ADVANCED ENERGY MATERIALS
www.advenergymat.de

Workflow Engineering in Materials Design within the BATTERY 2030+ Project

*Joerg Schaarschmidt, Jie Yuan, Timo Strunk, Ivan Kondov, Sebastiaan P. Huber, Giovanni Pizzi, Leonid Kahle, Felix T. Bölle, Ivano E. Castelli, Tejs Vegge, Felix Hanke, Tilmann Hickel, Jörg Neugebauer, Celso R. C. Rêgo, and Wolfgang Wenzel**

In recent years, modeling and simulation of materials have become indispensable to complement experiments in materials design. High-throughput simulations increasingly aid researchers in selecting the most promising materials for experimental studies or by providing insights inaccessible by experiment. However, this often requires multiple simulation tools to meet the modeling goal. As a result, methods and tools are needed to enable extensive-scale simulations with streamlined execution of all tasks within a complex simulation protocol, including the transfer and adaptation of data between calculations. These methods should allow rapid prototyping of new protocols and proper documentation of the process. Here an overview of the benefits and challenges of workflow engineering in virtual material design is presented. Furthermore, a selection of prominent scientific workflow frameworks used for the research in the BATTERY 2030+ project is presented. Their strengths and weaknesses as well as a selection of use cases in which workflow frameworks significantly contributed to the respective studies are discussed.

1. Introduction

Materials with tailored properties are an essential basis for the development of new technological solutions in the fields of energy and environment, health, information and communication, manufacturing, or security and transport, but their development and adaptation is often a time- and resource-intensive endeavor.[1,2] Experimental efforts to develop and design new materials are increasingly complemented by computational strategies. This mirrors the trend in many other application areas, where computer-aided design has significantly accelerated product development, often reducing the cost at the same time. Examples are the automotive, aerospace, and electronics industries, where the development of novel products are nowadays unthinkable without computer-aided design. A prerequisite for the successful application of such a strategy is the availability of predictive simulation protocols, which can be used as digital twins[3,4] for devices in the context of development and design.

Materials design is still behind other fields in the application of computer-aided design strategies, not for the lack of effort, but because of the complexity of the underlying task. The computational challenges for understanding the material properties encompass interdisciplinary research, where the comprehension of its nature runs through different scales of materials behavior, requiring multi-scale approaches. However, the field is lacking a monolithic computational framework to cover all of these scales, in both space and time, with the available computational resources.[5]

J. Schaarschmidt, J. Yuan, C. R. C. Rêgo, W. Wenzel
Karlsruhe Institute of Technology (KIT)
Institute of Nanotechnology KIT
76344 Eggenstein-Leopoldshafen, Germany
E-mail: wolfgang.wenzel@kit.edu

I. Kondov
Karlsruhe Institute of Technology (KIT)
Steinbuch Centre for Computing
76344 Eggenstein-Leopoldshafen, Germany

T. Strunk
Nanomatch GmbH
76185 Karlsruhe, Germany

The ORCID identification number(s) for the author(s) of this article can be found under https://doi.org/10.1002/aenm.202102638.

DOI: 10.1002/aenm.202102638

S. P. Huber, G. Pizzi
Theory and Simulation of Materials (THEOS) and National Centre for Computational Design and Discovery of Novel Materials (MARVEL)
École Polytechnique Fédérale de Lausanne
Lausanne CH-1015, Switzerland

L. Kahle
National Centre for Computational Design and Discovery of Novel Materials (MARVEL)
IBM Research Europe
Rüschlikon CH-8803, Switzerland

F. T. Bölle, I. E. Castelli, T. Vegge
Department of Energy Conversion and Storage
Technical University of Denmark
Kgs. Lyngby DK-2800, Denmark

F. Hanke
Dassault Systèmes BIOVIA
334 Science Park, Cambridge CB4 0WN, UK

T. Hickel, J. Neugebauer
Max-Planck-Institut für Eisenforschung
Department Computational Materials Design
40237 Düsseldorf, Germany

Workflow engines are able to address various challenges in computational materials design, which is a nontrivial field. For example, choosing the correct calculation parameters is rather challenging even for experts. Validated workflows can provide a predefined set of computational parameters that allow scientists to focus on the underlying science and less on the mechanics of setting up and running a calculation. The status quo of computational materials science excludes users without an in-depth practical knowledge of methods and computer hardware from running calculations. In consequence, many experts in associated fields are excluded from the benefits originating from theoretical insights. Validated computational workflows can act as a bridge enabling non experts to access information without the complication of selecting computing resources or numerical parameters.

Computational materials development requires several methods to generate a predictive materials model or a digital twin. There is an enormous literature on the development and application of the components, and their integration into complex protocols available.[6–9] Recently, there have been several high-level initiatives, such as the Materials Genome Initiative or the Materials Project,[10,11] the "Materials design at the Exascale" European Centre of Excellence (MaX, www.max-centre.eu), or the Harvard clean energy initiative[12] that aim at the computational development of novel materials. To date, only a few reports about the tools that may be used to formalize the execution of the underlying protocols, named here as workflows, are available in the literature. In addition to the actual computation, various other complex steps, including retrieval, preprocessing, transformation, analysis, and deposition of data, are integral parts of the process and need to be addressed in a scientific workflow. This complexity illustrates the need for a systematic review of available workflow frameworks. Here, we present the major frameworks we consider relevant for the BATTERY 2030 + projects scope with their respective strengths and weaknesses, together with a selection of use cases.

2. Scientific Workflows

Scientific workflows can be viewed as an approach that models computational tasks in simulation and data analysis to understand the physical nature of complex systems. A workflow represents the coordinated execution of repeatable computational steps while accounting for dependencies and concurrency of tasks. While, in computational science, the workflow approach has a long-established tradition,[13–15] it has only started to gain relevance in computational materials science, biology, chemistry, and physics within the last decade.[16–19] A workflow can formally be described by a directed acyclic graph in which the vertices denote the actions, and the edges indicate the execution order and data dependencies, known as control flow and data flow, respectively. A conceptual overview of the components of a workflow framework is given in **Figure 1**. Concrete workflow examples are presented below.

The design and study of digital twins does not necessarily require a workflow framework. However, there are important reasons for their utilization. By providing a layer of abstraction, workflows enable scientists to design and conduct studies without in-depth knowledge of the software deployment on computational resources. Furthermore, they improve the transfer of knowledge within groups, collaborators, or users by providing a concise description of the input data, utilized software and scripts, and the respective parameters and settings used for a specific project. Consequently, applying the same methodology to a new system or extending the scale of a study only requires a conceptual understanding of the subject in order to properly adjust input data and run parameters. Especially with increasing efforts for scientific data to adhere to the FAIR guidelines,[20] which require the data to be Findable, Accessible, Interoperable, and Reusable, workflow frameworks provide an essential tool to improve the interoperability and reusability of published data.

The main benefits of using workflows in multi-scale and high-throughput simulations can be summarized as follows:

1. Automation: Workflow engines schedule the computational tasks according to their interdependencies, collect the output of preceding steps, and pass the input to subsequent steps. Some workflow systems support the execution of process steps on different computing resources.
2. Complexity reduction: Prevalidated workflows allow experts and out-of-field users to generate production-quality results, thus optimally leveraging their respective training and focus on science and not on procedure.
3. Scalability and high performance computing (HPC) readiness: Workflows automatically exploit concurrency of steps that do not exchange data, that is, that are not directly connected in the workflow graph. High workflow concurrency can be used to scale up the application on an HPC cluster or a large number of distributed resources (for example, in the cloud).
4. Data reusability: Workflows reuse data seamlessly in subsequent steps. Thus, a sub-workflow can often be nested without modifications in a workflow for another application.
5. Provenance: Workflows are persistent objects providing metadata and methods to track the origin of data and code. A workflow can be automatically reproduced and thus be used for validation purposes. Therefore, a workflow documents a simulation or data analysis and consequently improves reproducibility.
6. Reliability and resilience: Workflows provide mechanisms to authenticate users on the resources, track errors, and recover from failure. Failures of single steps do not invalidate the whole workflow, but only the affected steps and their descendants.
7. Rapid prototyping: Workflows enable multiscale modeling by reusing existing codes in a very flexible "drag-and-drop" fashion using a specialized workflow editor and increase model development productivity.

To leverage these benefits, workflow frameworks need to store not only the relevant data but also capture and store the associated metadata that describe in detail how the data was generated. This includes, amongst others, an uncertainty quantification of the data generated at each step, which is of particular relevance in multi-scale workflows where error propagation can be a serious concern. Finally a deposition of the datasets in repositories satisfying FAIR data-sharing principles such as NOMAD (https://nomad-lab.eu/) or Materials cloud

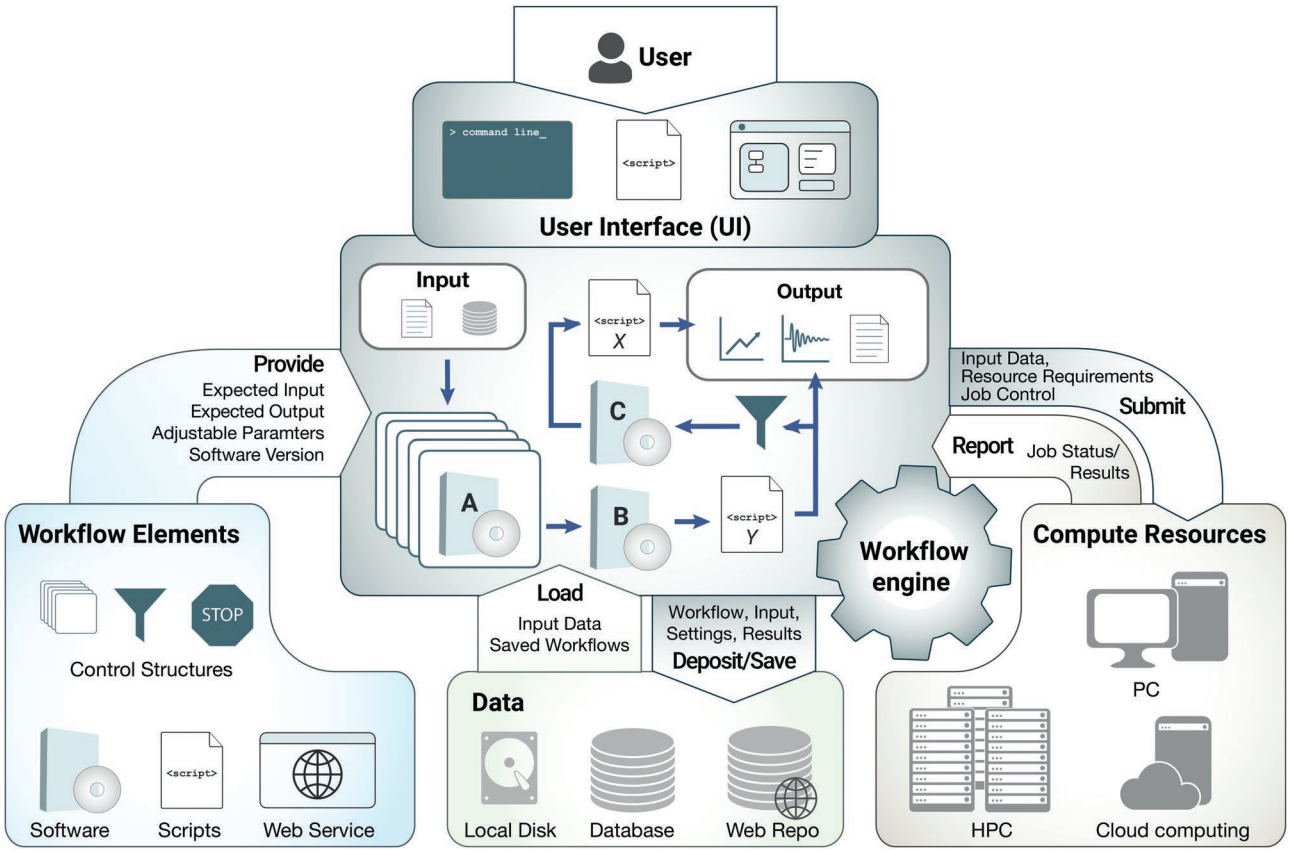

Figure 1. Schematic representation of the components of a workflow framework. A workflow consists of several interconnected data operations (center). Dependencies and data flow (indicated by arrows) between these operations are captured in the workflow and handled by the workflow framework. Furthermore, the workflow framework handles the interaction with various data sources and compute resources. The user interacts with the workflow framework via a user interface (UI), which enables the user to edit the workflow architecture, interact with data storage elements, target compute resources, as well as define and adjust parameters and settings of the individual workflow elements.

(https://www.materialscloud.org/), should be considered. This significantly improves the usefulness of the data for downstream applications and analysis as it provides a standardized format and well defined means of access. The same philosophy also applies to commercial enterprises, with the additional requirement that most internally generated research results will likely be confidential and not for public consumption. Ultimately, all data generated should be findable and accessible either in public or in closed corporate repositories following FAIR principles. This approach has the potential to enable the very efficient development of batteries and battery materials in a commercial setting.

3. Workflow Frameworks

Despite these significant advantages, many academic groups still rely on script-based approaches to implement increasingly complex computational protocols. This is partly due to the lack of information on existing workflow frameworks that have been specifically developed for application in the natural sciences. Another possible reason might be the effort needed to migrate all software from existing script-based code to a given workflow framework. Regardless of the reason, the amount of data and complexity related to a published simulation protocol increase every day. We expect that most of the groups working in materials modeling, and in particular those in the multiscale domain, will have to move to the workflows framework world. To guide this transition, we present an overview of a few representative workflow environments.

3.1. AiiDA

AiiDA is an open-source high-throughput workflow framework for computational science with a strong focus on reproducibility.[21,22] Workflows that are run by AiiDA are automatically stored in a provenance graph[19] with rich metadata, including all workflow inputs and outputs. The provenance graph is stored in a high-performance relational database which makes it possible to perform powerful queries on all data stored. Since all workflows and their inputs are stored, AiiDA can reuse calculations that have been already run with the same input parameters, or users can rerun completed workflows and compare the results for verification purposes. The framework is tightly integrated with job schedulers for high-performance computational resources and can submit jobs remotely over secure shell connections. This allows AiiDA users to easily

and efficiently distribute the workflow load over multiple computational resources.

AiiDA is domain agnostic and any code that can be run over the command line can be integrated through AiiDA's plugin system. An overview of existing public plugins is available on the plugin registry (https://aiidateam.github.io/aiida-registry/), with almost 100 different codes supported as of June 2021. Since AiiDA has its roots in computational materials science, a large number of plugins cover that domain, and in particular, many of the most popular density-functional theory (DFT) codes are interfaced to AiiDA. In addition, a recent collaboration[23] defined and implemented a common workflow interface for eleven of these plugins to automatically optimize the geometry of a crystalline or molecular structure, while at the same time automatically selecting all needed numerical parameters to ensure converged results: basis-set sizes, pseudopotentials, choice of algorithms, numerical thresholds (for energy, forces, stresses, ...). This makes it easy also for nonexperts to use any of these quantum engines with a single interface, providing as input only the atomic coordinates and no other numerical parameters.

To streamline access to simulation and workflow capabilities even more, also to users that are not familiar with Python, the AiiDAlab platform[24] leverages the Jupyter and JupyterHub technologies to provide an online graphical user interface (GUI) to AiiDA and existing workflows. In addition, a virtual machine called Quantum Mobile is available (https://quantum-mobile.readthedocs.io/en/latest/index.html). It comes preinstalled with AiiDA as well several simulation codes and their respective plugins. This makes it easy for new users to get started with AiiDA, in particular during schools and tutorials, but also to reproduce the results of published papers.[25] Finally, data from AiiDA databases can be shared with others by exporting all or parts of it to archive files, by directly exposing it through the integrated REST API, or by uploading it to the Materials Cloud web platform[26] (https://www.materialscloud.org/). Materials Cloud allows users to visually browse AiiDA provenance graphs, as well as explore many curated data sets (with full provenance metadata) via custom visualizations.

3.2. FireWorks

FireWorks (materialsproject.github.io/fireworks/)[18] is a Python-based generic workflow system that has been developed in the framework of the Materials Project[11] in the USA. FireWorks has been used extensively for high-throughput materials design, in particular via Atomate,[27] a specialized collection of FireWorks workflows covering the most common atomistic simulations in materials science. Workflows are composed of fireworks that appear as nodes in the workflow graph. Similar to AiiDA, rich provenance metadata is captured and persisted in a database, in this case MongoDB. FireWorks has two very powerful features: i) The FWAction object allows implementing dataflow when integrating Python functions as tasks, as well as dynamic workflows in which the workflow structure is modified during workflow execution. A workflow can be extended not only with single fireworks but also whole sub-workflows can be inserted or appended dynamically; ii) the DupeFinder object enables duplicates detection, that is, data from completed fireworks can be reused in other identical fireworks in the same or in other workflows to avoid repeated computations.

3.3. KNIME

The Konstanz Information Miner (KNIME - www.knime.com) is a java-based open-source modular environment, focused on the graphical assembly of a data pipeline and its interactive execution.[28] The data processing units in KNIME are referred to as nodes, and data is transferred between nodes in the form of class objects of a DataTable class, which includes meta-information about the represented data. KNIME offers an exhaustive spectrum of data analysis capabilities by integrating open source projects for statistics (R[29]), data mining and machine learning WEKA,[30] as well as data visualization JFreeChart.[31] Similar to AiiDA, external tools can be easily integrated using a plugin system that depends on subclasses of abstract classes for the node model, its dialog, and its view. A broad selection of KNIME nodes and workflows is publicly available at the online repository NodePit www.nodepit.com. Due to the GUI and broad distribution, KNIME workflows can be easily composed and reused without programming expertise. However Java programming is required if no nodes for the desired task exist.[32]

3.4. Pipeline Pilot

Pipeline Pilot (https://www.3ds.com/products-services/biovia/products/data-science/pipeline-pilot) is a chemically-aware commercial workflow engine developed and distributed by Dassault Systèmes BIOVIA.[32] It provides a graphical user interface in which a broad range of pre-defined components can be combined into protocols. Complete protocols can be incorporated as components into other protocols. The connections between components are referred to as pipes and represent the dataflow, which allows visual programming and rapid prototyping without requiring detailed knowledge of any specific programming language.

Pipeline Pilot components can represent simple actions such as loading, filtering, combination or manipulation of hierarchical data which can be in the form of molecules, reactions, materials, images, or standard data types. Components are grouped in collections, which handle specific topics. For example the Materials Studio Collection (MSC) provides components and protocols that utilize the functionality of BIOVIA Materials Studio, covering straight forward access to structure builders and symmetry functions, classical molecular dynamics (MD), mesoscale simulations, DFT calculations, as well as analysis components for all of these functions. Furthermore, MSC incorporates additional scientific functionality that builds on top of Materials Studio solvers to generate, for example, thermodynamic diagrams for metal alloys, protocols to determine glass transition temperature for polymers, and workflows to create crosslinked polymer structures. Available component collections include access to extensive functionality in cheminformatics, biomolecular modeling, and data science, as well as the automatic generation of static and interactive reports. Pipeline

Pilot also handles the high performance computing aspects of running simulations, including submission to queuing systems and parallelization over individual jobs as well as parallel execution of individual calculations where required.

3.5. SimStack

SimStack www.simstack.eu (https://simstack.readthedocs.io) is a graphical workflow editor based on Python. It allows the efficient implementation, adoption, and execution of complex and extensive simulation workflows. SimStack hides the complexity of high-performance computing on remote resources and enables users in academia or industry to incorporate competitive edge models and scalable scientific simulations into their virtual design process. Furthermore, it provides a highly flexible drag-and-drop environment that allows the quick adaptation of existing workflows to develop custom solutions fitting the user needs.

The Workflow elements are incorporated using Workflow Active Nodes (WaNos)—simple XML files defining the expected input and output and adjustable parameters. The WaNos act as a wrapper to call the respective program code and parameters are incorporated using a simple templating language. Thereby, incorporation of any arbitrary software into SimStack only requires knowledge of the XML syntax and elements and of the templating language. The end-user can then use the element by providing the required input and setting the parameters in a graphical user interface (GUI).

SimStack splits the whole virtual design process into client and server modes. The client, executed on the laptop, is dedicated to modeling workflows. In this framework, several modules are connected into complex workflows using drag and drop features. The most relevant parameters are set using the automatically generated GUI. The SimStack client automatically connects to the SimStack server installed on computational resources and handles the execution of the simulations, managing file transfer, submission, monitoring of workflows, and downloads the results files to the client user.

3.6. Pyiron

Pyiron is an integrated development environment (IDE) for computational materials science www.pyiron.org. While this workflow framework also has the goal to connect different tools in a single platform, it is particularly designed to interactively develop simulation workflows and upscale them for high-throughput simulations on available computing resources. Therefore, the pyiron IDE employs Jupyter notebooks as web-based source code editors, combining the advantages of a flexible programming environment with documentation, visualization and auto completion. The environment is further connected with a job queue system for building automation and a hierarchical data management solution. The elementary units of this interactive environment are pyiron objects based on an abstract class, which links application structures such as atomistic structures, projects, jobs, simulation workflows, and computing resources with persistent storage.

In general, the whole process in any simulation problem consists of model input, output generation (running simulation), and output analysis. However, between model input and output analysis, there are several issues to manage. All these issues are addressed by the pyiron framework, which orchestrates them in twelve generic steps, termed as 1) model, 2) project, 3) generic input, 4) code input, 5) simulation, 6) code output, 7) generic output, 8) job validation, 9) collect data, 10) analysis, 11) visualization, and 12) validation. Out of all these steps, only three of them (1, 2, 12) require mandatory inputs from the user, while the rest is automatically managed by the pyiron IDE.[33]

The use of the features of the Jupyter notebooks platform (www.github.com/jupyterhub/jupyterhub) has the additional advantage that it provides a powerful server–client system, where the user works with a default browser on the local compute resource to develop, run and analyze workflows that run on remote HPC resources, including binder instances for cloud computing.

3.7. MyQueue

MyQueue[34] is a task and workflow scheduling system which acts as a front-end for schedulers (SLURM/PBS/LSF). The main scope/function is to facilitate the submission of several thousands of tasks submitted to a cluster. One of the main advantages is the low entry barrier and its simplicity: Existing Python scripts containing the simulation steps can be composed into a workflow by setting up the dependencies using MyQueue. Each script is then submitted as a job to the scheduler and monitored by MyQueue. In principle, MyQueue can work with any simulation code and has recently been used together with the atomistic simulation environment (ASE).[35] ASE has been developed for supporting computational scientists in running, visualizing, and analyzing atomistic simulations through providing Python tools and modules. By supporting 44 simulation codes (version 3.21.1, June 2021), such as GPAW,[36] VASP,[37] Quantum ESPRESSO,[38] or FHI-Aims,[39] ASE can be easily used to bridge results from different simulation engines as well as from different time and length scales. At the core is the calculator object, providing a unified interface for the supported simulation packages. In essence, researchers can build complex simulation protocols consisting of generating the input script, managing and performing the calculation and postprocessing the results in a few lines of code. MyQueue does not need any database server to run the tools, while ASE supports different database back-ends if needed. For instance, this allows researchers to perform rapid prototyping of new workflows for which the exact dependency graph is not known upfront, while it can also be used for large scale high-throughput studies.

The ASE/MyQueue framework was first applied to create the computational 2D materials database (C2DB).[40] Although ASE and MyQueue do not incorporate the handling of data provenance or the robustness of simulation tasks, the developers have recently implemented an additional package called the atomic simulation recipes (ASR).[41] These recipes include data provenance and have the main advantage of dividing a complex workflow into simple tasks that can run both as a single calculation or together to form a full workflow.

3.8. Workflow Frameworks Summary

Table 1 summarizes the main differences among the workflow frameworks. This table presents some aspects, such as interface, workflow language, license, and required computational expertise levels of users to execute workflows. We believe that these features and advantages may guide researchers in choosing a specific framework for targeting a particular scientific community.

4. Use Cases

In the following we illustrate the application of workflow engines with some examples.

4.1. High-Throughput Screening for Solid-State Li-Ion Conductors

Introducing solid-state Li-ion conductors as Li-ion battery electrolytes has the potential to greatly improve battery safety and performance.[42] Besides mechanical and electrochemical stability, battery electrolytes must be electronically insulating but highly conducting for Li ions. The large number of potential candidate materials motivates an automated high-throughput computational screening. Kahle et al.[43] used the AiiDA framework for such a screening, starting from 1362 unique experimentally known Li-containing crystal structures with acceptable elemental composition. A first workflow was used to eliminate crystal structures that are electronic conductors at the PBE-DFT level. At the second stage, 971 crystal structures were successfully relaxed at the PBE-DFT level via AiiDA workflows.[22,44] At the third stage, a custom charge-density based force field[45] developed for molecular-dynamics simulations of Li-ion diffusion in solids was fitted for every material, requiring tens to hundreds of single-point SCF calculations per candidate. Molecular dynamics simulations were performed in a highly parallelized manner for 796 crystal structures, requiring an iterative procedure of restarts and checks for convergence of the dynamical property of interest (the diffusion coefficient of Li ions). For 132 highly diffusive materials, extensive Born–Oppenheimer first-principles molecular dynamics (FPMD) simulations were performed. In total, 2503 SCF calculations, 5214 variable-cell relaxations, 171 370 classical molecular dynamics simulations, and 11 525 FPMD simulations were performed on four different clusters, all managed via AiiDA workflows storing the provenance of every result. We show the directed acyclic graph for one candidate structure in **Figure 2**. The resulting calculations were uploaded to the Materials Cloud Archive,[46] and several Li-ion conductors studied or discovered in the computational screening were subsequently analyzed further in experiments.[47,48]

4.2. Multiscale Modeling of Organic Semiconductors

SimStack rapid prototyping was used mainly in the domain of multi-scale materials modeling, including the development of novel materials and elucidation of their characteristics.[49] Based on a widely used electron conducting material, Alq_3, novel semiconductors were designed and tested virtually for tailored electronic properties as illustrated in **Figure 3**.

The prediction process covers multiple scales from the atomic to the meso-scale:

1. The morphology structure is based on molecule-specific force fields parametrized by DFT calculations. A forcefield is generated containing partial charges and dihedral energy profiles.
2. A thin film of the parametrized material is deposited on a substrate in a Monte-Carlo simulated annealing approach.[50]
3. Frontier orbital energy levels are calculated in the environment by a self-consistent DFT-based approach[51,52]
4. Based on a macroscopic expansion of the morphology, transport properties are calculated using either a generalized effective medium model[53] for single layers or a KMC solution.

Most of these steps generated large amounts of data in their foreign data types, which had to be transferred, managed, and converted by the module's respective expert, making the individual execution of each module slow and unwieldy. The necessary complex modeling solutions were rendered into an easy-to-use, market-ready workflow for SimStack. Thus a novel organic semiconductor with a predicted three orders of magnitude improvement in electron mobility could be designed by systematically screening potential candidates, and the prediction was subsequently experimentally confirmed.[52]

4.3. High-Throughput Screening of ORR and OER Electrocatalysts

FireWorks workflows[18] have been employed to model electrocatalysts for the oxygen reduction reaction (ORR) in alkaline fuel cells[54–56] and the oxygen evolution reaction (OER) in alkaline electrolyzers for water splitting.[56,57] As a descriptor of the thermodynamic efficiency of the electrocatalyst, the

Table 1. Interface: Jupyter (Jupyter notebook), API (application programming interface), and GUI (Graphical user interface). Workflow Language: languages used to build the workflows. License: types of software licenses, (Comm.: commercial). Expected user level for execution: expected level of computational expertise to execute workflows for typical users addressed by the frameworks (Adv.: advanced, Non-exp.: non-expert.).

Features and advantages	AiiDA	FireWorks	KNIME	Pipeline Pilot	SimStack	PyIron	MyQueue
Interface	API Jupyter	API	GUI API	GUI API	GUI	Jupyter	API
Workflow anguage	Python	Fireworks schema	GUI Java	GUI, Pilotscript, Perl Java, Python, Jupyter	GUI	Python	Python
License	MIT	BSD-like	GPL	Comm.	Comm.	BSD	GPL V3
Expected user level for execution	Adv.	Adv.	Medium	Non-exp.	Non-exp.	Adv.	Adv.

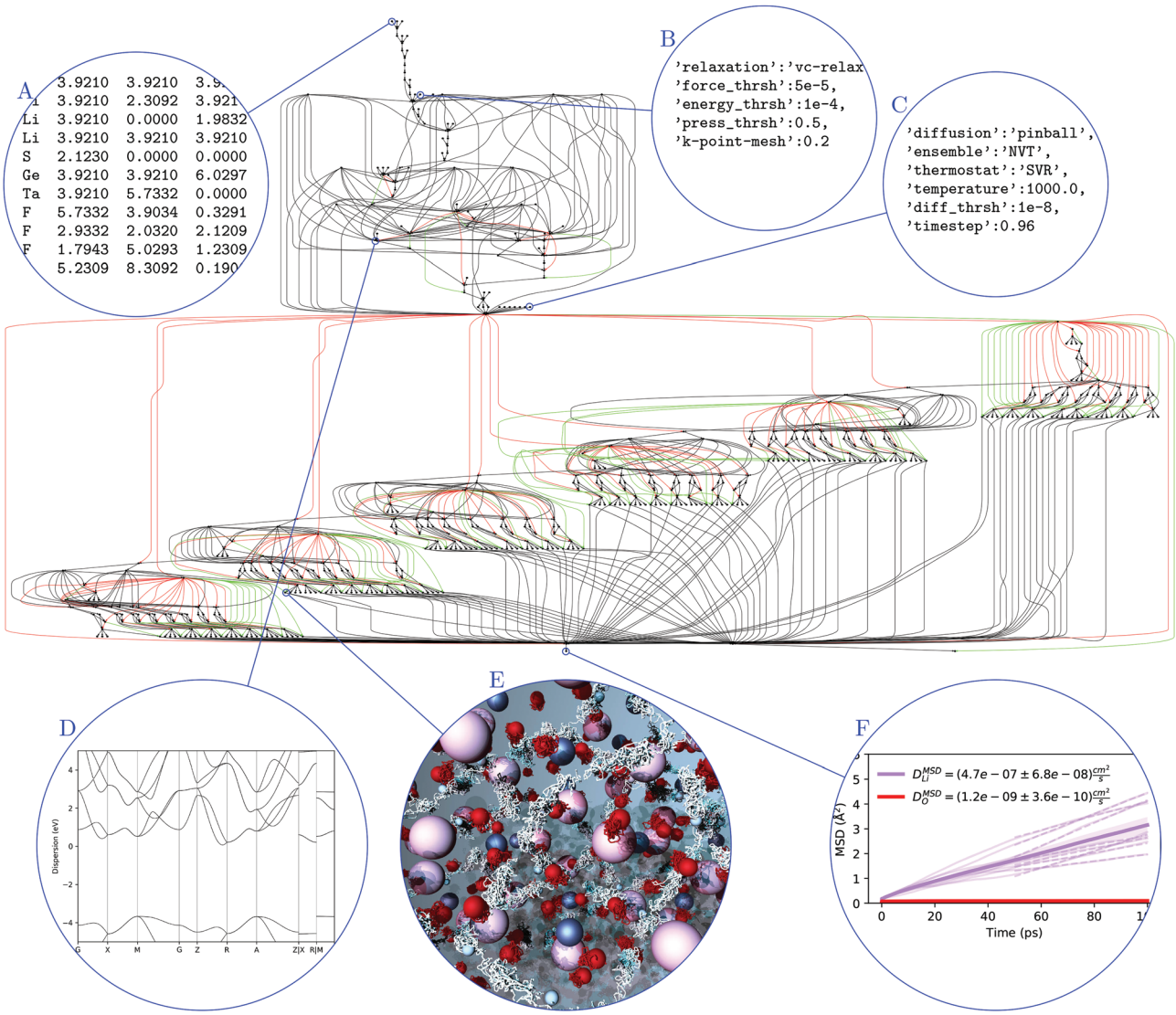

Figure 2. Schema of the screening workflow for Li-ion conductors as a sequence of calculations, resulting in a directed acyclic graph in the AiiDA database. We use black lines to draw the edges corresponding to inputs/outputs of a calculation; workflows calling other workflows or calculations (cf., Figure 1) are given by red lines, whereas workflows returning data are denoted by green lines. The workflow receives as input A) a structure, given by the *x*-, *y*-, and *z*-coordinates of atoms, and B) parameters for the calculations (B, C). Intermediate results such as D) the electronic band structure and E) molecular dynamics trajectories, as well as the final result, F) the diffusion of each species, are stored with their provenance.

critical potential has been computed and directly compared to experimentally measured ORR and OER onset potentials. The ORR (OER) critical potential U_{max} (U_{min}) is the thermodynamic upper (lower) bound of the electrode potential for which all ORR (OER) reactions are spontaneous. Thus U_{max} (U_{min}) defines a thermodynamic lower bound of the electrochemical overpotential that in turn determines the efficiency and the activity of the catalyst. In order to compute the critical potentials, the free energies of all species involved in the ORR/OER catalytic cycles have been computed using DFT. Therefore, the calculation for one specific case (active surface, active site, presence and position of dopants, presence of solvent molecules, etc.) requires several DFT calculations with inter-dependencies. A virtual screening of a large number of candidate structures requires running a workflow for every specific case.

The workflow, shown in **Figure 4**, includes a repeating pattern (a sub-workflow) which is executed for each species. Identical workflow steps and whole sub-workflows are automatically identified by FireWorks and reused in other workflows without the need to recompute the same results. This particularly occurs for the gaseous species and for structures that are common for two or more structural models or reaction pathways. The integration of atomic structures and of the VASP code, used to perform the DFT calculations, into the workflow has been implemented in Python using the ASE[35] and the PyTask class in FireWorks. The output of every sub-workflow is the free energy of the input atomic structure. In a recent work,[56] the workflow used previously[54,55] has been extended with a step for automated determination of the magnetic state of the species. Then the structure is relaxed within the selected

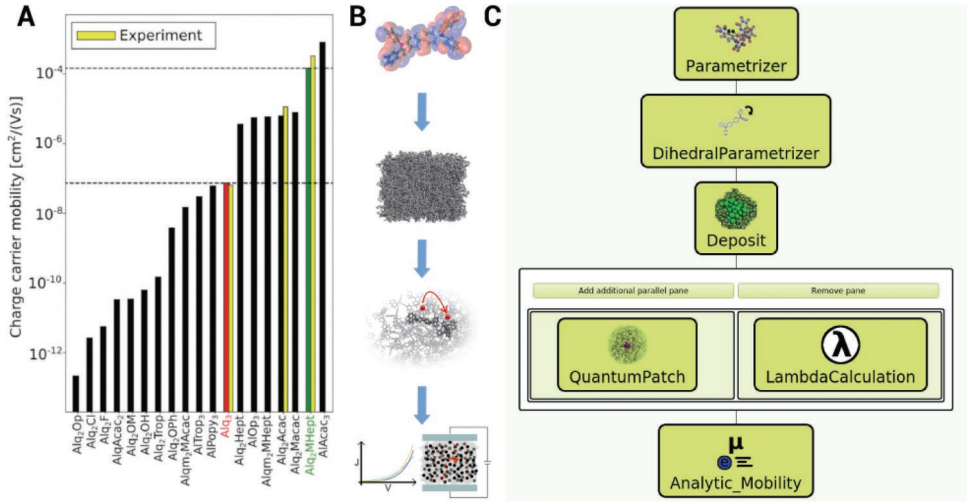

Figure 3. Screening of the charge carrier mobility of derivatives of the Alq_3 material. A) Using a multi-scale workflow a novel material (green) could be developed exhibiting a two orders of magnitude higher mobility than the original material (red), reproduced with permission.[52] Copyright 2021, John Wiley and Sons. B) The workflow consists of a parametrization stage based on single molecule DFT. A forcefield is generated, which is then used to generate thin-film morphology using simulated PVD.[50] For each pair in this morphology the electronic structure is relaxed in a self-consistent way.[51] Finally the mobility is calculated using a generalized effective medium model.[53] C) Implementation of the workflow in the SimStack workflow application. The complexity of the multiscale problem is hidden from the user by automatically interfacing the required applications for each scale. After the initial workflow assembly, the only significant remaining input is the structure of the initial molecule.

magnetic state and used in a harmonic normal-mode analysis to confirm the energy minima and to calculate the zero-point energy and entropy contributions to the free energies. Finally, using the thus computed free energies of all species, the electrochemical potentials U_{max} and U_{min} are calculated. The primary benefit of using FireWorks for this study has been the automation of computing the activity descriptors for a large set of surface structures taking advantage of the concurrency of the most time-consuming steps, as depicted in Figure 4.

4.4. Automated Calculation of Electrolyte Transport Properties

The precise composition of a battery electrolyte is an essential contributor that governs the long-term behavior of the battery cell, specifically with respect to degradation. At the same time, all individual components and their overall combination will influence the battery performance via their effect on the electrolyte transport properties. The goal of this particular workflow is to obtain these transport properties from the exact chemical formulation of an electrolyte candidate. This example is implemented using BIOVIA Pipeline Pilot[58] and is largely based on functionality also found in BIOVIA Materials Studio.[59]

We briefly outline the method, corresponding to the visualization in **Figure 5**a. We begin with a list of molecules that make up the electrolyte, along with the respective amounts that are going to be used in the simulation cell. This list is used as the input for an amorphous cell calculation,[60] which provides an energetically favorable approximation for the liquid electrolyte and which serves as the input for molecular dynamics simulations using the COMPASS force fields specifically designed for organic liquids and solids in the condensed phase.[61] The MD simulation is done in two stages with the initialization phase using variable cell dynamics to establish the density for

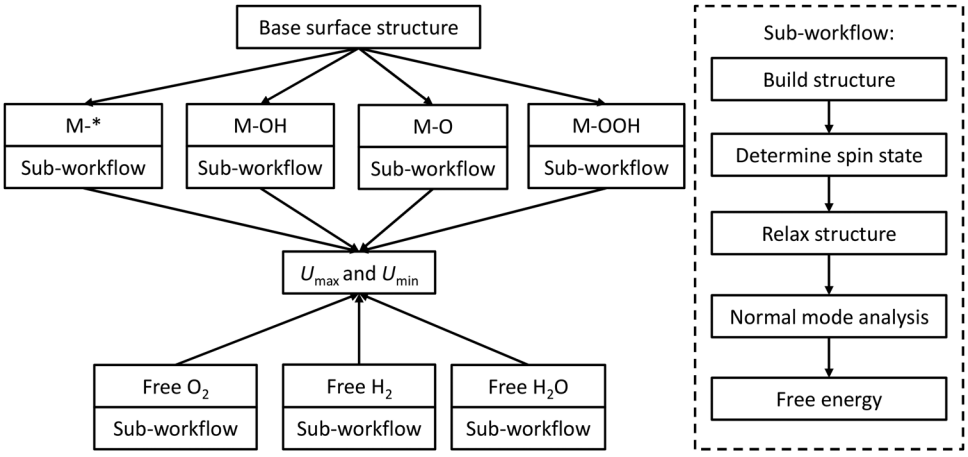

Figure 4. Schematic representation of a high-throughput screening workflow for ORR/OER electrocatalysts.

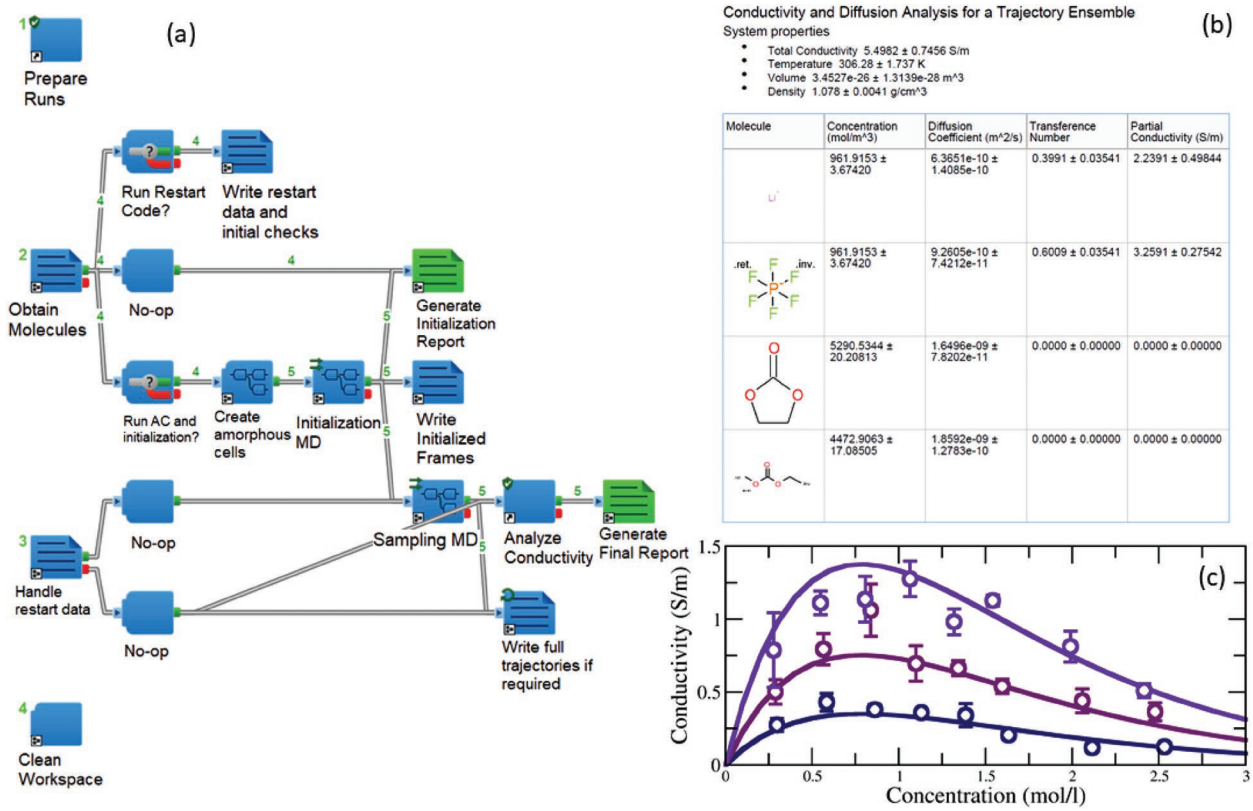

Figure 5. Workflow execution and reporting using Pipeline Pilot. a) visual form of the executed workflow in ref. [62] describing how to calculate the transport properties of a battery electrolyte. The green numbers denote the number of records passed, either four molecules or five samples b) report summary for a single protocol run describing the calculated properties and their statistical uncertainties for a single Li salt concentration and at a single temperature. c) Data from multiple runs for different temperatures and salt concentrations.

this particular formulation and the sampling phase providing the trajectories from which transport properties are calculated using standard analysis functions from Materials Studio and the Materials Studio Collection. These calculations are repeated automatically for a specified number of independent samples. Moreover, the workflow allows restarts to either add more samples or to extend existing trajectories when the results are not sufficiently well converged. A report for each of the runs returns information on the overall statistics and each individual molecular dynamics calculation, see Figure 5b.

Figure 5c shows the conductivity summary based on different runs at different temperatures and lithium salt concentrations, which can be used to fit the overall conductivity as a function of these two variables. This information, along with similar results for the Li diffusion coefficient and transference number, can be used to quantitatively reproduce measured discharge curves for battery cells.[62]

4.5. Automated Discovery of Materials for Intercalation Electrodes

Examples in materials science using ASE and MyQueue for fully automated and reproducible workflows include applications of solely ASE,[63,64] or the combination of ASE and MyQueue.[40,41,65,66]

With respect to battery materials, an automated workflow for calculating crucial ion-insertion battery properties in the framework of DFT has been established using ASE and MyQueue.[65] In detail, the stability is estimated through volume changes and the convex hull energy, open-circuit voltages (OCVs) are predicted using vacancy defect calculations and finally the kinetics are estimated through calculating migration barriers employing the nudged elastic band (NEB) method (**Figure 6**). The estimation of the migration barrier is further accelerated through exploiting reflection symmetries if present in the path (step 7a/b in Figure 6).[67] Automating the calculation of kinetic barriers using DFT+NEB is beneficial due the calculations being computationally expensive, prone to convergence problems as well as time-consuming to set up manually. One of the main achievements of the workflow, are the insights on automating calculations across different chemical structures through making them robust against possible failures. Due to the large amount of data generated using consistent parameters, a subsequent study explored data-driven methods to more efficiently guide the search toward new cathode materials.[68]

4.6. Automated Analysis of Interatomic Potentials Close to the Melting Point

The thermal tolerance of Li-ion batteries is a topic of major concern for many applications, including the performance at extreme temperatures[69] and the exothermic reactions known as

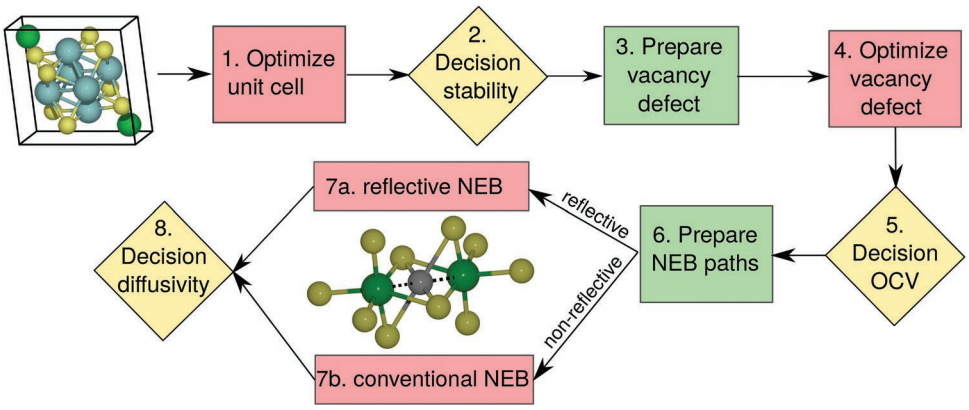

Figure 6. Simplified schematic representation of an ASE+MyQueue workflow for calculating properties of ion insertion materials. Each step represents a task in the workflow separating between optimization (red), preparation (green—structure generation and symmetry analysis) and decision tasks (yellow). The two structures show an example input structure (top left) as well as a relaxed NEB path structure generated by the workflow (bottom). The initial and final state (green spheres) are connected by the black dashed line and the transition state is visualized as a grey sphere.

"thermal runaway." Simulation workflows for high-temperature conditions typically rely on the availability of interatomic potentials, since these calculations are often too expensive for DFT. However, the performance of these potentials at temperatures that have not been part of the fitting strategy is often unclear. We discuss in the following a pyiron workflow for the automatic determination of the melting point of potentials, since these values can be easily bench-marked against experiments.

The workflow makes in five major steps use of the coexistence method,[71,72] in which the melting temperature is defined as the equilibrium of the solid and the liquid phase (upper part of **Figure 7**). This approach has conceptional advantages as compared to other criteria for melting, such as those of Lindemann[73] or Born.[74] The challenge for an automation is, however, that usually the expertise of the experienced scientist is required to supervise the MD simulation. An IDE as provided by pyiron[33] allows the user to visualize, analyze, modify, and test the individual steps of the workflow, until they are ready for automation.

The need for interactive development strategy already starts with setting up the interface structure (Step 2 in Figure 7). Here, overlapping atoms and void formation have to be avoided and a proper relaxation has to be ensured. Heating up only the liquid part of the supercell with selective dynamics turned out to be the method of choice. Another typical challenge is the formation of voids in the liquid that arise as an artifact for certain strain values (Step 4 in Figure 7). While such artifacts can be easily recognized by human inspection, for the automation a detection scheme based on Voronoi volumes had to be developed. A third example is the identification of atomic configurations to distinguish solid from liquid phases (first if statement in Figure 7). While this is usually done with a common

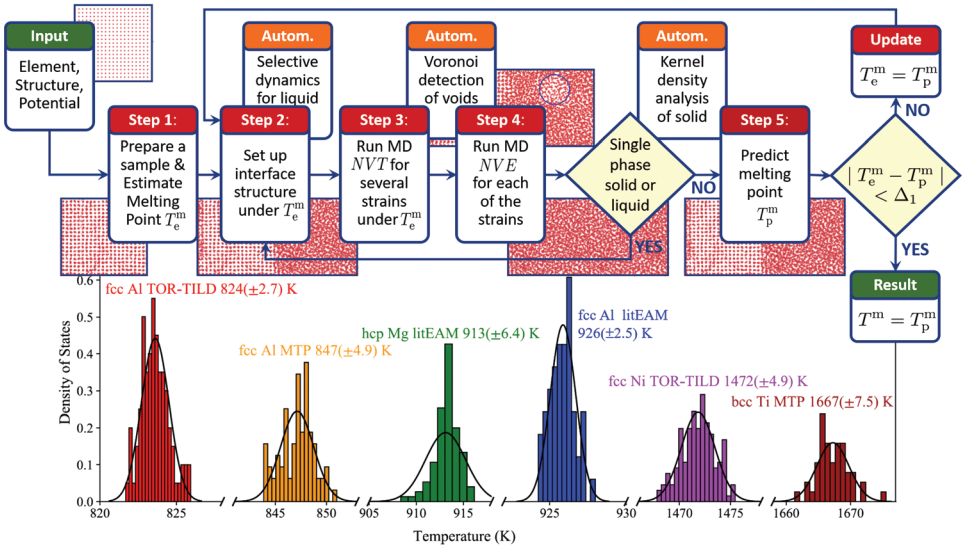

Figure 7. Calculation of the melting point of three diffeent interatomic potentials for fcc Al, as well as of potentials for fcc Ni, bcc Ti, and hcp Mg. The automated workflow as implemented in pyiron is schematically presented in the upper part with red boxes for the major steps and orange boxes for subroutines required for automation. For each potential the workflow is executed several times and the distribution of predicted results is plotted, for which three times the standard deviation is given as error in the parentheses. Figure adapted under the terms of the CC-BY licence.[70] Copyright 2021, The Authors.

neighbor analysis,[75] the detection rate turned out to be insufficient in the present context. Making use of the geometry of the supercell, a more efficient scheme based on a kernel density analysis was, therefore, developed.

Only with such a combination of scientific insights and a computational development environment was it possible to design a fully automated workflow for the melting point that robustly handles all the particularities of various potentials.[70] The lower part of Figure 7 shows examples for a variety of different potentials, including EAM potentials taken from the literature, on-the-fly developed potentials using the TOR-TILD methodology (*Two-Optimized Reference Thermodynamic Integration using Langevin Dynamics*),[72,76] and machine-learning potentials such as moment tensor potentials.[77] This gives access to a statistical analysis of confidence intervals and a comparison of the performance of these potentials. The Jupyter notebook for the automized workflow is publically available and can be downloaded at [www.mpie.de/4008196/Software] and [github.com/pyiron/pyiron_meltingpoint]. It can be executed after the desired element and potential have been selected. The application with Snakemake is explained in ref. [70].

5. Challenges of Workflow Frameworks

Workflows help construct the process topology, execute simulations, and include generic mechanisms to pass data from step to step. However, the data flow might not provide specific methods to transform the data so that it becomes usable in the next steps. The first issue that had to be solved was to ensure syntactic interoperability, that is, standard data formats, and data transfer protocols.

Several groups have developed solutions for this issue that are already available: In the ASE,[35] the simulation codes are wrapped by objects called calculators. The ASE Calculators process their input and produce outputs using Python data structures such as dictionaries and lists, which can be serialized in the standard JSON data format for dataflow transfers within a workflow. Besides, atomic structure data in ASE are captured and processed with Atoms objects that can be JSON serialized and passed to the next workflow steps or stored in databases for later reuse by other steps in the workflow.

A similar approach is adopted in Pymatgen,[78] the main package used in the Materials Project.[11] Notably, the REST (rpresentational state transfer) interface, implemented using the HTTP standard for data transfer and JSON standard for data encoding, has been employed for seamless data exchange between different Pymatgen components and users in a distributed fashion.[78] The framework developed in ref. [79] provides a graphical model editor allowing domain-specific data design based on a special meta-model for scientific data and automatically generates a REST interface to diverse data stores, such as SQL and NoSQL databases, and Amazon S3.

AiiDA instead addresses this issue with the concept of calcfunctions: any Python function that performs data operations and manipulations can be wrapped with the AiiDA *@calcfunction* decorator. Each execution of a calcfunction will be automatically stored by AiiDA and represented with interconnected nodes in the AiiDA graph, ensuring that the provenance describing how the outputs were generated is automatically tracked and stored.

Beyond converting between data formats, there are efforts to address semantic interoperability, for example, linking data entities with different names and the same meaning. An even more difficult problem is to map relationships between data entities into different representations. It remains an open challenge for the materials modeling domain due to the high heterogeneity, variety, complexity, and dynamics of materials.[80] Currently, the EMMC (European Materials Modeling Council - www.emmc.info) makes efforts to develop a materials ontology that can help different programs to "understand" the data scheme of input data from a large variety of sources, both simulation and experimental characterization of materials.[9] In the future, such an ontology can give rise to domain-specific languages for data in materials science, similar to, for instance, the chemical markup language developed in the past.[81]

In this context, a notable standardization effort that is worth mentioning is the OPTIMADE consortium (www.optimade.org). OPTIMADE involves more than ten of the major materials databases worldwide and is open to further participants. The consortium has developed a standard REST API specification[82] to allow to query and extract crystal structures and related metadata from any server using the same syntax, and is actively working to further extend the specifications to more data types that are relevant in materials science (like molecular-dynamics trajectories and computer simulations, for instance).

The preservation of input data, custom parameters, and order of workflow components is a feature of most workflow frameworks. However, the level of detail at which this information is captured varies significantly between the various workflow frameworks, and as a consequence, also the reproducibility level that the framework can actually guarantee. A crucial element in capturing the complete workflow is the proper description of the utilized software. Separating the standalone software packages wrapped in the framework still is a challenge due to the software version, the underlying platform, compiler settings, and other factors that might influence the computation result and compatibility with the workflow framework itself. In KNIME, for instance, workflow templates link to a specific version of a node. If newer versions for a node are available, the old nodes will be marked as deprecated but still part of the workflow template, and the user has to decide whether to use the newer version of the node. Similarly, calculation nodes in AiiDA always have an input node in the provenance graph, representing the actual executable and machine where the code was run and thus allowing to trace back the code that was used and its run environment. Another solution to the issue is to use software encapsulated in a virtual machine or Docker containers, which make execution independent of the platform where it is run. We expect that a broader support for efficient code containerization also in the context of HPC simulations, and their adoption in major HPC centers, will help to effectively address this issue in the next few years. However, most workflow frameworks also provide the option to incorporate web services or are entirely based on connecting web services like MDStudio (www.github.com/MD-Studio/MDStudio). In this case, the workflow framework cannot preserve the state of

this workflow element and changes to it. Its discontinuation can either lead to unexpected results or failure of the workflow.

The primary purpose of workflows is to limit the required user input. Nevertheless, it is not always possible or feasible to completely automate all required decisions. Thus, many workflow frameworks provide the capabilities to insert breakpoints and user decisions into the workflow. Generally speaking, the availability of a GUI, specifically a workflow editor, significantly enhances the clarity and, most importantly, makes workflows easier to implement and more transferable. An alternative approach to improve transferability has been recently implemented in the wfGenes tool[83] that automatically generates workflows for different workflow management systems from a simple workflow description language, abstracting from the details of the target platforms. Aspects of maintenance and transferability are notoriously overlooked in academic software development projects, where usually individual students/postdocs or small groups develop software with their specific application in mind. Lack of documentation and clarity often limits the reuse of the software even within the same group.

Encapsulating software complexity into a workflow with a clear GUI comes at the cost of the creation of it. The creation of the graphical user interface for modules and workflows requires an in-depth knowledge of the workflow engine for many available systems. Often this creates a barrier to the use of GUIs, in particular in academic development projects where rapid prototyping is required. Within state-of-the-art software, it is necessary to incorporate new modules and protocols within the workflow framework quickly. Few systems available have paid attention to this bottleneck. One example is the SimStack framework. The client mode generates GUIs for a set of exposed parameters from an XML file. This concept enables the nonexpert developer to build a simple GUI, exposing the parameters required for a particular application of any code without spending much time. At a later stage, when the workflow is mature, the GUI can be enhanced to meet more complex needs.

The capability to document and describe a complex simulation is one of the most compelling arguments for workflow frameworks. The distribution of workflows to colleagues, the community, or customers is of central importance. While simulations can also be described informally and shared on a platform like protocols.io, distribution of formalized workflows improves reusability due to the reasons mentioned above. For this purpose, either dedicated repositories such as www.nodepit.com/product/nodepit, but also workflow framework-agnostic platforms such as www.myexperiment.org[84,85] are available. In the case of Python frameworks such as AiiDA, Pyiron or FireWorks, Jupyter notebooks can enrich the workflow Python code with interspersed documentation.

6. Summary

In the past few years, workflow frameworks have increased their importance as mandatory tools to perform complex computational studies: they automate large high-throughput simulation projects and capture and formalize all tasks, thereby dramatically improving reproducibility and reusability of the study and, thus, its impact. Many workflow frameworks are already available and used for simulations in materials science. However, it is crucial to keep in mind the benefits and shortcomings of each of them. The best workflow framework for any given project may depend on the user expertise, availability of tools and plugins for the desired task, capability to connect to computational resources, as well as the possibility to adjust, reuse and share workflows and their results within a group, with collaborators or with the whole scientific community. In addition, the specific requirements of the application use cases can strongly influence the choice of a workflow framework. While workflow frameworks have started to gain relevance in virtual materials design, efforts to make the adoption of scientific workflow frameworks more widespread in the field are still required.

Acknowledgements

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 957189 (BIG-MAP). It is also part of BATTERY 2030+ initiative under grant agreement No. 957213. J.S., J.Y., C.R.C.R., W.W., (KIT), T.H., and J.N. (MPIE) thank the German Federal Ministry of Education and Research (BMBF) for financial support of the project Innovation-Platform MaterialDigital through project funding FKZ No: 13XP5094A (KIT) and 13XP5094C (MPIE). G.P., S.P.H., and L.K. acknowledge support by the MARVEL National Centre of Competence in Research (NCCR) funded by the Swiss National Science Foundation (grant agreement ID 51NF40-182892) and by the European Union's Horizon 2020 research and innovation programme under grant agreement No. 824143 (European MaX Centre of Excellence "Materials design at the Exascale"). F.T.B. and I.E.C. acknowledge support from the Independent Research Fund Denmark (Green Transition Project 1, title "Reconfigurable Metamaterials for Next Generation High-Capacity Batteries," grant number 0217-00111B).

Open access funding enabled and organized by Projekt DEAL.

Conflict of Interest

The authors declare no conflict of interest.

Keywords

high-throughput materials simulation, multi-scale modeling, multi-scale simulation

Received: August 27, 2021
Revised: November 22, 2021
Published online:

[1] T. DebRoy, T. Mukherjee, J. O. Milewski, J. W. Elmer, B. Ribic, J. J. Blecher, W. Zhang, *Nat. Mater.* **2019**, *18*, 1026.
[2] A. Rinaldi, *EMBO Rep.* **2007**, *8*, 995.
[3] A. C. Ngandjong, T. Lombardo, E. N. Primo, M. Chouchane, A. Shodiev, O. Arcelus, A. A. Franco, *J. Power Sources* **2021**, *485*, 229320.
[4] B. Wu, W. D. Widanage, S. Yang, X. Liu, *Energy AI* **2020**, *1*, 100016.
[5] E. van der Giessen, P. A. Schultz, N. Bertin, V. V. Bulatov, W. Cai, G. Csányi, S. M. Foiles, M. G. D. Geers, C. González, M. Hütter, W. K. Kim, D. M. Kochmann, J. LLorca, A. E. Mattsson, J. Rottler,

A. Shluger, R. B. Sills, I. Steinbach, A. Strachan, E. B. Tadmor, *Model. Simul. Mater. Sci. Eng.* **2020**, *28*, 043001.
[6] P. Giannozzi, S. Baroni, N. Bonini, M. Calandra, R. Car, C. Cavazzoni, D. Ceresoli, G. L. Chiarotti, M. Cococcioni, I. Dabo, A. D. Corso, S. de Gironcoli, S. Fabris, G. Fratesi, R. Gebauer, U. Gerstmann, C. Gougoussis, A. Kokalj, M. Lazzeri, L. Martin-Samos, N. Marzari, F. Mauri, R. Mazzarello, S. Paolini, A. Pasquarello, L. Paulatto, C. Sbraccia, S. Scandolo, G. Sclauzero, A. P. Seitsonen, et al., *J. Phys.: Condens. Matter* **2009**, *21*, 395502.
[7] N. Zographos, C. Zechner, I. Martin-Bragado, K. Lee, Y.-S. Oh, *Mater. Sci. Semicond. Process.* **2017**, *62*, 49.
[8] M. J. Buehler, *MRS Bull.* **2013**, *38*, 169.
[9] R. Lula, A. Baas, *What Makes a Material Function? Let Me Compute the Ways: Modelling in FP7 NMP Programme Materials Projects*, Publications Office of the European Union, Luxembourg **2012**.
[10] M. L. Green, C. L. Choi, J. R. Hattrick-Simpers, A. M. Joshi, I. Takeuchi, S. C. Barron, E. Campo, T. Chiang, S. Empedocles, J. M. Gregoire, A. G. Kusne, J. Martin, A. Mehta, K. Persson, Z. Trautt, J. V. Duren, A. Zakutayev, *Appl. Phys. Rev.* **2017**, *4*, 011105.
[11] 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.* **2013**, *1*, 011002.
[12] J. Hachmann, R. Olivares-Amaya, S. Atahan-Evrenk, C. Amador-Bedolla, R. S. Sánchez-Carrera, A. Gold-Parker, L. Vogt, A. M. Brockway, A. Aspuru-Guzik, *J. Phys. Chem. Lett.* **2011**, *2*, 2241.
[13] Y. Gil, E. Deelman, M. Ellisman, T. Fahringer, G. Fox, D. Gannon, C. Goble, M. Livny, L. Moreau, J. Myers, *Computer* **2007**, *40*, 24.
[14] E. Deelman, D. Gannon, M. Shields, I. Taylor, *Future Gener. Comput. Syst.* **2009**, *25*, 528.
[15] B. Ludäscher, I. Altintas, C. Berkley, D. Higgins, E. Jaeger, M. Jones, E. A. Lee, J. Tao, Y. Zhao, *Concurrency Comput.: Pract. Exper.* **2006**, *18*, 1039.
[16] S. Curtarolo, W. Setyawan, G. L. 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.* **2012**, *58*, 218.
[17] B. Lutz, C. Sinner, S. Bozic, I. Kondov, A. Schug, *BMC Bioinf.* **2014**, *15*, 292.
[18] A. Jain, S. P. Ong, W. Chen, B. Medasani, X. Qu, M. Kocher, M. Brafman, G. Petretto, G.-M. Rignanese, G. Hautier, D. Gunter, K. A. Persson, *Concurrency Comput.: Pract. Exper.* **2015**, *27*, 5037.
[19] G. Pizzi, A. Cepellotti, R. Sabatini, N. Marzari, B. Kozinsky, *Comput. Mater. Sci.* **2016**, *111*, 218.
[20] M. D. Wilkinson, M. Dumontier, I. J. Aalbersberg, G. Appleton, M. Axton, A. Baak, N. Blomberg, J.-W. Boiten, L. B. da Silva Santos, P. E. Bourne, J. Bouwman, A. J. Brookes, T. Clark, M. Crosas, I. Dillo, O. Dumon, S. Edmunds, C. T. Evelo, R. Finkers, A. Gonzalez-Beltran, A. J. Gray, P. Groth, C. Goble, J. S. Grethe, J. Heringa, P. A. 't Hoen, R. Hooft, T. Kuhn, R. Kok, J. Kok, et al., *Sci. Data* **2016**, *3*, 1.
[21] S. P. Huber, S. Zoupanos, M. Uhrin, L. Talirz, L. Kahle, R. Häuselmann, D. Gresch, T. Müller, A. V. Yakutovich, C. W. Andersen, F. F. Ramirez, C. S. Adorf, F. Gargiulo, S. Kumbhar, E. Passaro, C. Johnston, A. Merkys, A. Cepellotti, N. Mounet, N. Marzari, B. Kozinsky, G. Pizzi, *Scientific Data* **2020**, *7*, 300.
[22] M. Uhrin, S. P. Huber, J. Yu, N. Marzari, G. Pizzi, *Comput. Mater. Sci.* **2021**, *187*, 110086.
[23] S. P. Huber, E. Bosoni, M. Bercx, J. Bröder, A. Degomme, V. Dikan, K. Eimre, E. Flage-Larsen, A. Garcia, L. Genovese, D. Gresch, C. Johnston, G. Petretto, S. Poncé, G.-M. Rignanese, C. J. Sewell, B. Smit, V. Tseplyaev, M. Uhrin, D. Wortmann, A. V. Yakutovich, A. Zadoks, P. Zarabadi-Poor, B. Zhu, N. Marzari, G. Pizzi, *arXiv:2105.05063* **2021**.
[24] A. V. Yakutovich, K. Eimre, O. Schütt, L. Talirz, C. S. Adorf, C. W. Andersen, E. Ditler, D. Du, D. Passerone, B. Smit, N. Marzari, G. Pizzi, C. A. Pignedoli, *Comput. Mater. Sci.* **2021**, *188*, 110165.
[25] V. Vitale, G. Pizzi, A. Marrazzo, J. R. Yates, N. Marzari, A. A. Mostofi, *npj Comput. Mater.* **2020**, *6*, 66.
[26] L. Talirz, S. Kumbhar, E. Passaro, A. V. Yakutovich, V. Granata, F. Gargiulo, M. Borelli, M. Uhrin, S. P. Huber, S. Zoupanos, C. S. Adorf, C. W. Andersen, O. Schütt, C. A. Pignedoli, D. Passerone, J. VandeVondele, T. C. Schulthess, B. Smit, G. Pizzi, N. Marzari, *Sci. Data* **2020**, *7*, 299.
[27] K. Mathew, J. H. Montoya, A. Faghaninia, S. Dwarakanath, M. Aykol, H. Tang, I. heng Chu, T. Smidt, B. Bocklund, M. Horton, J. Dagdelen, B. Wood, Z.-K. Liu, J. Neaton, S. P. Ong, K. Persson, A. Jain, *Comput. Mater. Sci.* **2017**, *139*, 140.
[28] M. R. Berthold, N. Cebron, F. Dill, T. R. Gabriel, T. Kötter, T. Meinl, P. Ohl, C. Sieb, K. Thiel, B. Wiswedel, in *Data Analysis, Machine Learning and Applications* (Eds: C. Preisach, H. Burkhardt, L. Schmidt-Thieme, R. Decker), Springer, Berlin, Heidelberg **2008**, pp. 319–326.
[29] R Core Team, R: A Language and Environment for Statistical Computing, **2018**.
[30] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, I. H. Witten, *ACM SIGKDD Explor. Newslett.* **2009**, *11*, 10.
[31] D. Gilbert, *The JFreeChart Class Library Developer Guide v l. 0.0* **2005**, https://www.jfree.org/jfreechart/devguide.html (accessed: August 2021).
[32] W. A. Warr, *J. Comput.-Aided Mol. Des.* **2012**, *26*, 801.
[33] J. Janssen, S. Surendralal, Y. Lysogorskiy, M. Todorova, T. Hickel, R. Drautz, J. Neugebauer, *Comput. Mater. Sci.* **2019**, *163*, 24.
[34] J. J. Mortensen, M. Gjerding, K. S. Thygesen, *J. Open Source Software* **2020**, *5*, 1844.
[35] A. H. Larsen, J. J. Mortensen, J. Blomqvist, I. E. Castelli, R. Christensen, M. Dułak, 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* **2017**, *29*, 273002.
[36] J. J. Mortensen, L. B. Hansen, K. W. Jacobsen, *Phys. Rev. B* **2005**, *71*, 035109.
[37] G. Kresse, J. Hafner, *Phys. Rev. B* **1993**, *47*, 558.
[38] 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. D. Corso, S. de Gironcoli, P. Delugas, R. A. D. 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, et al., *J. Phys.: Condens. Matter* **2017**, *29*, 465901.
[39] V. Blum, R. Gehrke, F. Hanke, P. Havu, V. Havu, X. Ren, K. Reuter, M. Scheffler, *Comput. Phys. Commun.* **2009**, *180*, 2175.
[40] S. Haastrup, M. Strange, M. Pandey, T. Deilmann, P. S. Schmidt, N. F. Hinsche, M. N. Gjerding, D. Torelli, P. M. Larsen, A. C. Riis-Jensen, et al., *2D Mater.* **2018**, *5*, 042002.
[41] M. Gjerding, T. Skovhus, A. Rasmussen, F. Bertoldo, A. H. Larsen, J. J. Mortensen, K. S. Thygesen, *arXiv:2104.13431* **2021**.
[42] A. Manthiram, X. Yu, S. Wang, *Nat. Rev. Mater.* **2017**, *2*, 16103.
[43] L. Kahle, A. Marcolongo, N. Marzari, *Energy Environ. Sci.* **2020**, *13*, 928.
[44] N. Mounet, M. Gibertini, P. Schwaller, D. Campi, A. Merkys, A. Marrazzo, T. Sohier, I. E. Castelli, A. Cepellotti, G. Pizzi, N. Marzari, *Nat. Nanotechnol.* **2018**, *13*, 246.
[45] L. Kahle, A. Marcolongo, N. Marzari, *Phys. Rev. Mater.* **2018**, *2*, 065405.
[46] L. Kahle, A. Marcolongo, N. Marzari, *Mater. Cloud Arch.* **2019**, DOI: 10.24435/materialscloud:2019.0077/v1.
[47] L. Kahle, X. Cheng, T. Binninger, S. D. Lacey, A. Marcolongo, F. Zipoli, E. Gilardi, C. Villevieille, M. El Kazzi, N. Marzari, D. Pergolesi, *Solid State Ionics* **2020**, *347*, 115226.

[48] N. Flores-González, N. Minafra, G. Dewald, H. Reardon, R. I. Smith, S. Adams, W. G. Zeier, D. H. Gregory, *ACS Mater. Lett.* **2021**, *3*, 652.
[49] F. Symalla, P. Friederich, A. Massé, V. Meded, R. Coehoorn, P. Bobbert, W. Wenzel, *Phys. Rev. Lett.* **2016**, *117*, 27.
[50] T. Neumann, D. Danilov, C. Lennartz, W. Wenzel, *J. Comput. Chem.* **2013**, *34*, 2716.
[51] P. Friederich, F. Symalla, V. Meded, T. Neumann, W. Wenzel, *J. Chem. Theory Comput.* **2014**, *10*, 3720.
[52] P. Friederich, V. Gómez, C. Sprau, V. Meded, T. Strunk, M. Jenne, A. Magri, F. Symalla, A. Colsmann, M. Ruben, W. Wenzel, *Adv. Mater.* **2017**, *29*, 1703505.
[53] V. Rodin, F. Symalla, V. Meded, P. Friederich, D. Danilov, A. Poschlad, G. Nelles, F. von Wrochem, W. Wenzel, *Phys. Rev. B* **2015**, *91*, 15.
[54] I. Kondov, P. Faubert, C. Müller, *Electrochim. Acta* **2017**, *236*, 260.
[55] P. Faubert, I. Kondov, D. Qazzazie, O. Yurchenko, C. Müller, *MRS Commun.* **2018**, *8*, 160.
[56] M. Vandichel, K. Laasonen, I. Kondov, *Top. Catal.* **2020**, *63*, 833.
[57] N. Akbari, I. Kondov, M. Vandichel, P. Aleshkevych, M. M. Najafpour, *Inorg. Chem.* **2021**, *60*, 5682.
[58] Dassault Systèmes, BIOVIA Pipeline Pilot **2021,** https://www.3ds.com/products-services/biovia/products/molecular-modeling-simulation/biovia-materials-studio/ (accessed: August 2020).
[59] Dassault Systèmes, BIOVIA Materials Studio **2021,** https://www.3ds.com/products-services/biovia/products/data-science/pipeline-pilot/ (accessed: August 2020).
[60] R. L. Akkermans, N. A. Spenley, S. H. Robertson, *Mol. Sim.* **2013**, *39*, 1153.
[61] H. Sun, Z. Jin, C. Yang, R. L. Akkermans, S. H. Robertson, N. A. Spenley, S. Miller, S. M. Todd, *J. Mol. Model.* **2016**, *22*, 47.
[62] F. Hanke, N. Modrow, R. L. Akkermans, I. Korotkin, F. C. Mocanu, V. A. Neufeld, M. Veit, *J. Electrochem. Soc.* **2020**, *167*, 013522.
[63] A. S. Rosen, J. M. Notestein, R. Q. Snurr, *J. Comput. Chem.* **2019**, *40*, 1305.
[64] A. S. Tygesen, J. H. Chang, T. Vegge, J. M. García-Lastra, *npj Comput. Mater.* **2020**, *6*, 65.
[65] F. T. Bölle, N. R. Mathiesen, A. J. Nielsen, T. Vegge, J. M. García Lastra, I. E. Castelli **2020**, *3*, 470.
[66] F. T. Bölle, A. E. Mikkelsen, K. S. Thygesen, T. Vegge, I. E. Castelli, *npj Comput. Mater.* **2021**, *7*, 41.
[67] N. R. Mathiesen, H. Jonsson, T. Vegge, J. M. García Lastra, *J. Chem. Theor. Comput.* **2019**, *15*, 3215.
[68] F. T. Bölle, A. Bhowmik, T. Vegge, J. M. García Lastra, I. E. Castelli, *Batteries Supercaps* **2021**, *4*, 2100086.
[69] M.-T. F. Rodrigues, G. Babu, H. Gullapalli, K. Kalaga, F. N. Sayed, K. Kato, J. Joyner, P. M. Ajayan, *Nat. Energy* **2017**, *2*, 8.
[70] L.-F. Zhu, J. Janssen, S. Ishibashi, F. Körmann, B. Grabowski, J. Neugebauer, *Comput. Mater. Sci.* **2021**, *187*, 110065.
[71] D. Alfe, M. Gillan, G. Price, *J. Chem. Phys.* **2002**, *116*, 7127.
[72] L.-F. Zhu, B. Grabowski, J. Neugebauer, *Phys. Rev. B* **2017**, *96*, 27.
[73] F. Lindemann, *Phys. Z.* **1910**, *11*, 609.
[74] M. Born, *J. Chem. Phys.* **1939**, *7*, 591.
[75] A. Stukowski, *Model. Simul. Mater. Sci. Eng.* **2012**, *20*, 4.
[76] L.-F. Zhu, F. Körmann, A. Ruban, J. Neugebauer, B. Grabowski, *Phys. Rev. B* **2020**, *101*, 3.
[77] A. V. Shapeev, *Multiscale Model. Simul.* **2016**, *14*, 1153.
[78] 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.* **2013**, *68*, 314.
[79] A. Bender, A. Poschlad, S. Bozic, I. Kondov, *Proc. Comput. Sci.* **2013**, *18*, 1087.
[80] J. R. Rumble, *Integr. Mater. Manuf. Innovation* **2017**, *6*, 172.
[81] P. Murray-Rust, J. A. Townsend, S. E. Adams, W. Phadungsukanan, J. Thomas, *J. Cheminf.* **2011**, *3*, 43.
[82] C. W. Andersen, R. Armiento, E. Blokhin, G. J. Conduit, S. Dwaraknath, M. L. Evans, Á. Fekete, A. Gopakumar, S. Gražulis, A. Merkys, F. Mohamed, C. Oses, G. Pizzi, G.-M. Rignanese, M. Scheidgen, L. Talirz, C. Toher, D. Winston, R. Aversa, K. Choudhary, P. Colinet, S. Curtarolo, D. D. Stefano, C. Draxl, S. Er, M. Esters, M. Fornari, M. Giantomassi, M. Govoni, G. Hautier, et al., *arXiv:2104.13431* **2021**.
[83] M. Roozmeh, I. Kondov, in *IEEE/ACM Workflows in Support of Large-Scale Science (WORKS)*, IEEE, Piscataway, NJ **2020**, pp. 9–16.
[84] C. A. Goble, J. Bhagat, S. Aleksejevs, D. Cruickshank, D. Michaelides, D. Newman, M. Borkum, S. Bechhofer, M. Roos, P. Li, D. D. Roure, *Nucleic Acids Res.* **2010**, *38*, W677.
[85] D. D. Roure, C. Goble, R. Stevens, *Future Gener. Comput. Syst.* **2009**, *25*, 561.