otoole: OSeMOSYS Tools for Energy Work

Trevor Barnes ⓘ [1*¶] **and Will Usher** ⓘ [2*]

1 School of Sustainable Energy Engineering, Simon Fraser University, Vancouver, Canada **2** Energy Technology, KTH Royal Institute of Technology, Stockholm, Sweden ¶ Corresponding author * These authors contributed equally.

Summary

Data handling for energy system optimisation models is a necessary but tedious task. Depending on the workflow, user skill level, and model implementation, the data interfacing requirements can be significantly different. OSeMOSYS Tools for Energy Work, or otoole, is a Python package providing OSeMOSYS energy modellers with options to use different input data formats, visualize and validate input data, and process result data. otoole exposes three different input data formats to the user, can process result data from four popular solvers, and is designed to be modular and extensible to enable interoperability between OSeMOSYS models developed in different programming languages.

Statement of need

The Open Source energy MOdelling SYStem, or OSeMOSYS (Howells et al., 2011), is a highly cited[1] and widely used open-source framework for conducting long-term energy system planning studies (Groissböck, 2019). The original, and still highly used, implementation of OSeMOSYS is formulated in the mathematical programming language, GNU MathProg. While MathProg (*GNU Linear Programming Kit, Version 4.65,* 2012) is open-source, fitting with the ethos of OSeMOSYS, it requires all data to be stored in a single large and unwieldy text file. Data edits to this file are tedious, error-prone, and difficult to integrate into automated workflows. Furthermore, model result files are often difficult to interface with unless processing work is first performed. In addition to the MathProg implementation, OSeMOSYS has also been formulated in PuLP, Pyomo, Julia JUMP and GAMS. A software gap exists to provide OSeMOSYS modellers with an easy way to work with different input data formats and process result files.

Compared to other open energy modelling frameworks that are based around a single (Python) implementation, such as PyPSA (Brown et al., 2018) and Calliope (Pfenninger & Pickering, 2018), the OSeMOSYS framework does not have standard data processing functions. For example, in PyPSA, users instantiate a `Network` object which has methods to import data (from various formats such as CSV, netCDF, and pypower), manipulate data, build and solve the model, and export results. In contrast, users of OSeMOSYS are required to prepare the input data and process results data themselves. otoole addresses the need in the OSeMOSYS community to standardize data handling functions across multiple different model implementations.

Utilities to facilitate data management in energy system models have existed for as long as the energy system modelling frameworks themselves. For example, see the proprietary solutions TIMES-VEDA and PLEXOS, or the open-source solutions pyam (Huppmann et al., 2021)

---

[1]A SCOPUS search for "TITLE-ABS-KEY (osemosys) AND LANGUAGE (english)" returned 81 results between 2018 to 2023. Moreover, the original OSeMOSYS paper (Howells et al., 2011) has over 440 citations in SCOPUS as of September 2023.

---

and Spine-Toolbox (Kiviluoma et al., 2022). These utilities offer some similar functionality to otoole but are often tailored to specific frameworks (in the case of proprietary solutions), focus on workflow management, or are concerned with result analysis rather than data processing. Given the flexible nature of OSeMOSYS, many different models exist with unique parameters, variables, and constraints. The OSeMOSYS community requires a tool to process input and result data regardless of model structure and implementation.

otoole supports three different input file formats: wide-format Excel files (pivoted on the year index), long-format CSV files, and GNU MathProg files. With otoole, users can convert between any of these formats to meet their skill level and workflow requirements. Furthermore, otoole can process result solution files from the open-source solvers GLPK (*GNU Linear Programming Kit, Version 4.65*, 2012) and CBC (Forrest et al., 2022), and the commercial solvers Gurobi (Gurobi Optimization, LLC, 2022) and CPLEX (International Business Machines Corporation, 2022), into tabulated CSV results.

Additionally, otoole can visualize and validate input data. Through the visualization function, users can create a reference energy system from their input data, as shown in Figure 1; this is a common step in energy modelling where a schematic is used to visualize the flow of energy. Moreover, if the input data follows a standardized naming scheme, the user can validate input data to quickly identify improperly named technologies and commodities and ensure energy flow paths are complete.
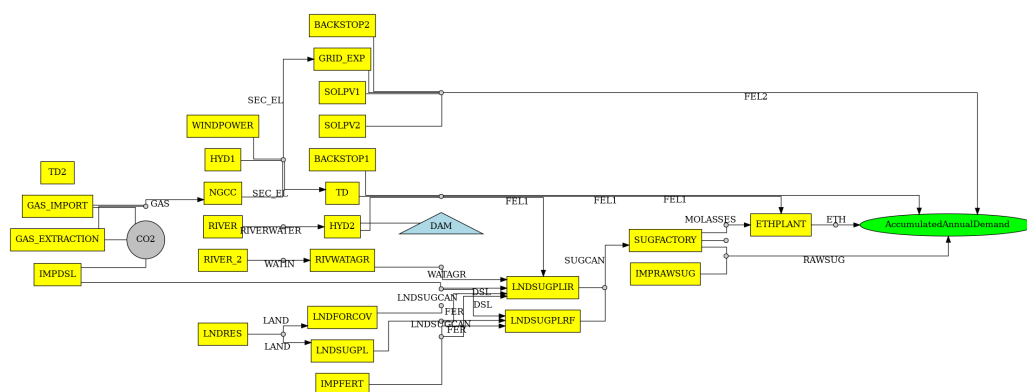

Figure 1: Reference Energy System Example

Publications

The development of otoole was originally mentioned in (Niet et al., 2021) through a discussion on consolidating and standardising scripts and data formats within the OSeMOSYS community. Since then, otoole has grown and been implemented in various published research projects. A paper on conducting global sensitivity analysis in the context of energy systems (Usher et al., 2023) uses otoole to convert data for hundreds of simultaneous OSeMOSYS model runs. OSeMOSYS Global (Barnes et al., 2022), an open-source global electricity system model generator, uses otoole in its workflow to programmatically create unique OSeMOSYS models based on user inputs. Finally, (Ramos et al., 2022) mentions otoole in a paper discussing how to use OSeMOSYS to perform climate, land, energy, and water system modelling.

Extensibility

While otoole was originally created to assist MathProg OSeMOSYS modellers, its functionality can easily extend to other implementations of OSeMOSYS, and any workflow that uses non-OSeMOSYS MathProg data files.

otoole follows a strategy design pattern (Gamma et al., 1994, p. 315). Two abstract base classes define how otoole reads and writes data, named ReadStrategy and WriteStrategy respectively. The ReadStrategy class reads model data or result data into an internal data structure (currently a dictionary of pandas DataFrames), with a ReadResults abstract class responsible for calculating intermediate results if required. The WriteStrategy class writes out data into a format specified by the user. A schematic of this design pattern is shown in Figure 2.
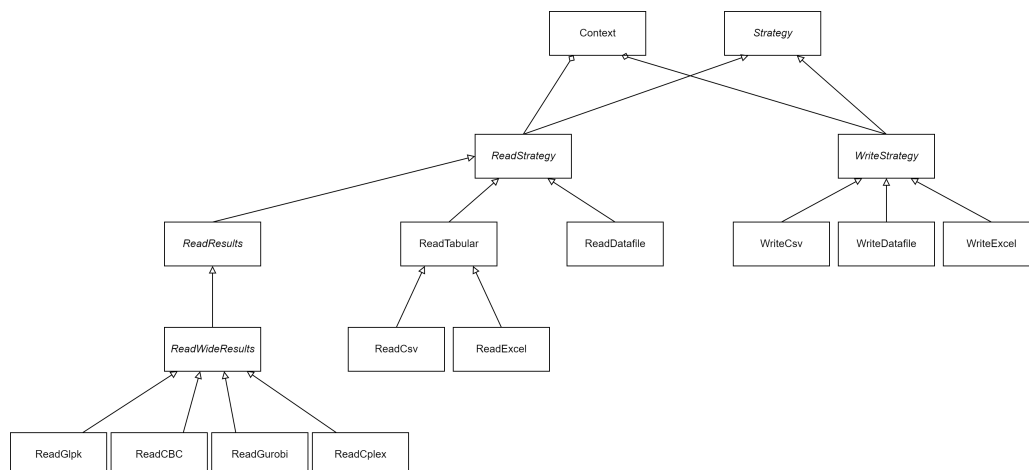

Figure 2: otoole Design Pattern

The advantage of designing otoole around a strategy pattern is that it allows for the easy addition of new file formats and model implementations. For example, there are existing OSeMOSYS implementations written in Python and GAMS, and new implementations may be created in the future. New reading and writing classes can easily add data conversion into and out of these formats. This is especially useful for benchmarking each implementation against one another to check for inconsistencies, measure performance, or to migrate OSeMOSYS models from one format to another. This same extensibility logic applies to the reading of results. For example, support for the new open-source solver, HiGHS (Huangfu & Hall, 2018), can be added through a class that inherits from ReadResults and implements reading logic specific to HiGHS.

When converting between formats, the user supplies otoole with a configuration file that describes the parameters, sets, and variables in the model. This allows otoole to work with any version of OSeMOSYS regardless of the model's parameters, sets, or variables. This is particularly useful for extensions of OSeMOSYS, such as the Climate, Land, Energy, and Water framework (Bazilian et al., 2011), which adds parameters to OSeMOSYS to represent other energy sectors. Moreover, describing the model through a configuration file allows otoole to work with any MathProg model file; not just OSeMOSYS models.

Installation and Example

otoole is published to the Python Packaging Index (PyPI) and can be installed via pip

```
pip install otoole
```

A sample repository, titled Simplicity, holds a simple OSeMOSYS model to demonstrate the functionalities of otoole. Instructions on the core functions of otoole, in addition to examples using otoole through the command line and through the Python API can be found on the documentation site.

References

Barnes, T., Shivakumar, A., Brinkerink, M., & Niet, T. (2022). OSeMOSYS Global, an open-source, open data global electricity system model generator. *Scientific Data*, *9*(1), 623. https://doi.org/10.1038/s41597-022-01737-0

Bazilian, M., Rogner, H., Howells, M., Hermann, S., Arent, D., Gielen, D., Steduto, P., Mueller, A., Komor, P., Tol, R. S. J., & Yumkella, K. K. (2011). Considering the energy, water and food nexus: Towards an integrated modelling approach. *Energy Policy*, *39*(12), 7896–7906. https://doi.org/10.1016/j.enpol.2011.09.039

Brown, T., Hörsch, J., & Schlachtberger, D. (2018). PyPSA: Python for Power System Analysis. *Journal of Open Research Software*, *6*(4, 1). https://doi.org/10.5334/jors.188

Forrest, J., Ralphs, T., Santos, H. G., Vigerske, S., Forrest, J., Hafer, L., Kristjansson, B., jpfasano, EdwinStraver, Lubin, M., Brito, S., Cristina, Saltzman, M., Pitrus, B., & MATSUSHIMA, F. (2022). *Coin-or/Cbc*. Zenodo. https://doi.org/10.5281/zenodo.6522795

Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1994). *Design patterns*. Addison Wesley. ISBN: 0201633612

GNU linear programming kit, version 4.65. (2012). https://www.gnu.org/software/glpk/glpk.html

Groissböck, M. (2019). Are open source energy system optimization tools mature enough for serious use? *Renewable and Sustainable Energy Reviews*, *102*, 234–248. https://doi.org/10.1016/j.rser.2018.11.020

Gurobi Optimization, LLC. (2022). *Gurobi Optimizer Reference Manual*. https://www.gurobi.com

Howells, M., Rogner, H., Strachan, N., Heaps, C., Huntington, H., Kypreos, S., Hughes, A., Silveira, S., DeCarolis, J., Bazillian, M., & Roehrl, A. (2011). OSeMOSYS: The Open Source Energy Modeling System: An introduction to its ethos, structure and development. *Energy Policy*, *39*(10), 5850–5870. https://doi.org/10.1016/j.enpol.2011.06.033

Huangfu, Q., & Hall, J. A. J. (2018). Parallelizing the dual revised simplex method. *Mathematical Programming Computation*, *10*(1), 119–142. https://doi.org/10.1007/s12532-017-0130-5

Huppmann, D., Gidden, M. J., Nicholls, Z., Hörsch, J., Lamboll, R. D., Kishimoto, P. N., Burandt, T., Fricko, O., Byers, E., Kikstra, J. S., Brinkerink, M., Budzinski, M., Maczek, F., Zwickl-Bernhard, S., Welder, L., Alvarez Quispe, E. F., & Smith, C. J. (2021). Pyam: Analysis and visualization of integrated-assessment and macro-energy scenarios [version 2; peer review: 3 approved]. In *Open Research Europe* (Vol. 1). https://doi.org/10.12688/openreseurope.13633.2

International Business Machines Corporation. (2022). *CPLEX Users Manual*. IBM. https://www.ibm.com/products/ilog-cplex-optimization-studio

Kiviluoma, J., Pallonetto, F., Marin, M., Savolainen, P. T., Soininen, A., Vennström, P., Rinne, E., Huang, J., Kouveliotis-Lysikatos, I., Ihlemann, M., Delarue, E., O'Dwyer, C., O'Donnel, T., Amelin, M., Söder, L., & Dillon, J. (2022). Spine toolbox: A flexible open-source workflow management system with scenario and data management. *SoftwareX*, *17*, 100967. https://doi.org/10.1016/j.softx.2021.100967

Niet, T., Shivakumar, A., Gardumi, F., Usher, W., Williams, E., & Howells, M. (2021). Developing a community of practice around an open source energy modelling tool. *Energy Strategy Reviews*, *35*, 100650. https://doi.org/10.1016/j.esr.2021.100650

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

Ramos, E. P., Sridharan, V., Alfstad, T., Niet, T., Shivakumar, A., Howells, M. I., Rogner, H., & Gardumi, F. (2022). Climate, land, energy and water systems interactions – from key concepts to model implementation with OSeMOSYS. *Environmental Science & Policy*, *136*, 696–716. https://doi.org/10.1016/j.envsci.2022.07.007

Usher, W., Barnes, T., Moksnes, N., & Niet, T. (2023). Global sensitivity analysis to enhance the transparency and rigour of energy system optimisation modelling. *Open Research Europe*, *3*, 30. https://doi.org/10.12688/openreseurope.15461.1