rswap – an R package for automated and command-based interaction with the SWAP4.2 model

NIBIO
NORWEGIAN INSTITUTE OF BIOECONOMY RESEARCH

Moritz Shore & Csilla Farkas

EGU24, 2024-04-15, Vienna
EGU24-8608ECS

v0.5.0

What?

SWAP (Soil-Water-Atmosphere-Plant) is a one-dimensional process-based model utilizing the Richards' equation to calculate water movement within a layered soil profile in the unsaturated (vadose) zone and upper sections of the saturated zone (groundwater), accounting for plant and atmospheric interaction [1,3]. Created in 1974, SWAP has become an important tool in its domain, and now celebrates a 50th anniversary in supporting soil-water research, engineering, and education worldwide [2,3].

SWAP 4.2, the latest version [1], takes in input data in the form of text files (Figure 1, top) and creates output in the same format, leaving it to the user to process results and modify input files. While the text-file approach is inherently flexible and adaptable to most use-cases, it requires users to invest significant amounts time into setting up their workflows and creates a "barrier of entry" for initial applications of the model.

rswap is an open-source R-package designed to interface with SWAP 4.2, allowing users a command-based interaction within the R-programming environment. By automating common tasks in model setup, calibration, and analysis, *rswap* simplifies and streamlines (Figure 1, bottom) user interaction while increasing accessibility. *rswap* accomplishes this using functional programming with flexible (user-accessible) base functions (translating text files, loading / converting data) which go on to support more advanced higher-level functions such as output analysis (Figures 2, 3) or running SWAP in parallel (Figure 4).

How?

Input manipulation

1. **Translate** SWAP parameters, tables, and lists *(into R-compatible dataframes)*
2. **Modify** dataframes *(using standard R-based operations)*
3. **Write** modified dataframes *(into SWAP compatible text files)*

Model Execution

1. **Detect** SWAP file paths *(paths to input files, executables, observed data)*
2. **Translate** SWAP parameters, tables, and lists *(into R-compatible dataframes)*
3. **Alter** parameter values to match user demands *(ie. paths, observed data, settings)*
4. **Re-write** SWAP files *(using modified parameter values)*
5. **Run** the model in the R-console *(or in parallel)*

Output Analysis

1. **Plotting** of model outputs / variables *(interactively, and/or with observed data)*
2. **Model performance** using goodness of fit statics *(from package hydroGOF)*
3. **Comparing** different model runs *(visually / statistically)*

Further Analysis

1. **Sensitivity Analysis** *(to output variables, performance indicators, Figure 5)*
2. **Scenario Analysis** *(under development)*
3. **Parameter Estimation** / Hard calibration / Autocalibration *(under development)*
4. **SWAPtools** integration *(under development)*
5. *...*

← More details on the website

moritzshore.github.io/rswap/

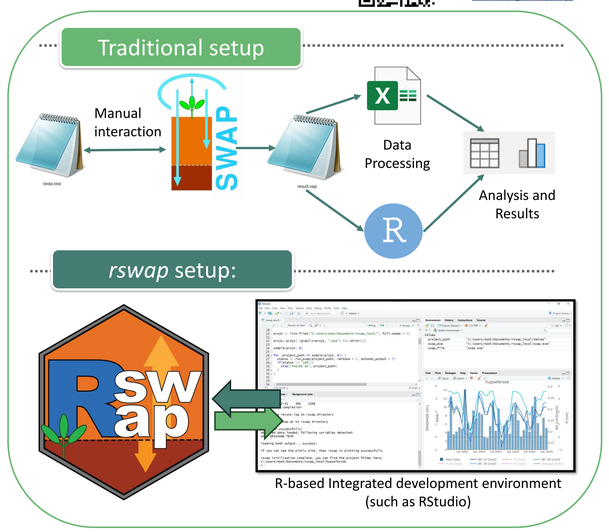

Figure 1: A visual comparison of a traditional SWAP workflow and the streamlined *rswap* workflow, entirely within the RStudio IDE.

```
rswap_plot_compare(project_path, variable = "WC", depth = 15)
```

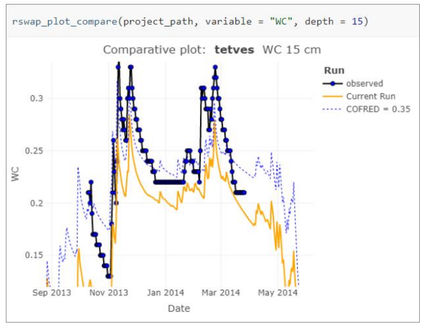

Figure 2: Comparison of two saved model runs with differing parameter values and the observed timeseries.

Why?

- *For* Reproducibility *(command based / scripted)*
- *For* Automation *(scalability, scenario analysis)*
- *For* Streamlining *(more efficiency, less error-prone)*
- *For* Compatibility *(with extensive R research software)*

```
rswap_plot_multi(project_path, vars = c("H", "WC", "DRAINAGE"))
```

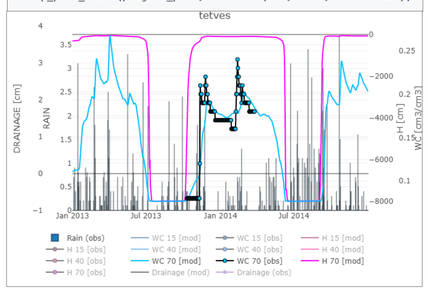

Figure 3: Interactive multivariate plot of user-chosen input, output, and observed timeseries.

Notable Features

- Backwards compatible with existing SWAP projects *(SWAPtools compatibility under development)*
- Intuitively flexible and scalable to user requirements *(designed for OPTAIN, a project with 14 different case studies, all with different needs)*
- Functional programming with low level access *(core functions are documented and accessible to the user, allowing low-level custom applications)*
- Open sourced and (hopefully) community driven development *(on github.com/moritzshore/rswap)*
- Quick-setup functionality *(Set up rswap and run SWAP with a single command: `rswap_init()`)*
- Documented with runnable examples *(allowing for ease of use, no steep learning curve)*

Advanced Functionality

→ The core function of *rswap* is `run_swap()`, which runs a SWAP project with a modified routine using lower level *rswap* functions. If multiple projects are passed to this function, they are run in **parallel** using `run_swap_parallel()`. The performance increases outstrip normal running dramatically, as seen in Figure 4, allowing for more computationally heavy workflows. Further improvements to runtime are planned, such as "forking" support for Linux and improved handling of input files in memory.

→ Parallel running functionality is utilized within the **sensitivity analysis** function `check_swap_sensitivity()` which also employs *rswap* functions to perform a sensitivity analysis. The function returns a dataframe of the analysis results, as well as an interactive *plotly* plot seen in Figure 5.

→ The ability of *rswap* functions to build upon themselves, such as the two aforementioned functions allows for an increasingly **abstracted** (easier) environment in which to build more and more advanced operations. Planned advanced features include scenario analysis, parameter estimation and hard-calibration plus ideas suggested (or implemented) by users on GitHub (github.com/moritzshore/rswap).

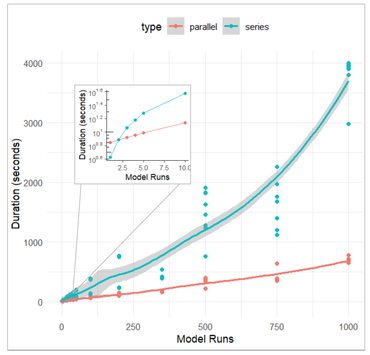

Figure 4: Comparison between running SWAP projects in series and in parallel using *rswap*. The benchmarking was done with a model setup which takes 3.14 seconds to run and was run in parallel on 28 PSOCK threads (Windows). Parallel running is already quicker at three runs, and 5.5x faster at 1000 runs. Further improvements are expected with Linux compatibility and "forking".

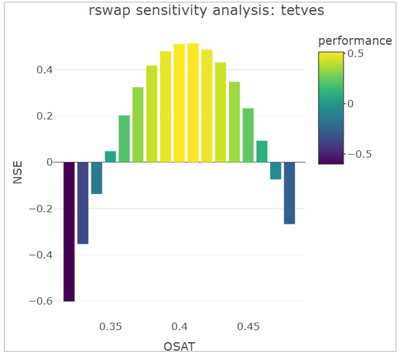

Figure 5: *rswap* can use its own exported functions to run a (parallelized) sensitivity analysis for any parameter and statistical metric using `check_swap_sensitivity()`. An interactive *plotly* plot is created, as well as a dataframe containing analysis results. *rswap* uses `modify_swap_file()`, `run_swap_parallel()` and `get_swap_performance()` for this functionality.

References

[1] Heinen, Marius, Martin Mulder, and Joop Kroes. SWAP 4: technical addendum to the SWAP documentation. Wageningen Environmental Research, 2021.

[2] Heinen, M., Mulder, M., van Dam, J., Bartholomeus, R., de Jong van Lier, Q., de Wit, J., de Wit, A., and Hack-ten Broeke, M.: SWAP 50 year: Advances in modelling soil-water-atmosphere-plant interactions, EGU General Assembly 2024, Vienna, Austria, 14–19 Apr 2024, EGU24-21812, https://doi.org/10.5194/egusphere-egu24-21812, 2024.

[3] Kroes, J. G., Van Dam, J. C., Bartholomeus, R. P., Groenendijk, P., Heinen, M., Hendriks, R. F. A., ... & Van Walsum, P. E. V. (2017). SWAP version 4: theory description and user manual. Alterra-rapport-Wageningen University and Research Centre, (2780).

nibio.no

PO Box 115, N-1431 Ås, Norway
+47 406 04 100

Corresponding Author:
Moritz Shore
moritz.shore@nibio.no

OPTAIN

This package was developed for the OPTAIN project (https://optain.eu) and has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No. 862756.