.github.io/spglib.
10. Y. Hinuma, G. Pizzi, Y. Kumagai, F. Oba, I. Tanaka, *Comput. Mater. Sci.* **128**, 140 (2017).
11. *seekpath* Web Interface, http://www.materialscloud.org/tools/seekpath.
12. S. Gražulis, A. Merkys, A. Vaitkus, A. Le Bail, D. Chateigner, L. Vilčiauskas, S. Cottenier, T. Björkman, P. Murray-Rust, *Acta Crystallogr. A Found. Adv.* **70**, C1736 (2014).
13. Nomad Meta-Info, https://metainfo.nomad-coe.eu/nomadmetainfopublic/archive.html.
14. Open Databases Integration for Materials Design, http://www.optimade.org.
15. A. Merkys, N. Mounet, A. Cepellotti, N. Marzari, S. Gražulis, G. Pizzi, *J. Cheminform.* **9**, 56 (2017).
16. P. Giannozzi, O. Andreussi, T. Brumme, O. Bunau, M.B. Nardelli, M. Calandra, R. Car, C. Cavazzoni, D. Ceresoli, M. Cococcioni, N. Colonna, I. Carnimeo, A. Dal Corso, S. de Gironcoli, P. Delugas, R.A. DiStasio Jr., A. Ferretti, A. Floris, G. Fratesi, G. Fugallo, R. Gebauer, U. Gerstmann, F. Giustino, T. Gorni, J. Jia, M. Kawamura, H.-Y. Ko, A. Kokalj, E. Küçükbenli, M. Lazzeri, M. Marsili, N. Marzari, F. Mauri, N.L. Nguyen, H.-V. Nguyen, A. Otero-de-la-Roza, L. Paulatto, S. Poncé, D. Rocca, R. Sabatini, B. Santra, M. Schlipf, A.P. Seitsonen, A. Smogunov, I. Timrov, T. Thonhauser, P. Umari, N. Vast, X. Wu, S. Baroni, *J. Phys. Condens. Matter* **29**, 465901 (2017).
17. TCOD Database, http://www.crystallography.net/tcod.
18. Materials Cloud, https://www.materialscloud.org.
19. N. Mounet, M. Gibertini, Ph. Schwaller, D. Campi, A. Merkys, A. Marrazzo, Th. Sohier, I.E. Castelli, A. Cepellotti, G. Pizzi, N. Marzari, *Nat. Nanotechnol.* **13**, 246 (2018).
20. N. Mounet, M. Gibertini, Ph. Schwaller, D. Campi, A. Merkys, A. Marrazzo, Th. Sohier, I.E. Castelli, A. Cepellotti, G. Pizzi, N. Marzari, *Materials Cloud Archive* (2017), https://dx.doi.org/10.24435/materialscloud:2017.0008/v1.
21. M.I. Aroyo, Ed., *International Tables for Crystallography A Space-Group Symmetry*, 6th ed. (Wiley, 2016).
22. A. Togo, I. Tanaka, *Phys. Rev. B Condens. Matter* **87**, 184104 (2013).
23. R.W. Grosse-Kunstleve, P.D. Adams, *Acta Crystallogr. A* **55**, 383 (1999).
24. R.W. Grosse-Kunstleve, P.D. Adams, *Acta Crystallogr. A* **58**, 60 (2002).
25. S.R. Hall, *Acta Crystallogr. A* **37**, 517 (1981).
26. U. Shmueli, Ed., *International Tables for Crystallography B Reciprocal Space*, 3rd ed. (Springer, Dordrecht, The Netherlands, 2008).
27. Y. Hinuma, A. Togo, H. Hayashi, I. Tanaka, *Condens. Matter Mater. Sci.* (2015), https://arxiv.org/abs/1506.01455.
28. E. Parthé, L.M. Gelato, *Acta Crystallogr. A* **40**, 169 (1984).
29. R.W. Grosse-Kunstleve, N.K. Sauter, P.D. Adams, *Acta Crystallogr. A* **60**, 1 (2004).
30. Bilbao Crystallographic Server, http://www.cryst.ehu.es.
31. W. Setyawan, S. Curtarolo, *Comput. Mater. Sci.* **49**, 299 (2010).
32. seekpath, http://github.com/giovannipizzi/seekpath.
33. AiiDA Plugins for Quantum ESPRESSO, http://github.com/aiidateam/aiida-quantumespresso.
34. B. Kozinsky, S. Akhade, P. Hirel, A. Hashibon, C. Elsässer, P. Mehta, A. Logeat, U. Eisele, *Phys. Rev. Lett.* **116**, 055901 (2016).
35. AiiDA Team, http://www.aiida.net/team. □

Giovanni Pizzi is a research scientist at École Polytechnique Fédérale de Lausanne (EPFL), Switzerland. He obtained his PhD degree in physics from Scuola Normale Superiore di Pisa, Italy, and then completed postdoctoral research at EPFL. His current research interests include high-throughput computations and creation of material properties databases from *ab initio* simulations, ferroelectric properties of functional oxides, and methods and applications of Wannier functions. He leads the development teams of AiiDA and Materials Cloud and oversees their design. He is the author of *Seekpath* and the co-author of *Wannier90*. He is the recipient of the 2010 Giuseppe F. Bassani Prize of the Italian Physical Society. Pizzi can be reached by email at giovanni.pizzi@epfl.ch.

Atsushi Togo is currently an associate professor in the Department of Materials Science and Engineering, Kyoto University, Japan, working on the Elements Strategy Initiative for Structural Materials (ESISM) project. He obtained his PhD degree in engineering from Kyoto University. His current interests include physical phenomena on phonon–phonon and electron–phonon interactions. He has performed research and developed scientific software on these topics and on crystallography tools for more than 10 years. He is the author of the phonopy and spglib codes. Togo can be reached by email at togo.atsushi@gmail.com.

Boris Kozinsky has been an associate professor of computational materials science at Harvard University since 2018. He received his PhD degree in physics from the Massachusetts Institute of Technology. During the last 10 years as leader of the computational materials design team at Bosch Research, he initiated the development of database architectures and automated provenance tools for atomistic simulations and founded the AiiDA project. His research focuses on the development of computational methods and next-generation infrastructure for data-driven materials discovery, with application to batteries, ferroelectrics, thermoelectrics, and low-dimensional quantum materials. Kozinsky can be reached by email at bkoz@seas.harvard.edu.