

BattMoApp: A Web-Based application for running cell-level battery simulations

Lorena Hendrix¹, Oscar Bolzinger¹, Simon Clark¹, and Eibar Flores¹

¹ SINTEF Industry, Norway

DOI: [10.xxxxxx/draft](https://doi.org/10.xxxxxx/draft)

Software

- [Review](#)
- [Repository](#)
- [Archive](#)

Editor: [Open Journals](#)

Reviewers:

- [@openjournals](#)

Submitted: 01 January 1970

Published: unpublished

License

Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License ([CC BY 4.0](#)).

Summary

BattMoApp is a web-based application built upon the command-line based battery modelling software, BattMo. It features a user-friendly graphical interface that simplifies the simulation of battery cells. The development of BattMoApp has been centered on accessibility, intuitiveness, and usability, with the aim of making it a practical and valuable tool for both educational and research purposes in the battery field. Its design allows users to simulate, obtain, analyze, and compare results within just a few minutes. While BattMoApp leverages a small yet crucial portion of BattMo's capabilities, its intuitive and explanatory design also makes it an ideal starting point for those looking to explore the more comprehensive and complex BattMo software.

Statement of need

The Battery Modelling Toolbox (BattMo) is a framework for continuum modeling of electrochemical devices. Built primarily in Matlab, it offers a pseudo X-dimensional (PXD) framework for the Doyle-Fuller-Newman model of lithium-ion battery cells. Additionally, extensions for other battery chemistries and hydrogen systems are in development. BattMo provides a flexible framework for creating fully coupled electrochemical-thermal simulations of electrochemical devices using 1D, 2D, or 3D geometries. Besides the Matlab toolbox, the framework is also being developed in Julia to leverage increased simulation speed and the non-proprietary nature of Julia.

The primary objective of BattMoApp is to overcome common barriers associated with using battery modelling software. These barriers often arise from resource limitations, as many available software options are proprietary. Another challenge could be the complexity of using the software, whether due to a complicated graphical user interface or the difficulty of coding with command-line based software. While BattMo is open-source, its command-line interface may pose a hurdle for experimental researchers not familiar with coding, seeking to complement their lab work with simulations. BattMoApp, offering an accessible and intuitive graphical user interface, has the potential to reduce these barriers.

BattMoApp builds upon the P2D model implemented in the Julia version of BattMo. The development of BattMoApp has focused on accessibility, intuitiveness, and usability. Users can quickly and easily obtain results using the default input parameter sets available in the application or input their own values in a straightforward manner. The results can be easily analyzed and compared using the predefined plots that can handle multiple sets of simulation results. Users can also download their results and later upload them back into the application to review. Furthermore, significant effort has been made to ensure the parameters are realistic for both computational research and lab use, making it easier for experimentalists to fulfill the necessary inputs.

Another important aspect, besides accessibility, intuitiveness, and usability, is interoperability. To ensure that the data used in the simulation is inter-operable, the selected data format adheres to the FAIR principles and the 5-star open data guidelines. The data entered by the user is automatically formatted into a JSON Linked Data (LD) format which includes all the semantic metadata along with the actual data. This semantic data connects the actual data to the ontology documentations, EMMO and BattINFO, which contain descriptions of the linked data definitions. With this, the JSON LD format eliminates any confusion about definitions. If the user wishes to publish their results, they can include the JSON LD file in their publication, allowing anyone seeking to replicate the results to simply upload the JSON LD file into BattMoApp and obtain the anticipated results.

The documentation of BattMoApp includes an overview on what the application has to offer and a troubleshooting section that provides insights into the relationship between input parameters and results. As BattMoApp is designed to be informative and explanatory, it can also be a powerful tool for educational purposes, helping students understand batteries, battery modelling, and the impact of material and cell design parameters on battery cell performance.

Technical setup

The application consists of two main components: the graphical user interface (GUI), which includes the frontend, a database, and the backend that provides the frontend's functionality, and the application programming interface (API) that runs the BattMo software in the background. These two components are isolated from each other, each running in its own Docker container.

BattMo GUI

The frontend is Python-based and developed using the Streamlit framework. Streamlit was chosen because it is easy to use and does not require experience with JavaScript and CSS, while still offering the flexibility needed for BattMoApp. The backend, also written in Python, supports the frontend's functionality. The database that supports both the frontend and backend is created using the sqlite3 Python package.

BattMo API

Integrating Julia, a pre-compiled language, with Python, a runtime language, to form a smoothly running and stable application turned out to be complex. Therefore, a Julia-based API was created and containerized within a separate Docker container, isolating it from the BattMo GUI. This separation ensures that the Julia and Python components do not interfere with each other. The framework used for creating the API is Genie. Within the BattMo API's docker image a system image of BattMo's pre-compilation is created to ensure an instantaneous API response.

Examples

The following figures show screenshots of the 'Simulation' page and the 'Results' page of the application. On the 'Simulation' page, the input parameters can be defined, the cell geometry can be visualized, and a simulation can be initiated. On the 'Results' page, the results of the simulation can be visualized using the predefined plots. For the shown results in figure [Figure 2](#), two simulations were run using the default parameters sets from Chen et al. One with smaller negative and positive electrode coating thicknesses, and one with bigger negative and positive electrode coating thicknesses.

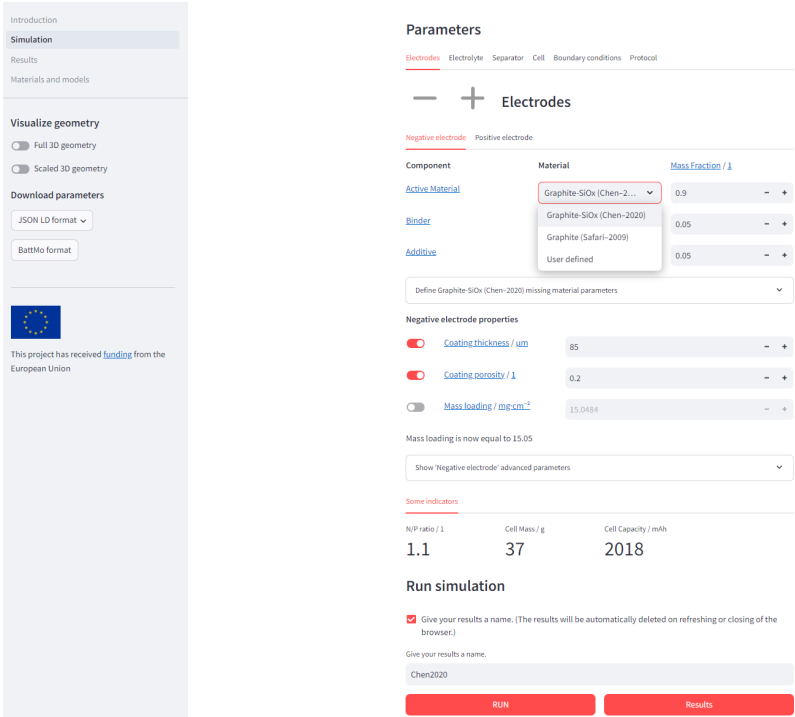


Figure 1: A screenshot of the Simulation page.

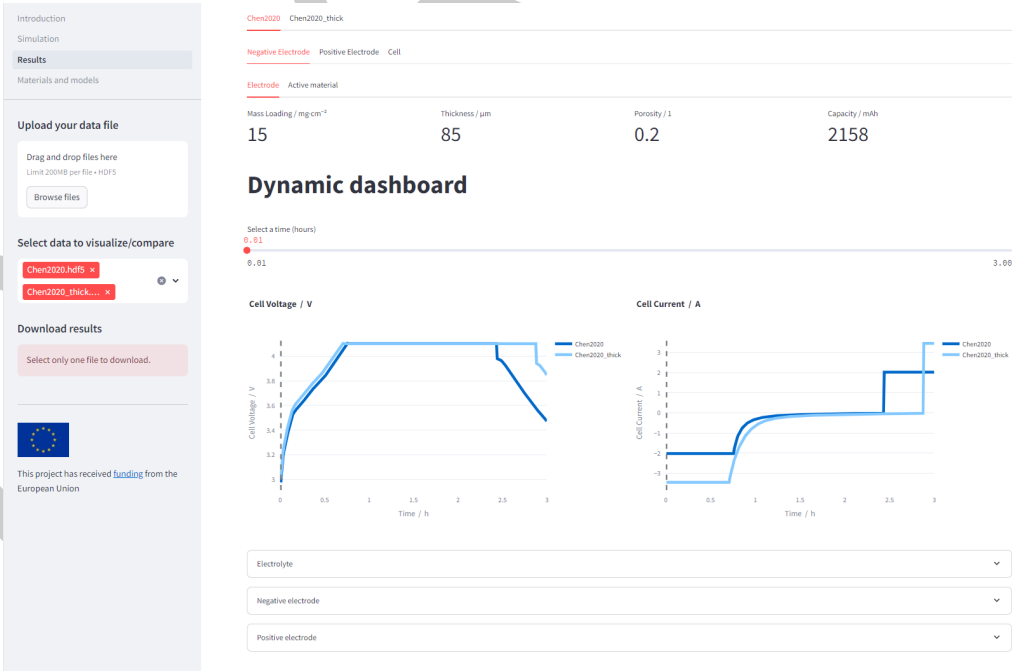


Figure 2: A screenshot of the Results page.

84 **Future work**

85 While BattMoApp has reached a matured state and provides a valuable design for P2D
86 simulations, there are still endless possibilities in its development. The development of

BattMoApp will go alongside the development of BattMo.jl so that in the future additional simulation models as well as other features like parametrization can be integrated in the application. BattMoApp will continue seeking feedback from its target audiences to keep improving its usability and practicality. Furthermore, more effort will be put into improving its performance and stability when having multiple users.

Installation and contribution

BattMoApp can easily be used online at the following address: `app.batterymodel.com`. Furthermore, it can be locally installed using Docker. The Docker images and a detailed instruction on how to install BattMoApp locally can be found in the Github repository.

Acknowledgements

BattMo has received funding from the European Union's Horizon 2020 innovation program under grant agreement numbers:

- 875527 HYDRA
- 957189 BIG-MAP

References