Proceedings World Geothermal Congress 2020+1
Reykjavik, Iceland, April - October 2021

T2GEORES: A Python library to manage TOUGH2 simulations on Geothermal Systems

José Erick Jiménez Majano

GRÓ-GTP, Geothermal Training Programme, Grensasvegur 9, 108 Reykjavik, Iceland

erick@grogtp.is

Keywords: TOUGH2, Python, numerical modelling.

ABSTRACT

Detail numerical reservoir modelling is the preferable approach for a resource assessment once exploitation has started in a geothermal field. Moreover, it can give an insight about the general understanding of a system once some wells have been drilled. At present, TOUGH2 is the main simulator for modelling in geothermal systems. However, due to the fixed format of its input file, creating a large model by manual editing could be time consuming and a source of error. A long the years many efforts have been done through the use of GUIs and scripting languages to manipulate the string of data on the preprocessing and postprocessing steps necessary to setting up a model, avoiding manual editing of input files and manual output selection. This paper introduces T2GEORES, a Python library developed with the objective to be handy and simple, specifically oriented to perform modelling in geothermal reservoirs. Providing TOUGH2 input files generation capability from mesh creation to parameters setting. It creates some output visualizations, reducing the time consumed during the calibration process of a model by comparing the results with real data from the field such as formation temperature, flowing enthalpy, drawdown pressure and cooling history.

1. INTRODUCTION

TOUGH2 is a nonisothermal, multiphase and multiphase fluid flow simulator (Pruess K. , 2004) suitable to perform modelling on geothermal reservoirs, nuclear waste disposal and other hydrology applications. The simulator has been used to perform reservoir assessments since it first release (Bjornsson et al., 2003; Croucher & O'Sullivan, 2008; Tómasdóttir, 2018) and it is part of the current state of art for geothermal reservoir modelling. TOUGH2 requeries a text-based input file, where every parameter has a fix position and length. Manual editing of file can lead to syntax and/or logic errors. Thus, many GUIs have been developed to perform from simple tasks such as mesh generation and rock edition up to a complete managing of a simulation: TIM(Yeh, Croucher, & O'Sullivan, 2013), Leapfrog (Newson, et al., 2012), Wingridder (Pan, 2003) and PetraSim (Yamamoto, 2008). However, the functionalities presented on the GUI depends on the developer and automatation of runs cannot be performed. In the other hand, scripting languages have gained relevenance for their allowance on automatition of processes that can explore many configurations a the model from a single structured file, the most complete solutions on this category are PyTOUGH (Croucher, 2011) and recently toughio (Luu, 2020). T2GEORES is a python library specifically coded for the development of numerical models on geothermal reservoirs, the modular approach is shared with a set of routines written on FORTRAN presented by Audigane et al. (2011). It sets a framework based on a defined folder structure to easily generate TOUGH2 input files and handles the simulator outcome. This paper proposes a methodolgy for numerical simulation based on T2GEORES. Three applications are presented. The first presents a mesh generation capability, the second evaluates a production management strategy on a geothermal field and third setup a simple model and explores its results.

2. WORKFLOW

The library is a compilation of modules containing several functions to handle mesh creation, pre-processing and post processing steps for computational models executed by TOUGH2. The use of Python as a selected programming language enhance the velocity of development since the standard library and other open source libraries implement algorithms that are ready to use to solve others problems in any particular case. The approach of T2GEORES is based on the calibration of the model by comparing the measurements from the field with the resulting outputs from the simulator. To handle some of the data a database managed by sqlite is fed with real data from the field, such as temperature and pressure logs, mass flow, flowing enthalpy, pressure drawdown, cooling history, well track, feedzone locations and well coordinates. The information is used to perform comparisons with the model output by tracking the results from the blocks assigned to well feedzones after every run and to generate the OBSERVATION section of iTOUGH2 (Finsterle, 2015) input file in case an inversion is performed. During calibration process, a lot of output files can be generated. Thus, to store and manage output files (images, PDFs, and text files), for every model it is suggested to have its own directory with a defined structure as shown on Figure 1. The content of every directory is described:

- calib: comparisons of current state of the model and previous stages. Necessary, to track the progress of the model during the forward runs.
- input: field information used to define and calibrate the model.
- mesh: input and output files from the mesh generator.
- model: necessary input file and consequently output files from the simulator.
- output: format text files and json files coming from postprocessing steps.
- scripts: suggested folder to write and execute the python modules from the T2GEORES library

The workflow presented on Figure 1 describe the general procedure of reservoir modelling on the framework of T2GEORES through the establishment of the module corresponding to each stage. The preprocessing steps are dominated by the generation of grid, establishment of model parameters, storage of real measurement and creation of TOUGH2 input file. While the postprocessing steps creates the output files for friendly user exploration and different comparisons based on chart basis. Meanwhile, during calibration it is necessary to update the rock distribution, sink/sources parameters, initial condition, and rock parameters before

every new run. Finally, after a good match is obtained the data preparation of the inverse model can be handled by the module it2Obs.

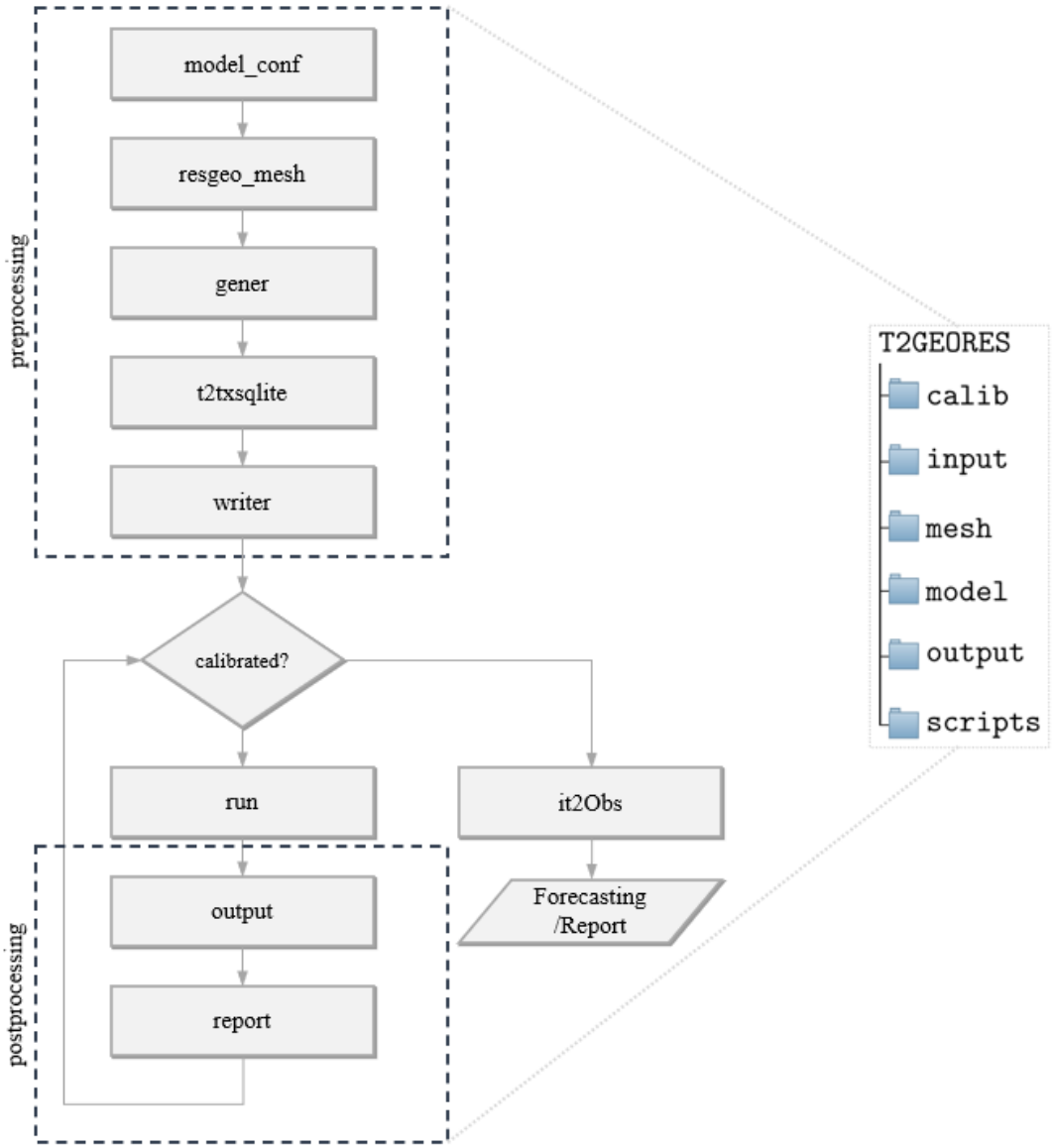


Figure 1: flowchart describing the interaction of modules during modelling and working structure

2.1 Input file generation

The parameter definition for the input file is stablished through a python dictionary. The keywords follow the nomenclature from the TOUGH2 user manual (Pruess, Oldenburg, & Moridis, 2012). The time stepping for the TIMES section on the TOUGH2 input file can be setup by using an array of datetime items and defined time gap. Another capability is the generation of initial conditions suitable to be used on the INCON section, boiling conditions are assume for a defined thermal gradient and initial temperature on the surface. After the first run the SAVE file (.sav) generated by TOUGH is added to the next run input file to calculate the new conditions from the previous stage.

The GENER elements are entered through a dictionary which manage the input data for constant heat sources and wells on deliverability. The producer and injector wells are considered as MASS elements, the database serves as input to generate the generation times and flow rate from the producer wells with respect to a defined reference date. For the injector wells the flowing enthalpy is also included. The predefined values for all the sections can be modified on the format file of the library.

2.2. Output handling

TOUGH2 generates a text file after every run which contains information about several parameters from all the elements in the model on each selected time step and the corresponding information for the sink and sources. The aim of the library is to extract the information from every well on text files to perform comparison with real measurements. Thus, for every block assigned to a well pressure and temperature are extracted from top to bottom layer at the beginning of the transient stage to compare results with natural state data. However, it can be extracted at every other desired time step such as the certain period of production. For each

feedzone the flowing enthalpy and mass generation is extracted and arrange on the file for every time step. Finally, at a defined time a json file can be generated for all the elements on the mesh to perform cross sections of a selected parameter.

3. EXAMPLES

3.1. Mesh generation

The module resgeo_mesh contains several subroutines to define a mesh based on finite-volume elements. The core calculation of the voronoi cells is performed by AMESH (Haukwa, 1999). A regular mesh grid with a bigger distance between each element can be defined for the outer section of the mesh as it usually contains constant conditions along the different steps of modelling. Smaller elements can be created on the wellfield to recreate some features of the systems such as faults. A polygon without cavities can establish the limits between the far and well field by using a shapefile or simple text file. The routine can establish the outer elements as inactive by setting as negative value the volume of the elements on the far field. Furthermore, the mesh can be rotated from the lower left corner of the defined boundaries. Another important feature is the creation a single element for the existing wells or an arbitrary coordinate, every well block is separated from any other regular block a fix distance. Thus, every element well will have assigned an output from it can be tracked.

As a common practice the mesh requires the projected position on surface of the main feedzone of well. However, several feedzones can be assigned to a well by defining a percentage contribution at different levels on its track. After the creation of a mesh, the routine generates a json file with the correlative value from every well. The mesh setup can be on a dictionary which handles 36 parameters to fully setup the domain and features.

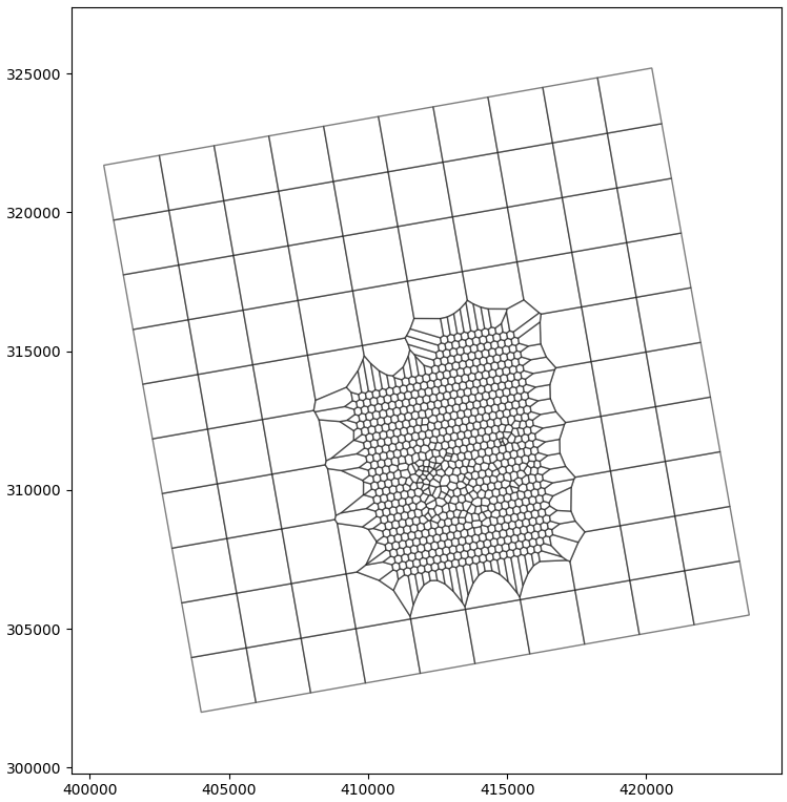


Figure 2: rotated mesh for a defined far and wellfield embedded in a polygon

3.2. Model forecasting

The setup of a model by manual edition is time consuming and can lead to errors. The T2GEORES currently manage a large quantity of output files for the user to review the trends and perform quick evaluation of specific elements most likely on well tracks. This aspect is the main difference between T2GEORES and PyTOUGH where the data is stored on the objects of a class. Despite the differences, it is possible to performed automatic runs structured with different model setups, such as: model forecasting for different exploitation scenarios or model calibration by comparing different output depending on range of values for rock permeabilities. The model forecasting can be done by totally defined the make-up wells through the establishment of initial date of operation, flow rate and expected flowing enthalpy for an injector on a dictionary. The Figure 3 represents a hypothetical scenario of production, by drilling six make-up wells: two injectors and four producers. The Figure 4 overlays the output of modelling on Ahuachapán Geothermal Field (Jiménez, 2019) at well AH-6 and the expected drawdown for three different production scenarios.

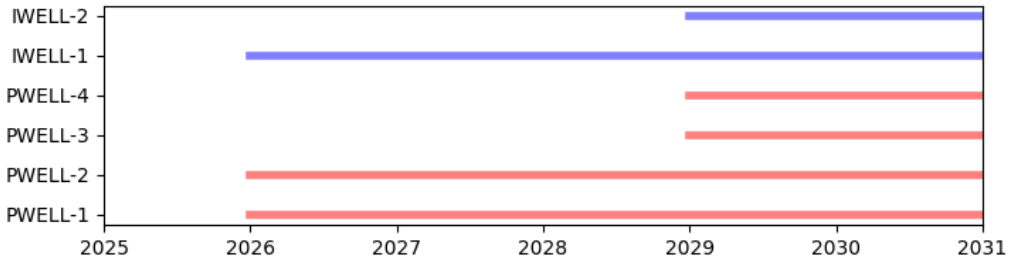


Figure 3: setup 1 for model forecasting

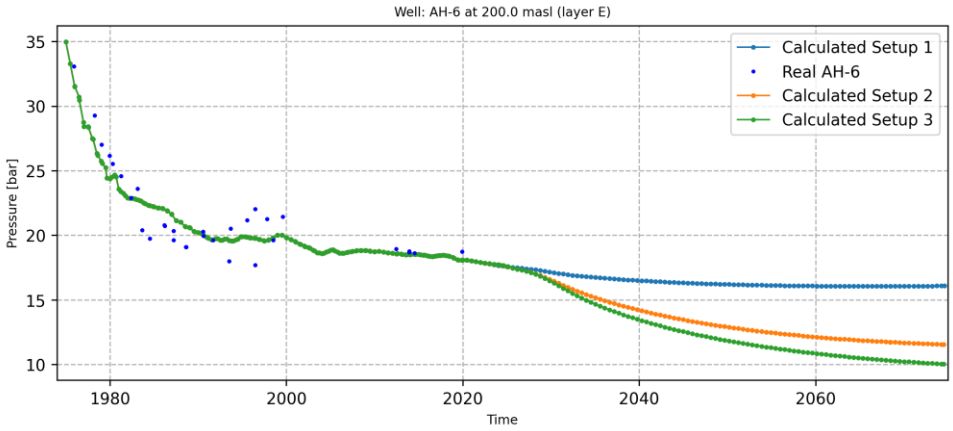


Figure 4: pressure forecasting on two different scenarios.

3.3. Simple geothermal system

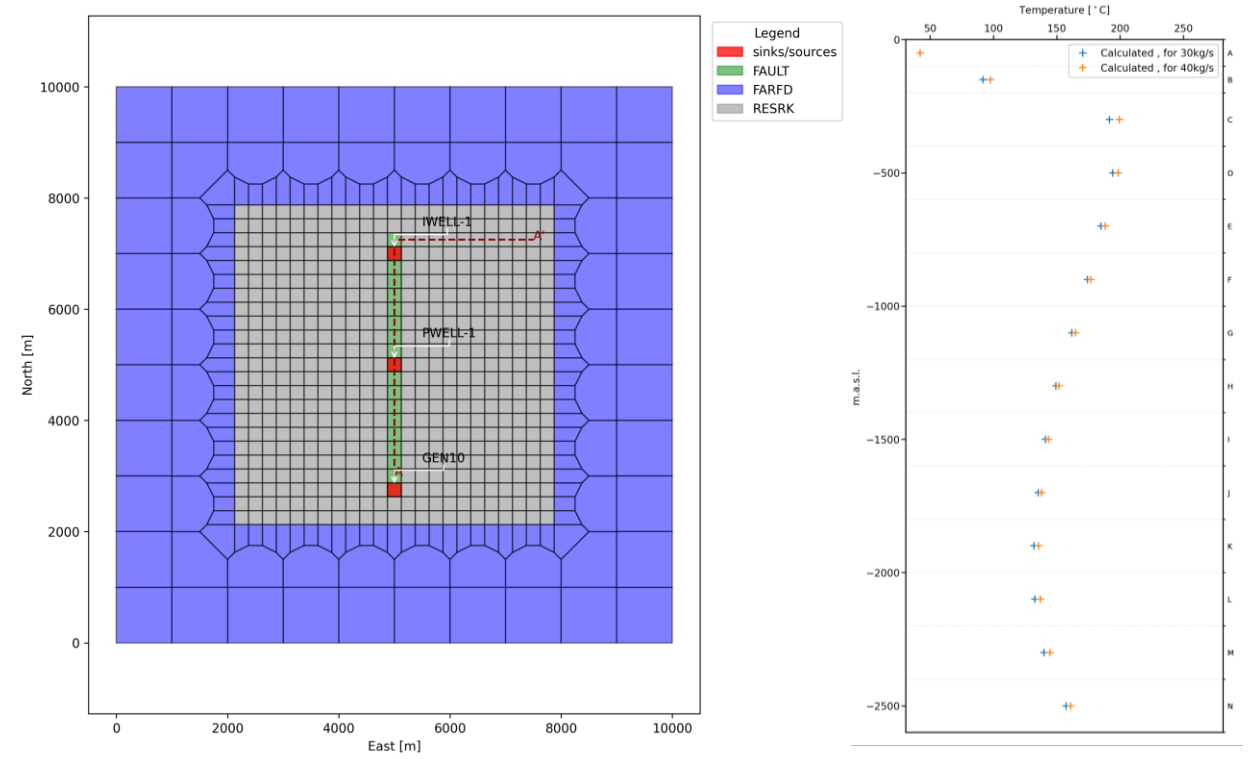
A simple model is setup to show some of the main features of the library. The grid consisting of a 10 km wide by 10 km long and a thickness of 3km, with 16 layers and 689 elements per layer. The initial conditions for all the elements are generated assuming a surface temperature of 30 °C, a gradient of 80 °C/km and considering boiling conditions from the top layer. The elements on the top and bottom layer as well as the blocks on the surrounds of the well field remain with constant conditions during the simulation. A high permeability channel is generated on the center line of the grid where an element injectes hot fluid at a specific enthalpy of 1100 kJ/kg. The rock distribution can be updated based on spatial boundaries by modifying a json file associating with the ELEME section of the TOUGH2 file. However, the module resgeo_mesh also can generate the necessary input files to work on Steinar (Vatnaskil Consulting Engineers, 2015) as part the preprocessing steps presented Figure 1.

The Figure 5 (a) shows the rock distribution on the layer C, the flow generator is located at the south, the producer well on the middle and the injector well at the north, the rock permeabilities used for the model are shown on the Table 1. A value of 2500 kg/m3 is used for density, 1000 J/kg°C as specific heat and 2.5 W/m°C as thermal conductivity for all rock types.

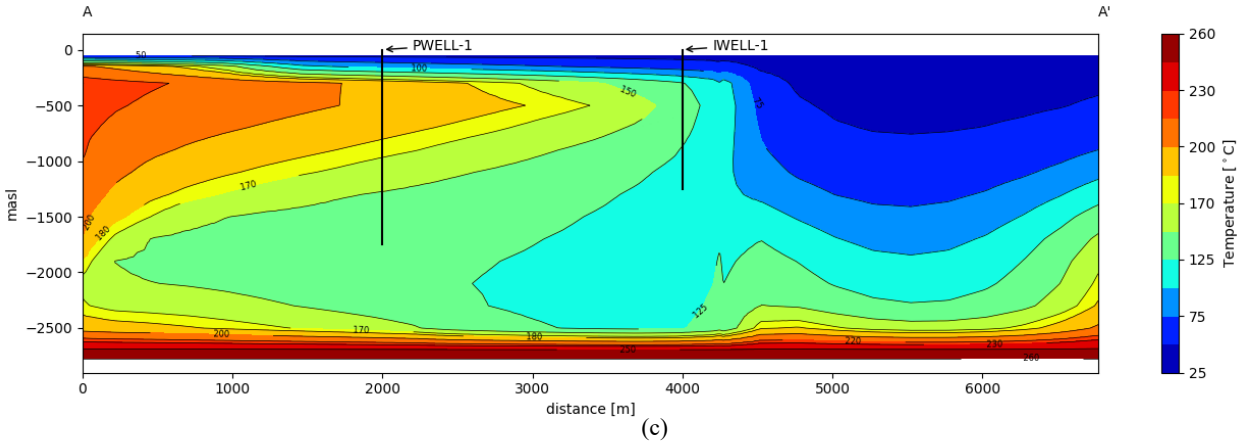
Table 1: Isotropic rock types on model

Rocktype	Permeability mD
CAPRK	1
FAULT	20
RESRK	10
FARFD	0.1
BASRK	0.01

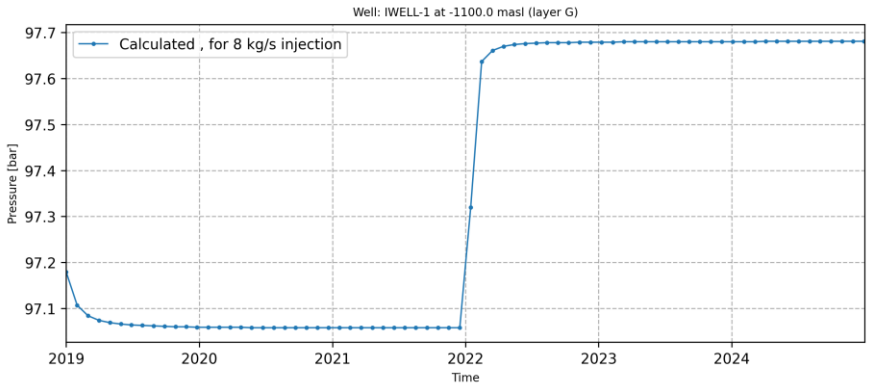
Usually, the first stage of model evaluation consists of the calibration of natural stage by comparing results with the formation temperature for every well. The scripts can be used to evaluate different model boundaries, the Figure 5 (b) shows the model temperature response for the location of the producer well PWELL-1 for two different inflow values of 30 and 40 kg/s coming from the source GEN10. The Figure 5 (c) shows the temperature on steady state along the cross section A-A', from Figure 5 (a), the configuration lead to a formation of a convective flow from the heat source to the producer well and inversion on temperature on the edges of the system at the north, close to the well injector well IWELL-1. Finally, the Figure 5 (d) evaluates the pressure change on the feedzone of the injector well due to the extraction on the producer well three years earlier than the injection. The preprocessing steps such as mesh generation can be setup in a single file from which the output can be manipulated to create the appropriated rock distribution. The postprocessing steps performed on this example can be executed from a single script that controls the simulator and carries the proper stream of data from the output files of TOUGH2.



(a) (b)



(c)



(d)

Figure 5: (a) rock distribution on layer C and sink/sources, (b) Temperature distribution on well PWELL-1 depending on inflow rate on GEN10, (c) Vertical temperature cross section over line A-A', (d) Monitoring pressure on feedzone on well IWELL-1.

4. LIBRARY ACCESS

T2GEORES and its documentation can be downloaded from GitHub (https://github.com/jejimenezm/T2GEORES) and from the Python Package Index under the MIT license.

5. CONCLUSION AND FUTURE WORK

A python library for managing TOUGH2 geothermal reservoir models have been developed. The library approach is distinguished from others by the use of a fixed directory structure. Nevertheless, it provides the basic functionality for geothermal reservoir modelling. The library is still on an early stage, two major improvements to come are the rock edition and 3D output visualization, for the user to perform the whole cycle of modelling within the library.

ACKNOWLEDGEMENTS

I would like to thanks to Julio Quijano for introduce and guiding me into the geothermal reservoir modelling and Andri Arnaldsson for providing ideas on the workflow of modelling during my staying in Iceland.

REFERENCES

Audigane, P., Chiaberge, C., Mathurin, F., Lions, J., & Picot-Colbeaux, G. (2011). A workflow for handling heterogeneous 3D models with the TOUGH2 family of codes: Applications to numerical modeling of CO2 geological storage. *Computers & Geosciences, 37*, 610-620.

Bjornsson, G., Hjartarson, A., Bodvarsson, G., & Steingrímsson, B. (2003). Development of a 3-D geothermal reservoir model for the greater Hengill volcano in SW-Iceland. *Lawrence Berkeley National Laboratory*, 10.

Croucher, A. (2011). PYTOUGH: A Python scripting library for automating TOUGH2 simulations. *New Zeland Geothermal Workshop Proceedings*, (p. 6). Auckland.

Croucher, A., & O'Sullivan, M. (2008). Application of the computer code TOUGH2 to the simulation of supercritical conditions in geothermal systems. *Geothermics*, 622-634.

Finsterle, S. (2015). *iTOUGH2 V7.0 Command Reference.* California: Lawrence Berkeley National Laboratory.

Haukwa, C. (1999). *A mesh creating program for integral difference method.* California: Ernest Orlando Lawrence Berkeley National Laboratory.

Jiménez, J. (2019). *Numerical modelling of Ahuachapán geothermal field, El Salvador.* Iceland: UNU Geothermal Training Programme.

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

Newson, J., Mannington, W., Sepulveda, F., Lane, R., Pascoe, R., & Clearwater, E. &. (2012). The Application of 3D Modelling and Visualisation Software to Reservoir Simulation: Leapfrog and TOUGH2. *Thirty-Seventh Workshop on Geothermal Reservoir Engineering* (p. 6). Stanford, California: Stanford University.

Pan, L. (2003). Wingridder --- An interactive Grid Generator for TOUGH2. *Proceedings of TOUGH Symposium* (p. 6). Berkeley: Lawrence Berkeley Laboratory.

Pruess, K. (2004). The TOUGH2 codes - A Family of Simulation Tools for Multiphase Flow and Transport Processes in Permeable Media. *Vadose Zone Jounal, 3*, 738-746.

Pruess, K., Oldenburg, C., & Moridis, G. (2012). *TOUGH2 User's Guide, Version 2.* California: Earth Sciences Division, Lawrence Berkeley National Laboratory.

Tómasdóttir, S. (2018). *Flow Paths in the Húsmúli Reinjection Zone, Iceland.* Uppsala: Uppsala University.

Vatnaskil Consulting Engineers. (2015, August 24). *STEINAR :: PRE- AND "POST"PROCESSOR FOR TOUGH2 MODEL MESHES.* Retrieved from http://www.vatnaskil.is/steinar

Yamamoto, H. (2008). A Graphical User Interface for the TOUGH2 Family of The TOUGH2 Family MultiPhase Flow and Transport Codes. *Ground Water, 46(4)*, 525-528.

Yeh, A., Croucher, A., & O'Sullivan, M. (2013). TIM- Yet another graphical tool for TOUGH2. *Conference: 35th New Zealand Geothermal Workshop.* Rotorua, New Zealand.