An Automatized Simulation Workflow for Powder Pressing Simulations Using SimStack

Bjoern Mieller,* Masood Valavi, and Celso Ricardo Caldeira Rêgo*

Automated computational workflows are a powerful concept that can improve the usability and reproducibility of simulation and data processing approaches. Although used very successfully in bioinformatics, workflow environments in materials science are currently commonly applied in the field of atomistic simulations. This work showcases the integration of a discrete element method (DEM) simulation of powder pressing in the convenient SimStack workflow environment. For this purpose, a Workflow active Node (WaNo) was developed to generate input scripts for the DEM solver using LIGGGHTS Open Source Discrete Element Method Particle Simulation code. Combining different WaNos in the SimStack framework makes it possible to build workflows and loop over different simulation or evaluation conditions. The functionality of the workflows is explained, and the added user value is discussed. The procedure presented here is an example and template for many other simulation methods and issues in materials science and engineering.

1. Introduction

An important pillar in the digitalization of materials science and engineering (MSE) is the implementation of computational infrastructures for structured workflows to improve reproducibility and simplify the use of computational simulations and data processing. A dedicated workflow environment is a powerful tool that by far not only standardizes the sequence of calculation steps but rather handles complex tasks like establishing the connection to a high-performance computing (HPC) cluster, calling up various programs and functions, running loops and systematic series of simulations, up to aggregating and evaluating results.

The use of workflow systems has been on the rise for several years. In life science, for example, Taverna offers a graphical user interface (GUI) to run bioinformatics experiments.[1] In contrast, Nextflow, another workflow environment with a strong background in bioinformatics, uses a domain-specific language to define the sequence and conditions of computing steps that result in a workflow.[2]

The Python-based AiiDA workflow system relies on an application programming interface instead of a GUI. It is designed to use various simulation codes in different scientific disciplines.[3] In MSE, a variety of workflow environments for ab initio calculations have been proposed, including Atomate, MAST, pyiron, and SimStack.[4–7]

This very brief introduction to workflow environments already shows that there are manifold concepts and philosophies to choose from when integrating scientific and computational workflows. A detailed comparison of the pros and cons of several workflow environments was published by Schaarschmidt et al.[8] As the respective literature is often written by and addressed to programmers and data scientists, the idea and benefit of computational workflows remain abstract or intimidating to more experimentally focused or technology-oriented materials scientists. It is relatively easy to get started with workflows in interdisciplinary teams. However, this is preceded by the insight and motivation of the experimentalists to join such a team.

Contrary to the viewpoint of some programmers and computational material scientists that the automation of simulations via workflows is only viable after a simulation approach is fully established, this work introduces SimStack as a dynamic workflow environment for developing a powder pressing simulation. By implementing the discrete element method (DEM) through the open-source library LIGGGHTS, this study showcases the advantageous utilization of workflow automation from the onset of the development process. It highlights how developers can expedite the developmental timeline by integrating the workflow approach early. Moreover, it demonstrates the ease of an example with a series of script-based simulations that can be defined and seamlessly executed on an HPC cluster through SimStack's user-friendly GUI. This attribute makes SimStack particularly appealing to experimentalists who may not have extensive computational background. The initial sections of the report provide a thorough overview of the SimStack workflow environment

B. Mieller, M. Valavi
Division of Advanced Multi-materials Processing
Bundesanstalt für Materialforschung und -prüfung (BAM)
Unter den Eichen 87, Berlin 12205, Germany
E-mail: bjoern.mieller@bam.de

C. R. Caldeira Rêgo
Institute of Nanotechnology (INT)
Karlsruhe Institute of Technology (KIT)
Hermann-von-Helmholtz-Platz 1, 76344 Eggenstein-Leopoldshafen, Germany
E-mail: celso.rego@kit.edu

The ORCID identification number(s) for the author(s) of this article can be found under https://doi.org/10.1002/adem.202400872.

and the DEM simulation capabilities of LIGGGHTS, setting the stage for the detailed objectives of the study.

1.1. SimStack

SimStack emerges as a holistic workflow framework crafted to tackle the intricacies of multiscale modeling across the scientific domain. It promotes the seamless integration of methodologies from disciplines as diverse as physics, materials science, chemistry, and computer science to characterize a system's properties accurately. Such integrations are vital for simulations that span multiple scales, necessitating the fusion of disparate methods into bespoke workflows. Traditionally, the construction of these workflows demanded a vast reservoir of expertise, encompassing scripting, command-line execution, and a profound comprehension of all involved methods and tools. By offering a platform that facilitates the rapid prototyping of simulation workflows through an intuitive drag-and-drop interface, SimStack significantly diminishes the complexity and requisite expertise for establishing multiscale simulations. Consequently, it renders advanced modeling techniques more accessible to a broader scientific audience, thereby democratizing the process of scientific investigation and discovery. Its client-server approach is at the heart of SimStack's work, simplifying the construction, modification, and configuration of workflows. Users can easily design and submit workflows on remote HPC resources, with SimStack handling data transfer and job management. This is facilitated through a GUI that abstracts the underlying complexities of high-performance computations. Workflow active Nodes (WaNos) form the building blocks of SimStack, allowing users to assemble new workflows or modify existing ones quickly. This ensures that even those without extensive computational background can engage in sophisticated modeling tasks, enhancing productivity and promoting innovation in materials design and other scientific fields. The advantages of SimStack to the scientific community are multifaceted: 1) It democratizes access to complex computational modeling by reducing the need for specialized knowledge in setting up and running simulations. This enables a broader range of researchers to efficiently explore and prototype simulation workflows, fostering collaboration and knowledge sharing across disciplines. 2) SimStack's design emphasizes reproducibility and scalability. By providing a standardized framework for simulations, it ensures that results are consistent and verifiable, addressing a critical challenge in scientific research. 3) The framework promotes the rapid adoption of new computational models and methods, facilitating the transition from theoretical development to practical application.

SimStack provides powerful tools for scientific discovery, advancing computational modeling according to the FAIR and TRUE principles.[9,10]

1.2. DEM LIGGGHTS

LIGGGHTS is an open-source particle simulation software for DEM calculations. The software does not offer a GUI, but executes commands that describe the simulation task in the form of an input script. The input script is a simple text file that is processed line by line during execution in LIGGGHTS. It principally contains a sequence of parameters, fix, compute, and dump commands to create and manipulate particles (fix), calculate specified values (compute), and store respective data (dump). As a result, various files are generated that have to be analyzed and evaluated using third-party software.

A decent introduction to DEM is provided by Ramírez-Aragón and co-workers.[11] In their study, among other things, the influence of the particle size distribution and the length of the calculation time step on the maximum force during powder compaction was analyzed. The authors emphasize that the study focuses on the comparison of contact models and does not propose models for real systems.[11] Leps and Hartzell investigated the effect of particle size distribution on the shear behavior of magnetorheological fluids.[12] By including a custom magnetic model, experimental data could be convincingly approximated. Podlozhnyuk, Pirker, and Kloss demonstrated realistic behavior of nonspherical particles in processes like static packing, hopper discharge, or rearrangement on moving bases by implementing superquadric particle shapes in LIGGGHTS.[13] Solid–liquid mixing in a stirred tank was investigated by Blais et al. using a combination of LIGGGHTS and OpenFOAM for fluid dynamics.[14] Although their coupled model was able to accurately predict the proportion of suspended particles, uncertainties remain regarding the effects of particular DEM parameters such as the coefficients of restitution and friction.

The examples of DEM-based studies lined out here show that a productive or even predictive use of the DEM approach necessitates a thorough calibration of the simulation model and adaption of parameters to the actual use case.[15] Workflow environments can support this process by automation in the first place. Furthermore, a common practice of publishing successfully adapted workflows would create a pool of templates as starting point for other developers, thereby speeding up subsequent adaption processes as well.

With this perspective in mind, this study is intended to be a starting point for the use of workflow clients for the development and operation of DEM simulations. For this purpose, the development of a SimStack WaNo and its integration into an automated workflow is reported. The system creates scripted batch jobs and so is aligned with and familiar to anyone working with conventional approaches. However, the parameters or commands relevant to the development task are implemented as variables. The values of these variables can then be set by the user via a graphical interfaces or automatically by program loops when using the workflow client.

2. Results and Discussion

2.1. Simulation of Powder Pressing

The core of the simulation workflow presented here is the DEM simulation of a pressing process in LIGGGHTS. This is defined in an input script. The input script and, consequently, the sequence of the simulation are divided into the following principle segments: 1) description of general simulation parameters; 2) description of the particles to be inserted into the simulation space; 3) loading of walls and press stamps into the simulation space; 4) execution of particle insertion into the simulation space; 5) computation of particle settlement under gravity; 6) execution

of the downward movement of the upper press stamp; and 7) computation of position and force on upper press stamp.

The general description includes parameters like the contact model to be applied (here, granular hertz model), the length of a timestep for the calculations (here, 0.0001 s), and the region of the simulation space (here, an upright cuboid). The height of the cuboid must be chosen generously so that sufficient volume is available to insert the desired number of particles. Two materials with high and low Young's modulus are defined for the particles to represent rigid ceramics and softer polymer binders. Six types of spherical particles are considered, each characterized by material, density, and radius. Five rigid particle types are defined with different radii to approximate the particle size distribution of a real powder. The sixth particle type is defined as soft to represent the binder. A constant density is applied for all six particle types to ensure a homogeneous settling. The numerical proportion of the individual particle types specifies the particle size distribution. Three STL files are loaded to form the pressing mold: a rectangular pipe as a pressing die and two tightly fitting square tiles as top and bottom press stamps.

Figure 1 shows snapshots of the particles during the different simulation segments. In the first step, the insertion phase, within 20 000 timesteps (2 s), the specified particles are evenly distributed in the die. During the settling phase, the created particles fall downward for 300 000 timesteps (3 s). This reduces the fill level in the die, and the bulk density of the particles increases. Figure 1 shows that this strategy achieves a homogeneous mixture of the different particles before pressing.

In the pressing step, the upper stamp is moved downward at a constant speed until a defined maximum force on the stamp is reached. The time required to achieve the maximum force depends on the fill level and the speed of the punch. Thus, the number of timesteps to be calculated has to be adapted depending on the number of particles and the pressing speed. Three files are generated as a result. Two trajectory files contain the position of every particle at every recorded time step, one for the insertion and settling step and one for the pressing step. In addition, the position of the press stamps and the forces on the upper press stamp during the pressing phase are written to a csv file (forces.csv) for subsequent evaluation of the compression of the powder.

2.2. The LIGGGHTS WaNo

The script job concept is applied to create a SimStack WaNo that generates LIGGGHTS input files for the described pressing simulation. The general concept is that all parameters that should be
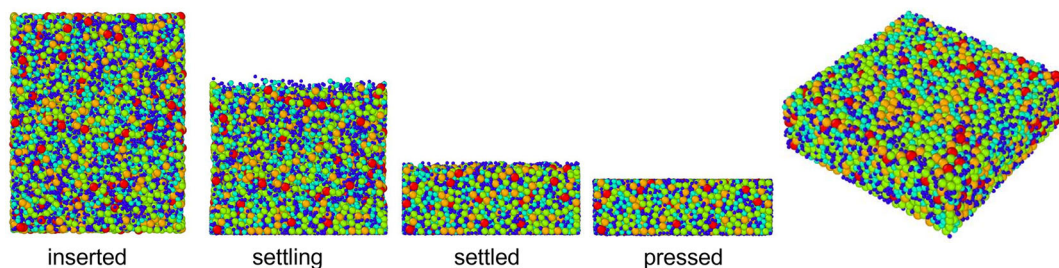
Definition of a WaNo parameter with default value:

```
<WaNoInt name="timesteps">20000</WaNoInt>
```

Appearance of parameter in the SimStack client:

```
timesteps          20000
```

Adjustment of the parameter by the user :

```
timesteps          30000
```

Transfer of parameter value in yml-file:

```
timesteps: 30000
```

Definition of variable in Python script:

```
timeSteps = int(wano_file["timesteps"])
```

Use of variable in Python script:

```
run {timeSteps}
```

Final command in compiled LIGGGHTS input file :

```
run 30000
```

Figure 2. Implementation of the script job concept in SimStack using the run command and the number of timesteps as an example.

editable for different simulations are implemented as variables in a LIGGGHTS input file template. This template is included in a Python script (createInput.py) as an advanced string. The Python script is used to read values for the variables from a yml file and insert these values at the respective locations in the advanced string, which is the input file template. Finally, the complete advanced string with all the variables inserted is saved as a separate LIGGGHTS input file named workflow.input.

createInput.py is used in the LIGGGHTS WaNo as summarized in **Figure 2**. All editable parameters of the LIGGGHTS WaNo are defined in one xml file. Figure 2 shows an example of this definition for an integer value named timesteps with a default value of 20 000. If the LIGGGHTS WaNo is loaded into the SimStack client, all variables defined in this way appear with their default values. Changes to the values can be made in the GUI.

When the WaNo is executed, a yml file with all values is generated. Figure 2 shows an example entry for the timesteps variable and a value of 30 000. After saving the yml file, createInput.py is executed automatically. The yml is loaded as wano_file, and the values are assigned to the respective variables. In the example in Figure 2, the yml entry for timesteps is set for the variable timeSteps. The variables are used to insert the values in the corresponding places in the input script template. The example in Figure 2 shows the code line of a run command in the template using the variable timeSteps. The corresponding line in the

Figure 1. Segments of the DEM powder pressing simulation with 15 000 hard particles in front view, color-coded particle radius, perspective view on the pressed compact on the right.

inserted   settling   settled   pressed

compiled workflow.input file is shown below. timeSteps is evaluated to 30 000 and consequently the run command is parameterized with 30 000 timeSteps. Using this concept, several parameters of the pressing simulation can be set using the WaNo. The LIGGGHTS WaNo takes the following into account: 1) the velocity of the upper press stamp; 2) the total number of rigid particles; 3) the size and number share of five classes of hard particles; 4) the number of timeSteps to be computed in the pressing step; 5) the dumprate for the trajectory file of the pressing step; and 6) the dumprate for the creation of vtk and stl files for visualization.

The parameter dumprate specifies after how many timeSteps an entry in the respective file is made during simulation. For example, if the pressing step is run for 20 000 timeSteps with a dumprate of 500, the corresponding trajectory file will contain 40 entries. File size and time resolution of the results can be tailored using this parameter.

vtk and stl files are necessary to visualize the results in third-party software like Paraview. A corresponding file must be created for each timeSteps to be visualized. As this can lead to many individual files, it is advisable to adjust this parameter. To facilitate the transfer and handling of these files, the WaNo automatically combines them in an archive.

In total, the following tasks are processed by the LIGGGHTS WaNo: 1) execution of createInput.py to generate workflow.input; 2) upload workflow.input and the necessary additional files to the HPC cluster connected to the SimStack client; 3) queue the simulation job on the HPC cluster; 4) monitor and display the status of the simulation job; and 5) execution of csv_to_yml.py.

Additional necessary files to be transferred to the HPC cluster include the three stl files that are loaded in the simulation to create the pressing die, the rendered_wano.yml that contains the values of all parameters, a bash file (config.sh) that specifies the order of tasks to be processed on the HPC cluster, and a short Python script (csv_to_yml.py) that converts the forces.csv to yml format. This yml file (forces_inputs.yml) is specified as the output of the WaNo and can be transferred as input to other WaNos in a workflow. Both variants of the simulation result, the csv file and the yml file, contain unprocessed position and force data. Data processing is necessary to visualize the compression of the powder as a stress–strain curve.

2.3. The DataDive WaNo

The DataDive WaNo handles the data processing of the simulation results. It executes a short Python script (dem_stress_strain.py). This script loads the forces_inputs.yml and converts the force data to mechanical stress. As in the simulation, the upper pressing stamp continuously moves downward; the initial contact with the powder must be determined based on the pressure on the stamp. The user can specify a contact pressure value, for example, 0.5 MPa, considered as point of contact. The value for the contact pressure is also entered and processed as shown in Figure 2. The distance between the press stamps is considered as the powder's thickness, whereas the upper stamp's position at the contact pressure is used to calculate the initial thickness. Data before the moment of contact are neglected, and a stress–strain curve is plotted and saved as a png file.

The processed dataset is saved as datadive_result.yml.

Technically, the evaluation step could be easily integrated into the LIGGGHTS WaNo. The following section will show that it has specific advantages in separating data generation and processing.

2.4. Workflows and Loops

Although using the LIGGGHTS WaNos alone already speeds up the preparation and execution of a simulation job, the real added value comes from combining the WaNos into workflows. **Figure 3** shows three examples of such workflows. Figure 3a is the simplest case, combining LIGGGHTS and DataDive to connect simulation and evaluation of the results. A stress–stress curve is plotted automatically by bypassing the forces_inputs.yml. To illustrate the advantage of the workflow concept even for this simple case, **Figure 4** compares the steps required to carry out and evaluate a simulation with and without the workflow environment. It is shown that in the workflow scenario, the total number of actions is reduced, and tedious and error-prone steps such as manually creating directories and copying data are eliminated.

The advantages of workflows become even more apparent when the execution of systematic series is automated. Figure 3b shows a workflow to investigate the effect of contact pressure on the resulting stress–strain curve. For this purpose, the auxiliary WaNos Mult-It and DB-Generator are included, and DataDive is placed in a loop. In the example, Mult-It lists contact pressures from 0.1 to 2.1 MPa in 0.1 MPa steps. LIGGGHTS executes one simulation and generates one forces_inputs.yml. DataDive is executed 11 times in the loop, each time with a different contact pressure, and generates 11 datadive_result.yml files. DB-Generator collects the result files and combines them in one yml file for convenient download and straightforward evaluation. This approach saves computing resources and time, as the resource-intensive and time-consuming DEM calculations are performed only once. This example clearly shows that treating data generation and evaluation separately makes sense. The results show that evaluation with higher contact pressure results in lower apparent maximum strain, i.e., less apparent powder compression. For contact pressures >1 MPa, the relative effect is less pronounced.

In the third example, Figure 3c, LIGGGHTS and DataDive are placed in a loop together. Here, Mult-It is used to create a list of numbers to be set as the maximum number of hard particles. For each number, a DEM simulation is performed and evaluated. In contrast to the second example, several forces_inputs.yml are created, but all are evaluated with the same contact pressure. DB-Generator again combines the results in one yml file. The evaluation of this yml file is automated by integrating GitHub and Colab. To do this, DB-Generator pushes the yml file to a specified GitHub repository (in the example: number_variation_normal.yml on https://github.com/BjoernMie/KNOW-NOW/).[16] This enables collaborative and distributed work with results independent of the workflow environment. To showcase such a use case, the plot in Figure 3b was created using a Python script on Colab, available on the above GitHub repo.[16] This script loads the yml file from the GitHub repository and plots the data automatically. This means that the entire workflow, from setting the parameters to the comparative visualization of the data, is carried out without transferring files manually. From the plot, it can be
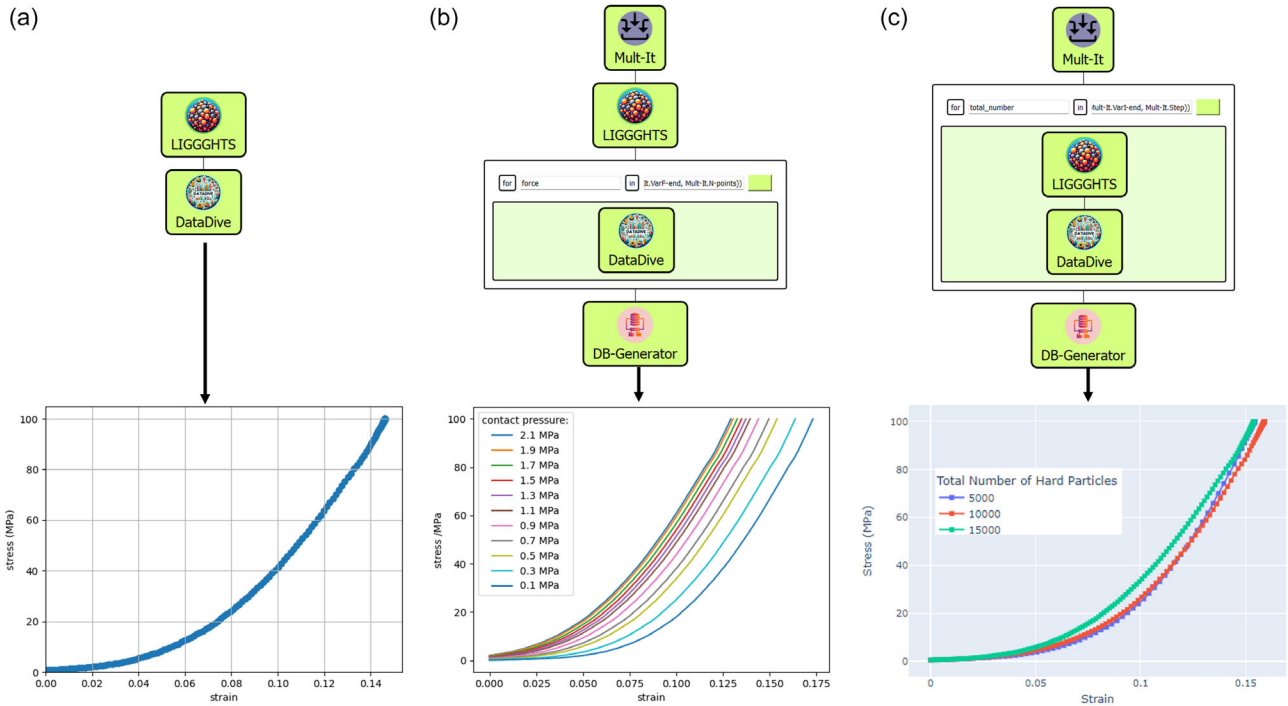
Figure 3. Examples of SimStack workflows using the LIGGGHTS WaNo: a) simple execution of one DEM pressing simulation with subsequent evaluation of stress–strain curve, b) looped workflow to test different evaluation conditions (contact pressure) based on one single DEM simulation, and c) looped workflow to perform multiple DEM calculations with different particle numbers including evaluation and aggregation of the results.
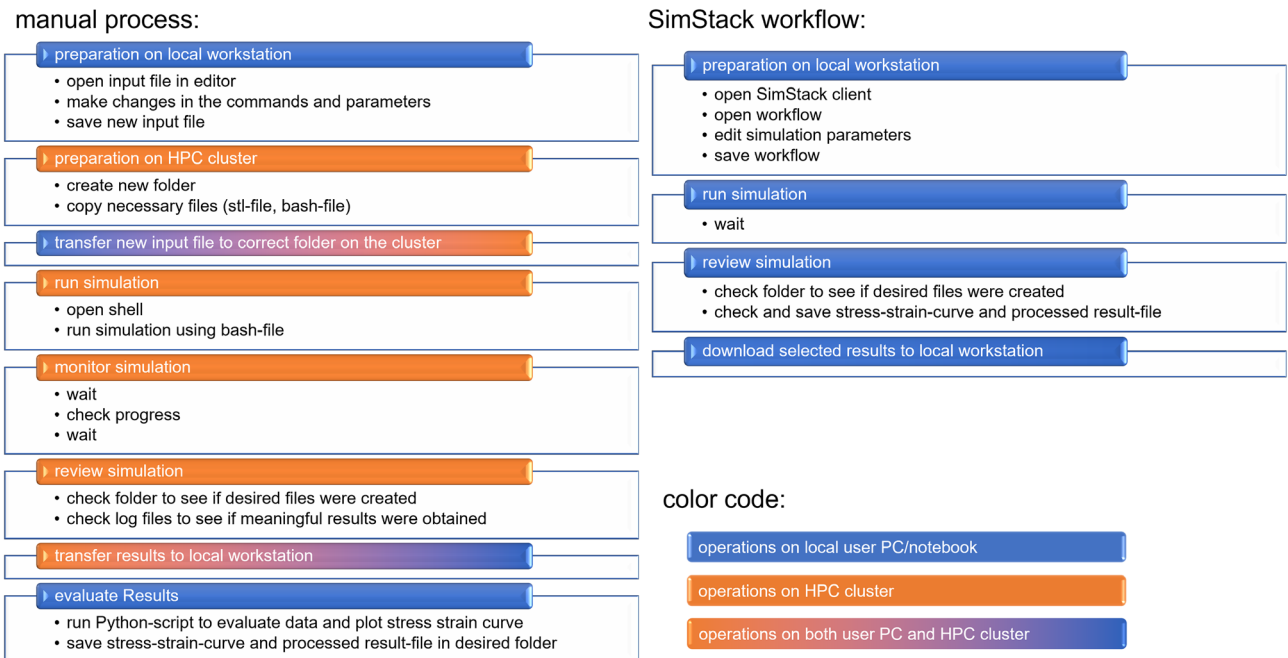

Figure 4. Comparison of necessary action steps to change and run a simulation job without (left) and with (right) workflow environment.

seen that there are subtle differences in the slope of the stress–strain curve depending on the number of hard particles. Extension of this analysis is readily possible by reusing the exact workflow in the SimStack client with more or different parameters.

2.5. Computational Details

Comprehensive documentation and guidance for running the workflows, connecting to HPC clusters, and installing

SimStack are accessible online.Details about the WaNos and their specific steps for executing the workflows can be found at GitHub.[16] This repository offers insight into the workflow creation, including scripts and configuration files necessary for the simulations. Further links are compiled in the Supporting Information.

2.5.1. Employing Workflows for Rapid Prototyping

As mentioned in the introduction, the workflow approach is not only suitable for finally developed simulation approaches for different use cases. Workflows, as lined out in this work, are also helpful in developing simulation approaches. Changes beyond the exposed parameters must be made to the input script to test different commands, insertion strategies, or pressing geometries in the pressing simulation presented here. Such interventions can smoothly be integrated into the process.

The WaNo is locally saved on the user client. Components like the stl files or the createInput.py can be changed or edited locally. Changes take effect immediately when the WaNo is loaded in the SimStack client. WaNos can also be reloaded during a running SimStack session simply by dragging and dropping. Combined with a simple workflow like the example in Figure 3a, this is a very effective setup to prototype, test, adapt, and improve a simulation approach.

In this respect, another strong feature of the SimStack environment is the easily adjustable HPC connection. This allows the selection of the HPC to be quickly adapted to the requirements of the individual simulation job. Particularly in the development process of simulation approaches, it can be very helpful to be able to switch quickly and easily to higher performance clusters when the complexity of the simulation increases.

3. Conclusion

This study explores how scientific workflows can transform digital materials science and engineering. We used the SimStack framework to integrate the DEM via the LIGGGHTS open-source library. SimStack makes it easy to develop and execute complex computational simulations. Our results show that SimStack speeds up research and development, enhances reproducibility, and encourages interdisciplinary collaboration. The LIGGGHTS WaNo allows us to generate customizable scripts from templates, which makes it easier for researchers and experimentalists without extensive computational backgrounds. This approach was successful in powder pressing simulation and has the potential for broader adoption in various domains.

Our findings show many advantages to integrating workflow automation from the beginning of simulation development. It simplifies the execution of complex simulations on HPC clusters, democratizes access to advanced modeling techniques, and promotes knowledge sharing across disciplines. Dividing simulation and analysis into distinct WaNos enhances the clarity and modularity of workflows, optimizes computational resources, and makes it easier to prototype simulation approaches.

SimStack and similar platforms are promising for the material science/engineering community. We can accelerate the development of new materials and technologies by fostering open

collaboration and sharing workflow templates. Exploring AI and machine learning integration within workflow environments could unlock new paradigms in predictive modeling and materials discovery. As computational workflows become increasingly integral to research and development, platforms like SimStack are poised to play a pivotal role in shaping the future of materials engineering, making sophisticated simulations more accessible, reproducible, and collaborative than ever before.

4. Experimental Section

LIGGGHTS Code: LIGGGHTS-PUBLIC 3.8.0 was used for the calculations. Relevant commands and parameters specific to LIGGGHTS used in the example calculations are summarized in **Table 1**. The square press stamps have an edge length of 20 m. Thus, the maximum force equals a mechanical stress of 100 MPa. The press stamps are loaded and moved using the fix style mesh/surface/stress/servo, for the die mesh/surface was used. For the interaction of the walls and particles, wall/gran was used.

DataDive: The DataDive WaNo basically calculates the thickness of the powder bed and the stress on the upper press stamp. The thickness t is determined according to Equation (1), where z_u and z_l are the positions of the upper and lower stamp recorded in forces_inputs.yml, respectively.

$$t = z_u - z_l \tag{1}$$

Stress σ is calculated from force F in forces_inputs.yml according to Equation (2).

$$\sigma = \frac{F}{4 \times 10^6} \tag{2}$$

This creates value pairs from thickness and stress. Thickness decreases linearly during the pressing phase as the stamp is moved with constant velocity. The stress is zero until the moment of contact when the stamp hits the powder bed. To determine this moment, the specified contact

Table 1. Relevant LIGGGHTS commands and parameters for example calculations.

Command/parameter	Value
Units	SI
Atom style	Sphere
Neighbor	0.002 bin
Pair style	Gran model hertz tangential history
Timestep	0.0001
Young's modulus hard particles	10^{10} Pa
Young's modulus soft particles	10^7 Pa
Poisson's ratio hard particles	0.5
Poisson's ratio soft particles	0.3
Coeff. of restitution hard–hard	0.9
Coeff. of restitution hard–soft	0.3
Coeff. of restitution soft–soft	0.07
Radii hard particles	0.2, 0.3, 0.4, 0.45, 0.5 m
Radius soft particles	0.2 m
Number share hard particles	0.091, 0.18, 0.363, 0.181, 0.091
Number share soft particles	0.091
Density	4 kg m^{-3}
Target value force	40×10^9 N

pressure σ_0 is used. Value pairs with $\sigma < \sigma_0$ are neglected. The thickness at σ_0 is defined as initial thickness t_0. Strain ε is then calculated according to Equation (3). It should be noted that strain is intentionally positive so that the stress–strain curve lies in the first quadrant.

$$\varepsilon = \frac{t_0 - t}{t_0} \tag{3}$$

Supporting Information

Supporting Information is available from the Wiley Online Library or from the author.

Conflict of Interest

The authors declare no conflict of interest.

Data Availability Statement

The data that support the findings of this study are openly available in Zenodo at https://zenodo.org/doi/10.5281/zenodo.11517338, reference number 11517339. For instructions on connecting to HPC resources and leveraging the full capabilities of SimStack for pressing simulations, interested parties should visit https://workflows.material-digital.de/. This platform provides various workflows and simulations relevant to materials science, offering users a comprehensive toolkit for their research needs. Additionally, https://simstack.readthedocs.io/ is a valuable resource for installing and utilizing SimStack, including step-by-step guides and troubleshooting tips to ensure users can navigate the software effectively. These resources collectively facilitate a seamless integration of simulation, data processing, and evaluation within the digital materials science domain, promoting efficient and collaborative research endeavors.

[1] T. Oinn, M. Addis, J. Ferris, D. Marvin, M. Senger, M. Greenwood, T. Carver, K. Glover, M. R. Pocock, A. Wipat, P. Li, *Bioinformatics* **2004**, *20*, 3045.

[2] P. Di Tommaso, M. Chatzou, E. W. Floden, P. P. Barja, E. Palumbo, C. Notredame, *Nat. Biotechnol.* **2017**, *35*, 316.

[3] M. Uhrin, S. P. Huber, J. Yu, N. Marzari, G. Pizzi, *Comput. Mater. Sci.* **2021**, *187*, 110086.

[4] 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.

[5] T. Mayeshiba, H. Wu, T. Angsten, A. Kaczmarowski, Z. Song, G. Jenness, W. Xie, D. Morgan, *Comput. Mater. Sci.* **2017**, *126*, 90.

[6] J. Janssen, S. Surendralal, Y. Lysogorskiy, M. Todorova, T. Hickel, R. Drautz, J. Neugebauer, *Comput. Mater. Sci.* **2019**, *163*, 24.

[7] C. R. C. Rêgo, J. Schaarschmidt, T. Schlöder, M. Penaloza-Amion, S. Bag, T. Neumann, T. Strunk, W. Wenzel, *Front. Mater.* **2022**, *9*.

[8] J. Schaarschmidt, J. Yuan, T. Strunk, I. Kondov, S. P. Huber, G. Pizzi, L. Kahle, F. T. Bölle, I. E. Castelli, T. Vegge, F. Hanke, T. Hickel, J. Neugebauer, C. R. C. Rêgo, W. Wenzel, *Adv. Energy Mater.* **2022**, *12*, 2102638.

[9] M. W. Thompson, J. B. Gilmer, R. A. Matsumoto, C. D. Quach, P. Shamaprasad, A. H. Yang, C. R. Iacovella, C. McCabe, P. T. Cummings, *Mol. Phys.* **2020**, *118*, e1742938.

[10] C. Goble, S. Cohen-Boulakia, S. Soiland-Reyes, D. Garijo, Y. Gil, M. R. Crusoe, K. Peters, D. Schober, *Data Intell.* **2020**, *2*, 108.

[11] C. Ramírez-Aragón, J. Ordieres-Meré, F. Alba-Elías, A. González-Marcos, *Materials* **2018**, *11*, 11.

[12] T. Leps, C. Hartzell, *Mater. Res. Express* **2021**, *8*, 085701.

[13] A. Podlozhnyuk, S. Pirker, C. Kloss, *Comput. Part. Mech.* **2017**, *4*, 101.

[14] B. Blais, M. Lassaigne, C. Goniva, L. Fradette, F. Bertrand, *J. Comput. Phys.* **2016**, *318*, 201.

[15] C. J. Coetzee, *Powder Technol.* **2017**, *310*, 104.

[16] B. Mieller, C. Ricardo, Kit-workflows/know-now: know-now, https://zenodo.org/doi/10.5281/zenodo.11517338 (accessed: April 2024).