The Journal of Open Source Software

TulipaEnergyModel.jl: A Modelling Framework Breaking the Tradeoff Between Fidelity and Computational Load

Abel Soares Siqueira[1], **Diego A. Tejada-Arango**[2,6], **Grigory Neustroev**[3], **Juha Kiviluoma**[4], **Lauren Clisby**[2], **Maaike Elgersma**[3], **Ni Wang**[2], **Suvayu Ali**[1], **Zhi Gao**[5], **Mathijs M. de Weerdt**[3], and **Germán Morales-España**[2,3]

1 Netherlands eScience Center, Netherlands **2** TNO - Netherlands Organisation for Applied Scientific Research, Netherlands **3** Delft University of Technology, Netherlands **4** Nodal Tools, Finland **5** Utrecht University, Netherlands **6** Universidad Pontificia Comillas

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Marlin Arnz

Reviewers:

- @DavidPozo
- @trulsf

Submitted: 20 October 2025
Published: unpublished

Summary

`TulipaEnergyModel.jl` is a modelling framework for analysing investment and operational decisions of future energy systems through capacity expansion and dispatch optimisation. `TulipaEnergyModel.jl` is the main package of the Tulipa Energy ecosystem, and it is developed in Julia (Bezanson et al., 2017) using JuMP.jl (Lubin et al., 2023). As a framework, Tulipa formulates models entirely based on input data. This allows users to analyse virtually any system using the generalised building blocks – production, consumption, conversion, storage, and transport. TulipaEnergyModel.jl focuses on model quality and efficient implementation, allowing it to break the tradeoff between model fidelity and computational load through: tighter MIP formulations; exact LP reformulations with fewer constraints and variables; more accurate LP approximations; and flexible model fidelity across temporal, technological, and spatial dimensions.

Statement of Need

Existing models and frameworks in Energy System Optimisation include EnergyModelsX (Hellemo et al., 2024), PowerModels (Coffrin et al., 2018), SpineOpt (Ihlemann et al., 2022), Sienna (Lara et al., 2021), GenX (Bonaldo et al., 2025), PyPSA (Brown et al., n.d.), and Calliope (Pfenninger & Pickering, 2018). However, they run into computational limits when solving large-scale problems and must resort to (over)simplifying the model to reduce computational burden. The common misconception is that the only strategy to speed up solving times without sacrificing model fidelity is through faster solvers or computers. However, the strategy that is widely overlooked is improving the quality of the mathematical formulations, which increases model fidelity while simultaneously solving faster than standard formulations. This insight inspired the development of TulipaEnergyModel.jl, with the core philosophy of advancing the state-of-the-art in formulation quality by: 1) lowering computational cost while maintaining model fidelity, by reducing the problem size (Tejada-Arango et al., 2025), and by creating tighter mixed-integer programs (MIP) (Morales-España et al., 2013). 2) increasing model fidelity without extra computational cost, e.g., by developing more accurate linear programming (LP) approximations (Elgersma et al., 2025; Gentile et al., 2016; Morales-España et al., 2022). Finally, 3) balancing computational burden with adaptive/flexible model fidelity, i.e., having different levels of detail in various parts of the model, in the temporal (Gao et al., 2025), technological (Morales-España et al., 2022) and spatial dimensions. These modelling strategies offer significant computational benefits, especially when handling large-scale problems:

covering a continent with multiple energy carriers, and optimising over decades while maintaining hourly resolution for key aspects (e.g., renewable generation). TulipaEnergyModel.jl had to be developed from scratch to be able to include all of these modelling breakthroughs, since they alter the foundation and structure of the model. Below, we present some of core modelling and software design innovations.

Modelling Innovations

Two of the main innovations include:

1. Fully flexible temporal resolution (Gao et al., 2025) for the assets/ flows,
2. Direct connections between assets (Tejada-Arango et al., 2025).

The following example illustrates these concepts:

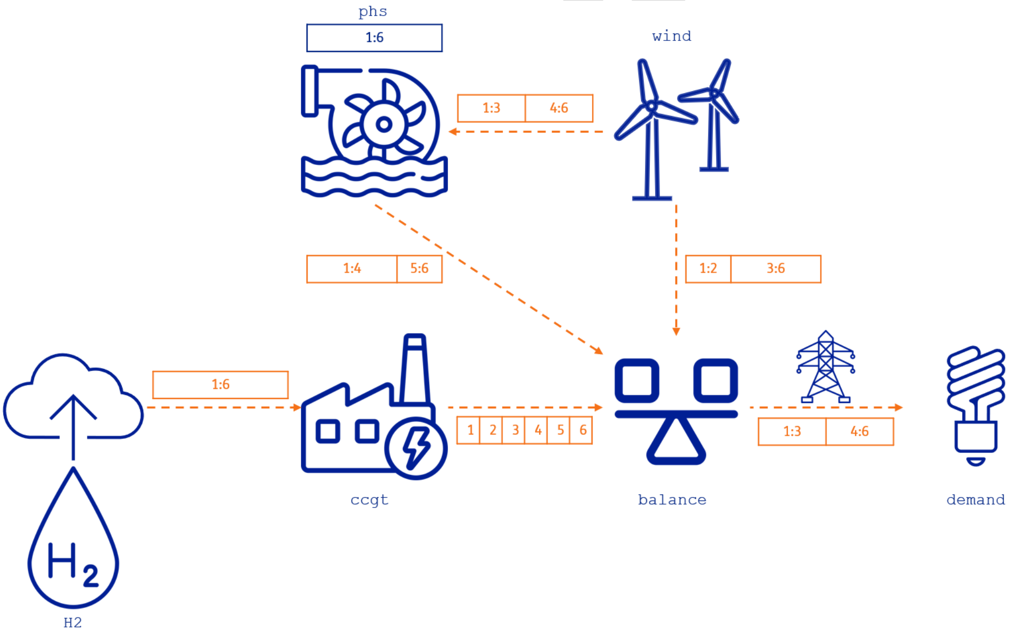

Figure 1: Example of network with flexible resolution of assets and flows

For the fully flexible temporal resolution, consider the 6-hour duration of this system. The flow between "H2" and "ccgt" has a resolution of 6 hours, while the flow between "ccgt" and "balance" is 1 hour. The resolution from "wind" to "phs" is 3 hours, and from "phs" to "balance" is irregular, a 4-hour block followed by a 2-hour block. Tulipa allows different temporal resolutions throughout the model, thereby reducing the number of variables and constraints. This feature can drastically speed up solving with little loss in accuracy (Gao et al., 2025).

For the direct connection between assets, the storage "phs" is directly connected to the "wind" to charge, and to "balance" to discharge. This for direct connection between assets completely avoids intermediate elements (connections/nodes), thereby eliminating unnecessary variables and constraints. Thus, accelerating solving times without any loss of accuracy (Tejada-Arango et al., 2025).

TulipaEnergyModel.jl is fundamentally focused on high-quality mathematical formulations. The model also includes other key features such as seasonal storage modelling using representative periods (Neustroev et al., 2025; Tejada-Arango et al., 2018), tight and compact MIP formulations for storage (Elgersma et al., 2025), unit commitment (Morales-España et al., 2013), and compact formulations for multi-year investment (Wang & Morales-España, 2025; Wang & Tejada-Arango, 2025).

Software Design Innovations

To accommodate flexible temporal resolution for assets and flows, many variables have a "time block" component instead of a "time step", and many variables and constraints have sparse indices. To explain how this is implemented, revisit the system from Figure 1. In an hourly implementation, there would be variables such as $f_{(\text{H2},\text{ccgt}),t}$, $f_{(\text{phs},\text{balance}),t}$ for $t = 1, \dots, 6$. Instead, looking at the time blocks of each flow, there are $f_{(\text{H2},\text{ccgt}),1:6}$, $f_{(\text{phs},\text{balance}),1:4}$, and $f_{(\text{phs},\text{balance}),5:6}$. In other words, this flow variable could be defined as

$$f_{e,b} \qquad \forall e \in E, b \in B(e),$$

where E is the set of edges of the graph, and $B(e)$ is the set of time blocks for each edge.

Similarly, some constraints and expressions have their own time blocks (e.g., the balance constraints), and obtaining the coefficients of the flows involves matching indices and computing the intersections between time blocks. Defining variables like this in JuMP leads to sparse indices, which are cumbersome and potentially slow. It can also create performance pitfalls, such as the "sum-if problem", when trying to determine the coefficients of each flow in each balance constraint.

To improve the storage of these objects, Tulipa uses a tabular format to collect the indices of each variable, constraint, and expression. Each row of the table stores the corresponding set of indices for each JuMP object, and the JuMP objects themselves are stored in a linearised way in an array. Returning to the example system, the first three rows of the flow variable defined above are shown in Table 1.

Table 1: "Example of linearised tabular indices of the `var_flow` table"

id	from asset	to asset	time block start	time block end
1	H2	ccgt	1	6
2	phs	balance	1	4
3	phs	balance	5	6

Storing the indices in tables avoids the sparse storage of JuMP objects, which can instead be stored in an array with their positions matching the IDs in the index tables. Figure 2 illustrates this method of storage.

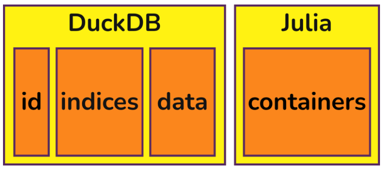

Figure 2: Representation of the storage of indices and JuMP objects

Another key design decision in Tulipa is handling data using a DuckDB (Raasveldt & Muehleisen, n.d.) connection - from processing input data, to generating internal tables for model creation, to storing outputs. This enables manipulating different data formats using DuckDB's capabilities (instead of Julia's), as well as decreasing data movement and duplication.

When creating variables and constraints, Tulipa can read the DuckDB tables row by row, reducing memory use compared to using an explicit Julia table, such as DataFrames.

Furthermore, using DuckDB for everything allows cleaner separation of the pipeline into (i) data ingestion and manipulation, (ii) indices creation, and (iii) model creation. Figure 3 summarises how Tulipa interacts with the DuckDB connection.

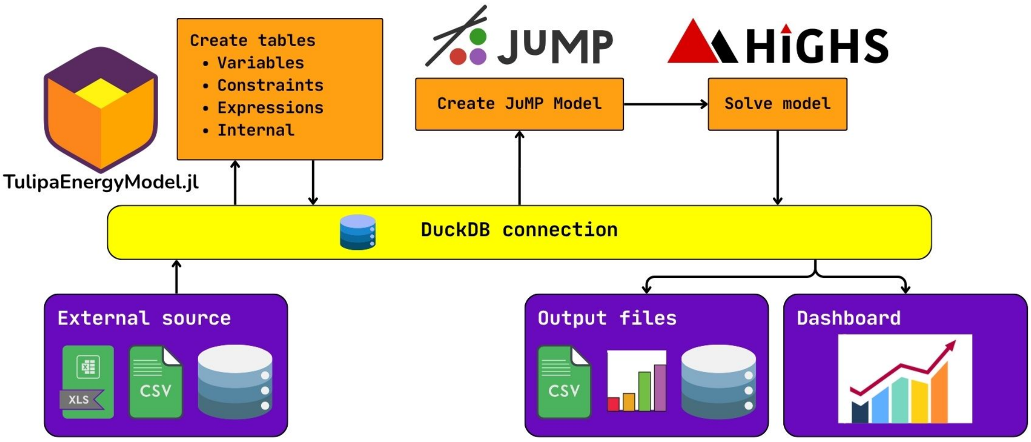

Figure 3: Overview of Tulipa's integration with DuckDB

First, users can prepare their data using any tools they prefer before loading it into the DuckDB connection. Then, the indices are created from the data in the DuckDB connection and saved back into it. While this step currently occurs in Julia, many of the operations use SQL and could potentially be moved outside Julia. Finally, the JuMP pipeline begins by reading the indices table and complementing the data as necessary, then creates the JuMP objects and generates the complete model, which is sent to the solver. Tulipa is tested and benchmarked using the HiGHS (Huangfu & Hall, 2017) solver, but other MIP solvers accepted by JuMP can be used as well.

Acknowledgements

We thank Oscar Dowson for the initial suggestion to use DataFrames, which motivated us to use tabular storage for all indices.

This publication is part of the project NextGenOpt ESI.2019.008, which is financed by the Dutch Research Council (NWO) and supported by the Netherlands eScience Center NLeSC C 21.0226. This project was also supported by TNO's internal R&D. This research is also (partially) funded by the European Climate, Infrastructure and Environment Executive Agency under the European Union's HORIZON Research and Innovation Actions under grant agreement no. 101095998.

References

Bezanson, J., Edelman, A., Karpinski, S., & Shah, V. B. (2017). Julia: A fresh approach to numerical computing. *SIAM Review*, *59*(1), 65–98. https://doi.org/10.1137/141000671

Bonaldo, L., Chakrabarti, S., Cheng, F., Ding, Y., Jenkins, J., Luo, Q., Macdonald, R., Mallapragada, D., Manocha, A., Mantegna, G., Morris, J., Patankar, N., Pecci, F., Schwartz, A., Schwartz, J., Schivley, G., Sepulveda, N., & Xu, Q. (2025). *GenX* (Version 0.4.4). https://doi.org/10.5281/zenodo.10846070

Brown, T., Hörsch, J., Hofmann, F., Neumann, F., Zeyen, L., Syranidis, C., Frysztacki, M., Schlachtberger, D., Glaum, P., & Parzen, M. (n.d.). *PyPSA: Python for Power System Analysis* (Version 0.35.1). https://doi.org/10.5334/jors.188

Coffrin, C., Bent, R., Sundar, K., Ng, Y., & Lubin, M. (2018). PowerModels.jl: An open-source framework for exploring power flow formulations. *2018 Power Systems Computation Conference (PSCC)*, 1–8. https://doi.org/10.23919/PSCC.2018.8442948

Elgersma, M. B., Morales-España, G., Aardal, K. I., Helistö, N., Kiviluoma, J., & Weerdt, M. M. de. (2025). *Tight MIP formulations for optimal operation and investment of storage including reserves*. https://arxiv.org/abs/2411.17484

Gao, Z., Gazzani, M., Tejada-Arango, D. A., Soares Siqueira, A., Wang, N., Gibescu, M., & Morales-España, G. (2025). Fully flexible temporal resolution for energy system optimization. *Applied Energy*, *396*, 126267. https://doi.org/10.1016/j.apenergy.2025.126267

Gentile, C., Morales-España, G., & Ramos, A. (2016). A tight MIP formulation of the unit commitment problem with start-up and shut-down constraints. *EURO Journal on Computational Optimization*, *5*(1), 177–201. https://doi.org/10.1007/s13675-016-0066-y

Hellemo, L., Bødal, E. F., Holm, S. E., Pinel, D., & Straus, J. (2024). EnergyModelsX: Flexible energy systems modelling with multiple dispatch. *Journal of Open Source Software*, *9*(97), 6619. https://doi.org/10.21105/joss.06619

Huangfu, Q., & Hall, J. A. J. (2017). Parallelizing the dual revised simplex method. *Mathematical Programming Computation*, *10*(1), 119–142. https://doi.org/10.1007/s12532-017-0130-5

Ihlemann, M., Kouveliotis-Lysikatos, I., Huang, J., Dillon, J., O'Dwyer, C., Rasku, T., Marin, M., Poncelet, K., & Kiviluoma, J. (2022). SpineOpt: A flexible open-source energy system modelling framework. *Energy Strategy Reviews*, *43*, 100902. https://doi.org/10.1016/j.esr.2022.100902

Lara, J. D., Barrows, C., Thom, D., Krishnamurthy, D., & Callaway, D. (2021). PowerSystems.jl — A power system data management package for large scale modeling. *SoftwareX*, *15*. https://doi.org/10.1016/j.softx.2021.100747

Lubin, M., Dowson, O., Dias Garcia, J., Huchette, J., Legat, B., & Vielma, J. P. (2023). JuMP 1.0: Recent improvements to a modeling language for mathematical optimization. *Mathematical Programming Computation*. https://doi.org/10.1007/s12532-023-00239-3

Morales-España, G., Latorre, J. M., & Ramos, A. (2013). Tight and compact MILP formulation for the thermal unit commitment problem. *IEEE Transactions on Power Systems*, *28*(4), 4897–4908. https://doi.org/10.1109/TPWRS.2013.2251373

Morales-España, G., Martínez-Gordón, R., & Sijm, J. (2022). Classifying and modelling demand response in power systems. *Energy*, *242*, 122544. https://doi.org/10.1016/j.energy.2021.122544

Neustroev, G., Tejada-Arango, D. A., Morales-Espana, G., & Weerdt, M. M. de. (2025). *Hull Clustering with Blended Representative Periods for Energy System Optimization Models*. arXiv. https://doi.org/10.48550/arXiv.2508.21641

Pfenninger, S., & Pickering, B. (2018). Calliope: A multi-scale energy systems modelling framework. *Journal of Open Source Software*, *3*(29), 825. https://doi.org/10.21105/joss.00825

Raasveldt, M., & Muehleisen, H. (n.d.). *DuckDB*. https://github.com/duckdb/duckdb

Tejada-Arango, D. A., Domeshek, M., Wogrin, S., & Centeno, E. (2018). Enhanced representative days and system states modeling for energy storage investment analysis. *IEEE Transactions on Power Systems*, *33*(6), 6534–6544. https://doi.org/10.1109/TPWRS.2018.2819578

Tejada-Arango, D. A., Kiviluoma, J., & Morales-España, G. (2025). Debunking the speed-fidelity trade-off: Speeding-up large-scale energy models while keeping fidelity. *International*

Journal of Electrical Power & Energy Systems, *168*, 110674. https://doi.org/10.1016/j.ijepes.2025.110674

Wang, N., & Morales-España, G. (2025). *Vintage-based formulations in multi-year investment modelling for energy systems*. https://arxiv.org/abs/2505.00379

Wang, N., & Tejada-Arango, D. A. (2025). *Discounting approaches in multi-year investment modelling for energy systems*. https://arxiv.org/abs/2504.21709