The Journal of Open Source Software

r2ogs6: An R wrapper of the OpenGeoSys 6 Multiphysics Simulator

Ruben Heinrich[*1], **Johannes Boog**[†2], **Philipp Schad**[‡2], **and Thomas Kalbacher**[§2]

1 Leipzig University of Applied Sciences, Karl-Liebknecht-Strasse 132, 04277 Leipzig, Germany **2** Helmholtz Centre for Environmental Research, Department Environmental Informatics, Permoser Str. 15, 04318 Leipzig, Germany

DOI: 10.21105/joss.05360

Software

- Review
- Repository
- Archive

Editor: Kristen Thyng

Reviewers:

- @waternumbers
- @ldecicco-USGS

Submitted: 21 December 2022
Published: 18 December 2024

Summary

Understanding the impacts of climate change and hydrologic extreme events on our sub–surface earth system is even in temperate zones of utmost importance (Felfelani et al., 2017; Field et al., 2012; Wu et al., 2020). Key tools to develop such understanding are physics–based simulation models that describe the manifold interactions of involved natural phenomena across time and space (X. Li et al., 2023; Steefel et al., 2015).

Our package `r2ogs6` provides a file–based interface of the multi–physics simulation code `OpenGeoSys 6` (Bilke et al., 2019; Kolditz et al., 2012) to the `R` programing and statistical computing environment. `r2ogs6` enables `R` users to perform and analyze simulation models of the sub–surface earth system in `R`. It allows to access the capabilities of `OpenGeoSys 6` to simulate thermo-hydro-mechanical-chemical and biological (THMC/B) processes in porous and fractured media within `R`. In this way, `r2ogs6` enables `R` users to model sub–surface phenomena and technologies such as groundwater flow, reactive transport, geothermal energy usage and/or nuclear waste repositories as well as to analyze and further process simulation output. `r2ogs6` enables users to prepare and manipulate `OpenGeoSys 6` simulation models, run the simulations and retrieve corresponding output, all within an `R` session or simple `R` scripts. Therefore, `R` classes and functions were designed to communicate with the respective `OpenGeoSys 6` input and output files as well as executables. In addition to single-simulation runs, `r2ogs6` supports ensemble runs that can be used to set up uncertainty and sensitivity analyses as well as parameter studies. Furthermore, `r2ogs6` allows conducting and documenting `OpenGeoSys 6` simulations in reproducible `R` scripts or notebooks. As `OpenGeoSys 6` is continuously being developed further, code generation functions for `r2ogs6` developers were included to speed up the package updating process in case of future changes to `OpenGeoSys 6`.

`r2ogs6` was designed to be used by domain researchers, data scientists and students working with `OpenGeoSys 6`. Moreover, `r2ogs6` was intended to include `OpenGeoSys 6` into `R` based scientific workflows and aims to bridge the gap between data produced by a scientific simulation code and data science.

Statement of need

Major challenges humanity has to face in the coming decades are climate change and hydrologic extremes (Field et al., 2012). Understanding the impacts of climate change and hydrologic

*co-first author
†co-first author
‡co-first author
§co-first author

extreme events on our sub–surface earth system is even in temperate zones of utmost importance for ensuring adequate domestic and drinking water supplies, together with functioning lake and river systems with healthy aquatic ecosystems and ecosystem services (Felfelani et al., 2017; Wu et al., 2020). Of course, the needs of the population and the needs of nature are often in conflict, which increases the necessity to study the complex interaction of both within different scenarios. The core of such studies is most often the system and scenario analysis of individual or coupled earth systems compartments through physics simulations (X. Li et al., 2023; Steefel et al., 2015). In physics simulation models, multiple coupled natural processes are implemented, which are usually described with partial differential equations. Solving these equations requires appropriate numerical methods such as the finite element method (FEM). For reasons of performance, (multi) physics simulators are mostly implemented in languages such as `FORTRAN`, `C` or `C++` (Steefel et al., 2015).

One of these simulators is `OpenGeoSys` (OGS) (https://www.opengeosys.org/), a scientific open source project for the development of numerical methods to simulate thermo-hydro-mechanical-chemical and biological (THMC/B) processes in porous and fractured media (Bilke et al., 2019; Kolditz et al., 2012). OGS has applications ranging from small-scale geotechnical investigations (Grunwald et al., 2020), to reservoir studies (D. Li et al., 2014), nuclear waste repositories (Pitz et al., 2023) and even groundwater management of entire landscapes (Jing et al., 2018; Pujades et al., 2023).
To identify the right action needs and to derive further decision support for the above described problems it often requires to explore several problem scenarios through meaningful model ensembles. This translates to more and more complex and larger model setups (Asher et al., 2015). Thus, faster and more efficient model setup and parametrization procedures are needed. But while `OpenGeoSys` is a powerful FEM code, setting up, running and evaluating multiple simulations can prove complicated.

Here is where languages such as `R` and `Python` can prove useful. Via an interface that adds a layer on top of `OpenGeoSys 6`, the user can access preprocessing tools, the solver itself and postprocessing tools alike, thus increasing usability and accessibility. The development and application of high–level programming and/or scripting languages interfaces to geoscientific simulators has been gaining increasing attention in recent years. Examples are `FloPy`: a `Python` interface to the groundwater simulator `Modflow` Langevin et al. (2021), `RedModRPhree`: an `R` package with utility functions to the geo-chemical solver `Phreeqc` Charlton & Parkhurst (2011), the `toughio` Python interface to the multi-phase flow simulator `TOUGH` Pruess (2004), the Python interfaces to `OpenGeoSys 5`: `ogs5py` (Müller et al., 2021) and `OpenGeoSys 6`: `ogs6py` (Buchwald et al., 2021) as well as the `R` interface to `OpenGeoSys 5`: `r2ogs5` (Schad et al., 2023). `FloPy`, `ogs5py`, `ogs6py` and `r2ogs5` provide object-oriented frameworks to set-up, call the executables and import the output of the associated simulators. The idea of these packages is to provide `Python` and `R` functions to pre- and process simulation data but to just run simulations via a call to the the operating system to execute the simulators with the prepared input. `RedModRPhree` and `toughio` rather focus on additional pre- and post processing functions to simplify the set-up of simulations with `R` and `Python`. These packages still rely on external execution of the associated simulation program.

Altough there already exist the `OpenGeoSys 6 Python` interface `ogs6py`, we consider an `R` interface to be just as important, as `R` is a well known language in the environmental and geosciences. Furthermore, since `R` is a popular language in the field of data science with many powerful packages for data analysis and vizualisation, e. g. `dplyr` (Wickham, François, et al., 2021) and `ggplot2` (Wickham, Chang, et al., 2021), it's a natural choice for processing data generated by simulation tools such as `OpenGeoSys 6`. `r2ogs6` follows a similar design approach compared to `FloPy`, `ogs5py`, `ogs6py` and `r2ogs5`: it provides an object-oriented approach with `R` classes and functions to set-up an `OpenGeoSys 6` simulation, call the respective executable and import the output of finished simulations (see section *Package Structure* for more details).

Moreover, `r2ogs6` can facilitate the calibration of `OpenGeoSys 6` models. One possibility is to use implemented functions to design ensemble runs. A second possibility is to use available `R`

packages for model calibration such as `lhs` (Carnell, 2021), `mlrBO` (Bischl et al., 2017). For `R` users who do not have a lot of (or any) experience with environmental and geoscientific sub–surface simulations, yet an interest in such, `r2ogs6` provides a good starting point. Utilizing `r2ogs6`, users can easily set up their first `OpenGeoSys 6` simulations by choosing one of the provided benchmark files. Moreover, with `R` scripts and `R--Markdown` or `JupyteR` notebooks, modeling workflows can easily be documented, published and shared with peers.

`r2ogs6` has already been applied in (Heinrich, 2021), where its ensemble functionality was tested utilizing the `OpenGeoSys 6` Theis' problem benchmark files (Walther, 2020; Wang, 2020); or to calibrate groundwater flow models (Boog et al., 2021).

Package Structure

`r2ogs6` is thought to set up an `OpenGeoSys 6` simulation inside an `R` session by executing specified model creation functions or by reading existing `OpenGeoSys 6` input files. With further functions, the simulation can be executed and corresponding output can be read into the `R` session again. Figure 1 highlights the structure of `r2ogs6`. The central element that represents an `OpenGeoSys 6` simulation is the `OGS6` object, which is an instance of a `R6` class. This object represents a single simulation; multiple simulations can be defined with the `OGS6_Ensemble` class. An `OGS6` object contains several child objects that represent the simulation input and output. The main `OpenGeoSys 6` input files are the project file `*.prj`, geometry file `*.gml` and input FEM mesh file(s) `*.vtu`. These are read in or written via `S3` class based functions (block `read_in* / export*` in Figure 1). When reading in, the XML–based `*.prj` input file is parsed. Individual tags are represented as `S3` class objects which are available via active fields in the `OGS6` object. Individual `*.prj` tags may change due to ongoing development activities in `OpenGeoSys 6`, therefore, future updates of the related classes may be necessary. To simplify updates like this, helper functions for analyzing `*.prj` files as well as suggesting and creating classes were implemented.

As the `*.gml` and the `*.vtu` files are less complex and less likely to change, these files are represented as `R6` class objects and also available as active fields inside the `OGS6` object. To execute simulations, functions for writing the `OpenGeoSys 6` input (`ogs6_export_simfiles()`) and call the `OpenGeoSys 6` executable (`ogs6_run_simulation()`) were implemented. Note that an `OpenGeoSys 6` executable or singularity container needs to be present. Default executables and paths can be defined in a configuration file.

During execution `OpenGeoSys 6` generates output data as `*.vtu` files. These files are produced at user defined timesteps of the simulation and are referenced in a `*.pvd` file. The function `ogs6_read_output_files()` then attaches the output files to the `OGS6` object as `OGS6_pvd` objects (which in turn reference `OGS6_vtu` objects). In this way, all data required for and produced by `OpenGeoSys 6` can be represented as `R` native objects and results can be processed further using `R` functions.

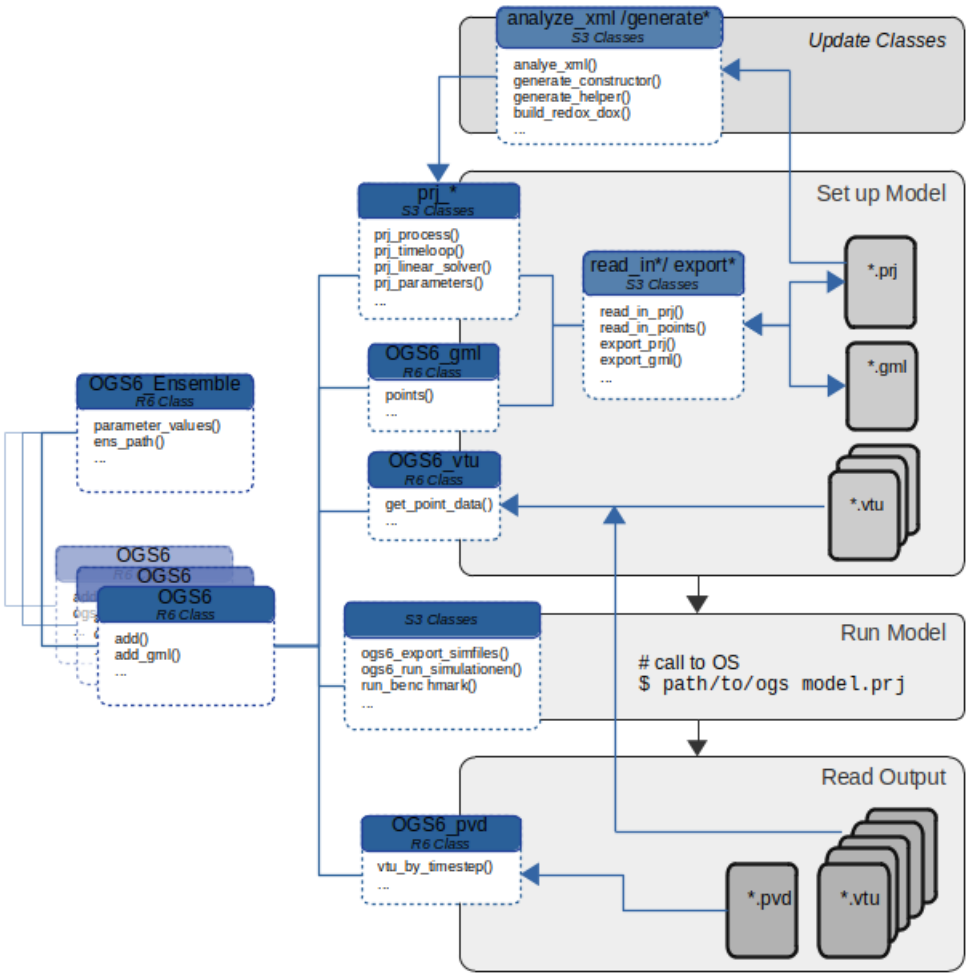

Figure 1: Schematic of the `r2ogs6` structure.

Examples

Quick Start with Theis' Problem

Theis' problem examines the lowering of the water level around a pumping well. Water is pumped from a well, which induces a lowering of the water level over time. The Theis' problem is a common benchmark for sub–surface flow simulators. See the respective desription on the OpenGeoSys 6 page here. The required input files are already present in the installed `r2ogs6` package.

You can just create an `.R` script with the `r2ogs6` commands to set-up a simulation from a benchmark file directly. Just load the package and get the path to the Theis' problem benchmark `.prj` file.

```
library(r2ogs6)

# Modify the prj_path depending on where you saved the benchmark file.
prj_path <- system.file("extdata/benchmarks/AxiSymTheis/",
                        "axisym_theis.prj", package = "r2ogs6")
```

Then define the path were the generated script is to be save (`script_path`) and were the simulation is to be run (`sim_path`). Note, that the folders should already exist. Finally, call the respective function to generate the script; the script will be named according to the name of the project file you specified but with the extension `.R` (here: `axisym_theis.R`).

```
script_path <- "path/to/scripts"
sim_path <- "path/to/sim"

ogs6_generate_benchmark_script(prj_path,
                               sim_path = sim_path,
                               script_path = script_path)
```

You can now open the generated script `axisym_theis.R` and have a all the commands to generate and run a simulation object ready to explore.

Sensitivity Analysis of the Theis' Problem

Here, we will set up an ensemble of models to visualize the sensitivity of the water level lowering to changes in the material specific parameter called `storage`.

The variable that corresponds to the water level is the `pressure` in the water, which increases with depth. So changes in `storage` will induce changes in the evolution of the `pressure` gradient around the well.

At first, load the required libraries.

```
library(r2ogs6)
library(ggplot2)
library(dplyr)
```

Then, create a simulation object to base the ensemble on and read in the `.prj` file.

```
# Change this to fit your system
testdir_path <- tempdir()
sim_path <- paste0(testdir_path, "/axisym_theis_sim")

# Create the simulation object
ogs6_obj <- OGS6$new(sim_name = "axisym_theis",
                     sim_path = sim_path)

# The input files should be present in your r2ogs6 installation directory
prj_path <- system.file("extdata/benchmarks/AxiSymTheis/",
                        "axisym_theis.prj", package = "r2ogs6")

# Now read in the input files
read_in_prj(ogs6_obj, prj_path, read_in_gml = T)
```

To examine the effects of `storage` we change it by 1%, 10% and 50%. We can use the `percentages_mode` parameter of `OGS6_Ensemble` for this.

```
# Assign percentages
percentages <- c(-50, -10, -1, 0, 1, 10, 50)

# Define an ensemble object
ogs6_ens <-
  OGS6_Ensemble$new(
    ogs6_obj = ogs6_obj,
    parameters = list(list(ogs6_obj$media[[1]]$properties[[4]]$value,
```

```
                                percentages)),
    percentages_mode = TRUE)
```

Now start the simulation, read and visualize the output.

```
# Start the simulation
ogs6_ens$run_simulation()

# Attach the output files to the ensemble object.
lapply(ogs6_ens$ensemble, ogs6_read_output_files)

# Extract point specific data of the `pressure` variable from the output files
storage_tbl <-
  ogs6_ens$get_point_data(point_ids = c(0, 1, 2),
                          keys = c("pressure"))

# Compute the spatial average of the pressure for all simulations
avg_pr_df <- storage_tbl %>%
  group_by(sim_id, timestep) %>%
  summarise(avg_pressure = mean(pressure))

# Plot the spatially averaged pressure over time for all simulations
ggplot(avg_pr_df, aes(x = as.numeric(as.factor(timestep)),
                      y = avg_pressure,
                      group = sim_id)) +
  geom_point(aes(color = as.factor(sim_id))) +
  geom_line(aes(color = as.factor(sim_id))) +
  labs(color = "sim id") +
  xlab("Timestep")
```

The plot then shows the development of the average pressure in each simulation of the ensemble over time.

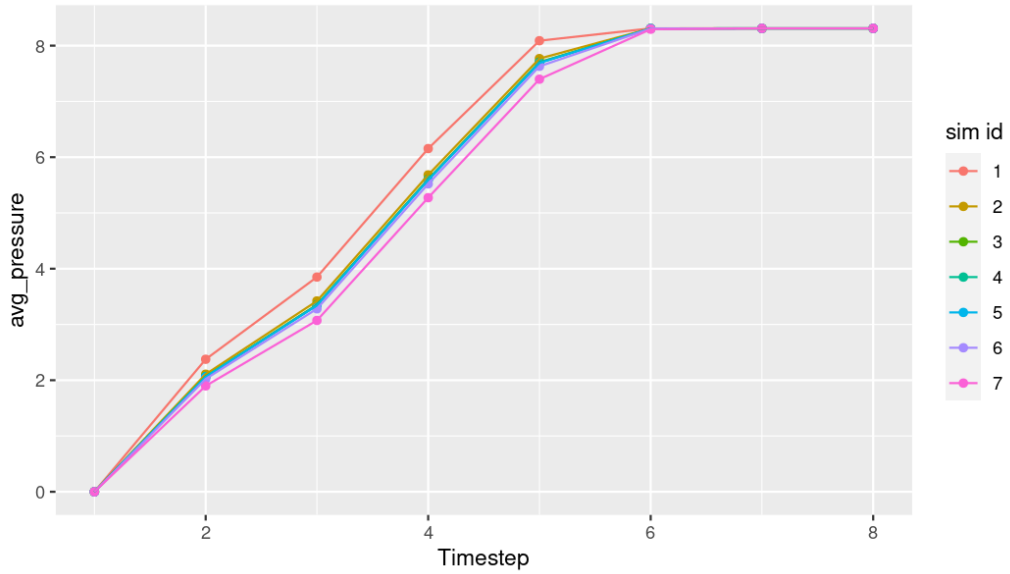

Figure 2: plot of chunk p-t-all-combined-plot

Check the following package vignettes for more information: - a guide to set up a single simulation of a hydro-mechanics benchmark (link) - a guide to create ensemble runs (link) - a

development guide (link)

Acknowledgements

We acknowledge funding from the Initiative and Networking Fund of the Helmholtz Association through the project *Digital Earth* (funding code ZT-0025). Furthermore, we acknowledge the Helmholtz Centre for Environmental Research–UFZ for additional funding and support. We would like to express our gratitude to the *OpenGeoSys Community* for technical support and for hosting the GitLab server (https://gitlab.opengeosys.org) for our development. The package has in part been developed on the High-Performance Computing (HPC) Cluster EVE, a joint effort of both the Helmholtz Centre for Environmental Research - UFZ (http://www.ufz.de/) and the German Centre for Integrative Biodiversity Research (iDiv) Halle-Jena-Leipzig (http://www.idiv-biodiversity.de/). We would like to thank the administration and support staff of EVE who keep the system running and support us with our scientific computing needs: Thomas Schnicke, Ben Langenberg, Guido Schramm, Toni Harzendorf and Tom Strempel from the UFZ, and Christian Krause from iDiv.

References

Asher, M. J., Croke, B. F., Jakeman, A. J., & Peeters, L. J. (2015). A review of surrogate models and their application to groundwater modeling. *Water Resources Research*, *51*(8), 5957–5973. https://doi.org/10.1002/2015WR016967

Bakker, M., Post, V., Langevin, C. D., Hughes, J. D., White, J. T., Starn, J. J., & Fienen, M. N. (2016). Scripting MODFLOW model development using Python and FloPy. *Groundwater*, *54*(5), 733–739. https://doi.org/10.1111/gwat.12413

Bilke, L., Flemisch, B., Kalbacher, T., Kolditz, O., Helmig, R., & Nagel, T. (2019). Development of open-source porous media simulators: Principles and experiences. *Transport in Porous Media*, *130*, 337–361. https://doi.org/10.1007/s11242-019-01310-1

Bischl, B., Richter, J., Bossek, J., Horn, D., Thomas, J., & Lang, M. (2017). *mlrMBO: A modular framework for model-based optimization of expensive black-box functions*. https://doi.org/10.32614/CRAN.package.mlrMBO

Boog, J., Sips, M., De Lucia, M., Eggert, D., & Kalbacher, T. (2021). DATA-driven CALibration of large-scale physical groundwater flow models using meta-modeling and visual analytics. *AGU Fall Meeting Abstracts*, *2021*, H34D–04.

Buchwald, J., Kolditz, O., & Nagel, T. (2021). ogs6py and VTUinterface: Streamlining OpenGeoSys workflows in Python. *Journal of Open Source Software*, *6*(67), 3673. https://doi.org/10.21105/joss.03673

Carnell, R. (2021). *lhs: Latin hypercube samples*. https://doi.org/10.32614/cran.package.lhs

Charlton, S. R., & Parkhurst, D. L. (2011). Modules based on the geochemical model PHREEQC for use in scripting and programming languages. *Computers & Geosciences*, *37*(10), 1653–1663. https://doi.org/10.1016/j.cageo.2011.02.005

De Lucia, M., & Kühn, M. (2021). Geochemical and reactive transport modelling in R with the RedModRphree package. *Advances in Geoscience*, *56*, 33–43. https://doi.org/10.5194/adgeo-56-33-2021

Felfelani, F., Wada, Y., Longuevergne, L., & Pokhrel, Y. N. (2017). Natural and human-induced terrestrial water storage change: A global analysis using hydrological models and GRACE. *Journal of Hydrology*, *553*, 105–118. https://doi.org/10.1016/j.jhydrol.2017.07.048

Field, C., Barros, V., Stocker, T., & Dahe, Q. (2012). *Managing the risks of extreme events*

and disasters to advance climate change adaptation: Special report of the intergovernmental panel on climate change. https://doi.org/10.1017/CBO9781139177245

Grunwald, N., Maßmann, J., Kolditz, O., & Nagel, T. (2020). Non-iterative phase-equilibrium model of the H2O-CO2-NaCl-system for large-scale numerical simulations. *Mathematics and Computers in Simulation*, *178*, 46–61. https://doi.org/10.1016/j.matcom.2020.05.024

Heinrich, R. (2021). *Entwicklung und Implementierung der Schnittstelle R2OpenGeoSys zur Workflow-Optimierung am Beispiel der Modellierung eines Stofftransportproblems* [Master's thesis]. HTWK Leipzig.

Jing, M., Heße, F., Kumar, R., Wang, W., Fischer, T., Walther, M., Zink, M., Zech, A., Samaniego, L., Kolditz, O., & others. (2018). Improved regional-scale groundwater representation by the coupling of the mesoscale Hydrologic Model (mHM v5. 7) to the groundwater model OpenGeoSys (OGS). *Geoscientific Model Development*, *11*(5), 1989–2007. https://doi.org/10.5194/gmd-11-1989-2018

Kolditz, O., Görke, U.-J., Shao, H., & Wang, W. (Eds.). (2012). *Thermo-hydro-mechanical-chemical processes in porous media: Benchmarks and examples* (1. ed.). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-27177-9

Langevin, C., Hughes, J., Banta, E., Provost, A., Niswonger, R., & Panday, S. (2021). MODFLOW 6 modular hydrologic model version 6.2. 2. *US Geological Survey Software Release*, *10*, F76Q1VQV.

Li, D., Bauer, S., Benisch, K., Graupner, B., & Beyer, C. (2014). OpenGeoSys-ChemApp: A coupled simulator for reactive transport in multiphase systems and application to CO 2 storage formation in northern Germany. *Acta Geotechnica*, *9*, 67–79. https://doi.org/10.1007/s11440-013-0234-7

Li, X., Feng, M., Ran, Y., Su, Y., Liu, F., Huang, C., Shen, H., Xiao, Q., Su, J., Yuan, S., & others. (2023). Big data in earth system science and progress towards a digital twin. *Nature Reviews Earth & Environment*, 1–14. https://doi.org/10.1038/s43017-023-00409-w

Luu, K. (2020). toughio: Pre- and post-processing Python library for TOUGH. *Journal of Open Source Software*, *5*(51), 2412. https://doi.org/10.21105/joss.02412

Müller, S., Zech, A., & Heße, F. (2021). ogs5py: A Python-API for the OpenGeoSys 5 scientific modeling package. *Groundwater*, *59*(1), 117–122. https://doi.org/10.1111/gwat.13017

Pitz, M., Grunwald, N., Graupner, B., Radeisen, E., Maßmann, J., Ziefle, G., Thiedau, J., & Nagel, T. (2023). Benchmarking a new TH2M implementation in OGS-6 with regard to processes relevant for nuclear waste disposal. *Environmental Earth Sciences*, *82*(13), 1–24. https://doi.org/10.1007/s12665-023-10971-7

Pruess, K. (2004). The TOUGH codes—A family of simulation tools for multiphase flow and transport processes in permeable media. *Vadose Zone Journal*, *3*(3), 738–746. https://doi.org/10.2113/3.3.738

Pujades, E., Kumar, R., Houben, T., Jing, M., Rakovec, O., Kalbacher, T., & Attinger, S. (2023). Towards the construction of representative regional hydro (geo) logical numerical models: Modelling the upper Danube basin as a starting point. *Frontiers in Earth Science*, *11*, 1061420. https://doi.org/10.3389/feart.2023.1061420

Schad, P., Boog, J., & Kalbacher, T. (2023). r2ogs5: Calibration of numerical groundwater flow models with bayesian optimization in R. *Groundwater*, *61*(1), 119–130. https://doi.org/10.1111/gwat.13221

Steefel, C. I., Appelo, C. A. J., Arora, B., Kalbacher, D., Kolditz, O., Lagneau, V., Lichtner, P. C., Mayer, K. U., Meeussen, J. C. L., Molins, S., Moulton, D., Shao, D., Simunek, J., Spycher, N., Yabusaki, S. B., & Yeh, G. T. (2015). Reactive transport codes for subsurface environmental simulation. *Computational Geosciences*, *19*(3), 445–478. https:

//doi.org/10.1007/s10596-014-9443-x

Walther, M. (2020). *Theis solution for well pumping*. https://www.opengeosys.org/docs/benchmarks/hydro-component/theis/hc_theis/

Wang, W. (2020). *Theis' problem*. https://www.opengeosys.org/docs/benchmarks/liquid-flow/liquid-flow-theis-problem/

Wickham, H., Chang, W., Henry, L., Pedersen, T., Takahashi, K., Wilke, C., Woo, K., Yutani, H., & Dunnington, D. (2021). *ggplot2: Create elegant data visualisations using the grammar of graphics*. https://doi.org/10.32614/cran.package.ggplot2

Wickham, H., François, R., Henry, L., & Müller, K. (2021). *dplyr: A grammar of data manipulation*. https://doi.org/10.32614/CRAN.package.dplyr

Wu, W.-Y., Lo, M.-H., Wada, Y., Famiglietti, J. S., Reager, J. T., Yeh, P. J.-F., Ducharne, A., & Yang, Z.-L. (2020). Divergent effects of climate change on future groundwater availability in key mid-latitude aquifers. *Nature Communications*, *11*(1), 3710. https://doi.org/10.1038/s41467-020-17581-y