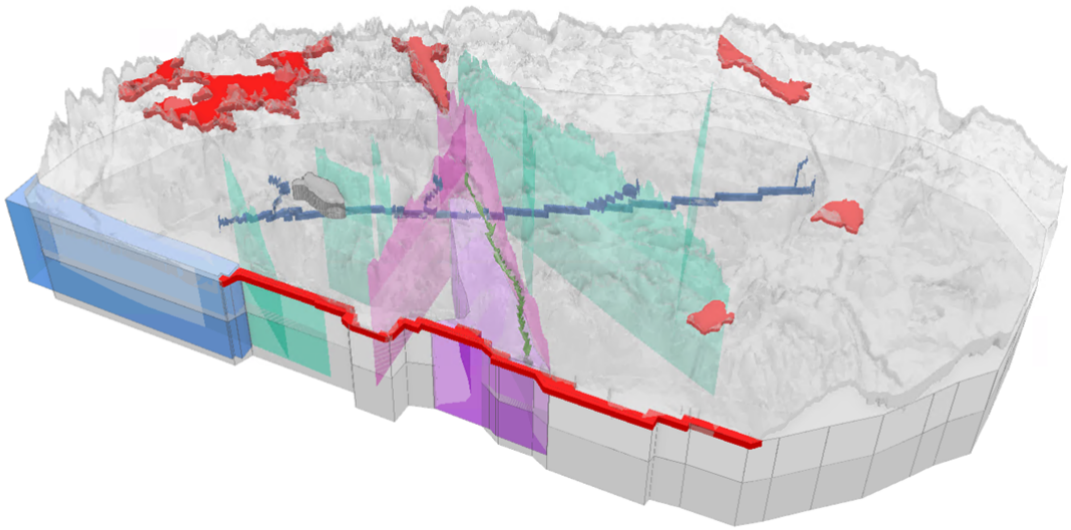
CHALMERS
UNIVERSITY OF TECHNOLOGY



Evaluating an Automated Approach for Groundwater Model Setup in Sweden

Master's Thesis in Infrastructure and Environmental Engineering

EBBA AMBY

DEPARTMENT OF ARCHITECTURE AND CIVIL ENGINEERING

CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2025
www.chalmers.se

Master's thesis 2025

Evaluating an Automated Approach for Groundwater Model Setup in Sweden

A Comparative Study of HydroModPy and MODFLOW

EBBA AMBY

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Architecture and Civil Engineering
Division of Geology and Geotechnics
Engineering Geology
Chalmers University of Technology
Gothenburg, Sweden 2025

Evaluating an Automated Approach for Groundwater Model Setup in Sweden
EBBA AMBY

Supervisors: Ezra Haaf, Department of Architecture and Civil Engineering
Andreas Hagberg, Sweco AB
Examiner: Lars Rosén, Department of Architecture and Civil Engineering

Master's Thesis 2025
Department of Architecture and Civil Engineering
Division of Geology and Geotechnics
Engineering Geology
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: The base model structure, including boundary conditions, layer discretization, and tunnel drainage (Sweco, 2024). The vertical scale is exaggerated by a factor of 10.

Typeset in LaTeX
Gothenburg, Sweden 2025

Evaluating a New Python-Based Approach for Groundwater Modeling in Sweden
EBBA AMBY
Department of Geology and Geotechnics
Chalmers University of Technology

Abstract

Groundwater modelling is a complex and time-consuming process, especially in regions with detailed topography and heterogeneous geology. Therefore, this thesis evaluates and compares two modelling approaches - a traditional MODFLOW-based workflow using ModelMuse, and an automated framework using HydroModPy. The comparison is applied to a real-world case study of the Kolmården Tunnel, a section of the Ostlänken high-speed railway project in Sweden.

The aim was to evaluate whether HydroModPy can replicate key behaviours of an expert-built MODFLOW model and assess its adaptability to Swedish input formats and modelling practices. Both models were implemented from the same input data, and their development, calibration, and outputs were systematically compared based on hydraulic head, boundary flows, and model performance at selected observation points. HydroModPy offered valuable automation for pre-processing and grid generation but lacked flexibility for custom geological layering, user-defined grids, and site-specific boundary conditions unless custom code modifications were made. In contrast, ModelMuse-based workflow provided full control over model architecture and produced more stable simulation results.

The study concludes that HydroModPy is a promising tool for automated model development and is capable of handling complex groundwater systems - provided the user has sufficient programming expertise. Without this, its application in detailed studies remains limited. The project further highlights the importance of early expert involvement in data interpretation and model design. These findings are relevant to model developers aiming to streamline model construction and researchers evaluating automation tools for hydrogeological impact assessments in complex infrastructure projects, such as tunneling in heterogeneous geological environments.

Keywords: hydrogeology, groundwater modelling, numerical model, MODFLOW, HydroModPy, Python.

Acknowledgements

I would like to thank my academic supervisor at Chalmers, Ezra, for your support, guidance, and valuable feedback throughout this thesis. A special thanks to Andreas Hagberg at Sweco for your practical support and for providing access to relevant data and expertise. I also thank David Ström for insights on MODFLOW, and Niclas Bockgård from Trafikverket for providing key documentation. Lastly, I acknowledge the developers of HydroModPy for making this work possible through their open-source framework.

Ebba Amby, Gothenburg, June 2025

List of Acronyms

Below is the list of acronyms that have been used throughout this thesis listed in alphabetical order:

CHD	Constant Head (MODFLOW Boundary Condition)
DEM	Digital Elevation Model
DIS	Discretization Package (MODFLOW)
DISV	Discretization with Vertices (Unstructured MODFLOW Grid)
DRN	Drain (MODFLOW Boundary Condition)
EPSG	European Petroleum Survey Group (Coordinate Reference System Standard)
GIS	Geographic Information System
GUI	Graphical User Interface
JSON	JavaScript Object Notation
MODFLOW	Modular Finite-Difference Groundwater Flow Model
NWT	Newton Solver for MODFLOW
Ostlänken	Eastern Link (Swedish High-Speed Railway Project)
RCH	Recharge (MODFLOW Boundary Condition)
SGU	Sveriges Geologiska Undersökning (Swedish Geological Survey)
SMHI	Swedish Meteorological and Hydrological Institute
USGS	United States Geological Survey
YAML	Yet Another Markup Language

Nomenclature

Below is the nomenclature of indices, sets, parameters, and variables that have been used throughout this thesis.

Parameters

q	Specific discharge [m/s]
K	Hydraulic conductivity [m/s]
h	Hydraulic head [m]
t	Time
x, y, z	Independent spatial variables [m]
W^*	Volumetric inflow rate from sources and sinks
S_s	Specific storage (volume of water released from storage per unit change in head, h)
q_x, q_y, q_z	Flux/flow rate in different axes
K_x, K_y, K_z	Principal components of the hydraulic conductivity tensor [m/s]
$\frac{\partial h}{\partial x}, \frac{\partial h}{\partial y}, \frac{\partial h}{\partial z}$	Components of the head gradient [m/m]

Contents

List of Acronyms **ix**

Nomenclature **xi**

List of Figures **xv**

List of Tables **xvii**

1 Introduction **1**
1.1 Aim . . . 3
1.1.1 Research Questions . . . 4
1.1.2 Scope & Limitations . . . 4

2 Methodology **5**
2.1 Model Development Using Graphical User Interface . . . 5
2.2 Script-Based Model Development . . . 7
2.2.1 Modelling with HydroModPy Framework . . . 7
2.2.2 Model Development using FloPy . . . 9

3 Case Study Implementation **11**
3.1 Case Description & Study Area . . . 11
3.2 Base Model Setup in ModelMuse . . . 13
3.2.1 Hydraulic Boundary Conditions . . . 13
3.2.2 Model Discretization . . . 15
3.2.3 Hydraulic Properties . . . 16
3.2.4 Solver Configuration and Numerical Settings . . . 16
3.3 Model Setup in HydroModPy . . . 17
3.3.1 Hydraulic Boundary Conditions . . . 17
3.3.2 Model Discretization and Hydraulic Parameters . . . 18
3.3.3 Solver Configuration and Numerical Settings . . . 18
3.3.4 Key Differences Between the Models . . . 20

4 Results & Discussion **21**
4.1 HydroModPy Model Implementation . . . 21
4.2 Simulation Performance and Results . . . 22
4.2.1 Calibration . . . 23
4.3 Workflow & Usability Evaluation . . . 26

4.4 Recommendations for Future Work 29

5 Conclusion **31**

Bibliography **33**

A Appendix 1 - Python Script **I**

B Appendix 2 - NRMSE Calculation on ModelMuse Model **III**

C Appendix 3 - NRMSE Calculation on HydroModPy Model 1.0 **V**

List of Figures

1.1 Development steps of a groundwater model. A workflow inspired by Anderson et al., 2015. 2

2.1 An illustration of an discretized artificial aquifer (Water Resources Mission Area (USGS), 2018). 6
2.2 Global workflow of HydroModPy (Gauvain et al., 2025). 8

3.1 Map of the Stavsjö–Loddby section. 12
3.2 A cross-sectional conceptual model illustrating the geology in the area. 13
3.3 Overview of the model domain together with active boundary conditions (Sweco, 2024). 15
3.4 Left: a uniformly structured grid. Right: a locally refined grid (Sweco, 2024). 15

4.1 Calibration of the base scenario with NRMSE = 2.5% (Sweco, 2024). 24
4.2 Calibration of the HydroModPy model with NRMSE = 64.3%. 24
4.3 Comparison of simulated and observed hydraulic heads at 22 observation wells. Open circles represent the reference model (ModelMuse); crosses represent the HydroModPy model. 25

List of Tables

3.1 Constant head boundaries applied to lakes and sea (SMHI, n.d.). . . . 13
3.2 Overview of drain boundary conditions used in the model. 14
3.3 Hydraulic conductivity (K_{soil}) values for different soil types. 16
3.4 Hydraulic conductivity (K_{rock}) for bedrock depending on depth. . . . 16
3.5 Solver settings for MODFLOW 6 simulation. 17
3.6 Solver settings for MODFLOW-NWT simulation with HydroModPy. 19
3.7 Overview of key differences between the reference and HydroModPy models 20

4.1 Comparison of boundary flows and mass balance between the two models 23

1

Introduction

Groundwater flow modelling is a critical component of infrastructure planning, particularly for projects involving tunnels, deep excavations, or construction in hydrogeologically sensitive areas. Numerical models are used to understand groundwater systems, estimate aquifer properties, and predict site-specific responses to construction or environmental change (Anderson et al., 2015). These models are essential for assessing potential leakage, supporting environmental impact assessments, and designing mitigation strategies. As modelling complexity increases, with detailed topography, heterogeneous geology, and large spatial datasets, the need for reproducible and efficient modelling workflows becomes more critical, making the choice of framework increasingly important (Leaf and Fienen, 2022).

A typical groundwater modelling workflow involves several steps: conceptual model development, numerical model setup, calibration, and uncertainty analysis, see Figure 1.1. These steps involve defining system boundaries, hydraulic properties, and water inputs and outputs, usually based on geomorphological, geological, and hydrological data (Anderson et al., 2015). Errors in boundary condition definitions can significantly affect model accuracy, emphasizing the importance of careful model design and calibration.

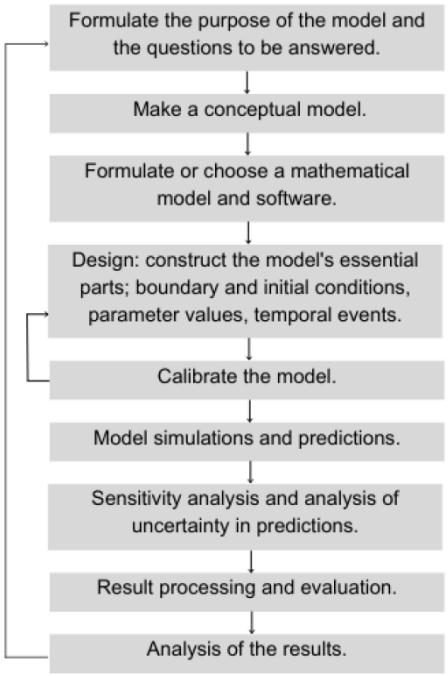


Figure 1.1: Development steps of a groundwater model. A workflow inspired by Anderson et al., 2015.

In Sweden, the importance of robust and transparent groundwater modelling has been highlighted by recent evaluations of national practice. A comprehensive study by Höglund et al. (2024) found that while MODFLOW is the dominant modelling platform in both research and consultancy, key aspects such as uncertainty analysis, model calibration transparency, and documentation are often insufficient in practice. This can lead to delays in permit approvals, difficulties in communicating model reliability to stakeholders, or challenges in defending impact assessments during legal or public review processes, particularly in large infrastructure projects (Höglund et al., 2024).

MODFLOW is a modular finite-difference groundwater flow model developed by the U.S. Geological Survey (USGS) (USGS, 2024). It remains the standard for groundwater simulation due to its long-standing development, extensive community support, and compatibility with both graphical and script-based tools (Höglund et al., 2024). Several modelling frameworks have been developed to support MODFLOW, ranging from intuitive graphical user interfaces (GUIs) to fully script-based environments. GUI-based tools, such as ModelMuse, Visual MODFLOW, GMS, and Groundwater Vistas, simplify model construction, input management, and visualization, and are particularly useful for conceptual model development and quality control in early project phases (Barlow and Harbaugh, 2006;Winston, 2022).

However, GUI-based workflows can become troublesome when dealing with large datasets or when performing repetitive tasks such as sensitivity analyses or scenario testing. As a result, script-based tools like FloPy and HydroModPy have emerged, enabling automation, reproducibility, and direct integration with Python (Bakker

et al., 2016).

FloPy, maintained by the USGS, acts as a Python wrapper around MODFLOW and allows users to create, run, and visualize models programmatically (Hughes, Langevin, Paulinski, et al., 2024). However, it does not include built-in preprocessing for geospatial data, which needs to be handled externally (e.g., with GDAL or custom scripts). HydroModPy extends this concept by offering an additional abstraction layer that automates many of these tasks. It integrates terrain processing (e.g., watershed delineation), raster-based geological attribution, and the application of MODFLOW boundary conditions, all within a modular Python-based workflow (Gauvain et al., 2025). Compared to FloPy, HydroModPy is more streamlined, prioritizing automation and reproducibility over flexibility. However, this comes at the cost of reduced support for detailed user customization.

HydroModPy is evaluated in this thesis for the first time using Swedish input data and a site-specific infrastructure case. This provides an opportunity to assess its adaptability to new data formats and contexts outside of its original French implementation environment.

This thesis investigates the applicability of HydroModPy to Swedish hydrogeological conditions using a case study of the Stavsjö–Loddby section of the Ostlänken (Eastern Link) high-speed railway project (Trafikverket, 2023a). The study area includes an 8-kilometre tunnel that passes through complex geological and hydrological environments, including fault zones, deep bedrock valleys, and hydraulically sensitive areas such as the Getå stream and Lake Skiren (Trafikverket, 2024). This site was selected primarily because a high-resolution, expert-built MODFLOW 6 model already exists for the area, providing a valuable benchmark for evaluating HydroModPy's performance.

Although the existing model was developed using a GUI-based workflow in ModelMuse, this study explores whether a script-based, automated approach using HydroModPy can replicate key aspects of model structure, behaviour, and calibration accuracy, and how the framework performs when adapted to Swedish datasets and modelling standards.

1.1 Aim

The aim of this thesis is to assess the performance of an automated model setup workflow, implemented through HydroModPy, in developing groundwater flow models for subsurface infrastructure applications. Constructing such models is typically a resource-intensive and iterative process, requiring both hydrogeological expertise and technical precision. Groundwater models for subsurface projects such as tunnels place high demands on spatial resolution, vertical discretization, and accurate boundary conditions, since small errors can lead to major implications for inflow risk assessments, permits, and mitigation design. The use of automated tools, such as HydroModPy, may offer a more efficient alternative to traditional manual work-

flows, potentially enabling faster and more reproducible model construction without compromising on output quality.

This thesis presents a comparative analysis of modelling efficiency and performance between HydroModPy and a conventional expert-built MODFLOW model, using a case study from the Kolmården Tunnel section of the Ostlänken high-speed railway project in Sweden.

1.1.1 Research Questions

- How does HydroModPy compare to a traditional GUI-based setup in groundwater modelling with MODFLOW?
- What are the strengths and limitations of HydroModPy when applied to Swedish hydrogeological conditions in general and for subsurface infrastructure projects in particular?

1.1.2 Scope & Limitations

- The project is limited to data from previous investigations and existing records.
- The focus will be on the comparison of HydroModPy's capability with an existing expert-built model, rather than on providing detailed quantitative predictions of groundwater inflow rates during construction. The expert-built model stands as a benchmark model for this project.
- The project will not address specific mitigation measures for groundwater management during the construction of the Kolmården Tunnel.

2

Methodology

This chapter outlines the methodological framework used to develop and evaluate the groundwater model setups presented in this thesis. Two parallel modelling workflows were implemented: one based on a traditional graphical user interface (GUI) using ModelMuse and MODFLOW 6, and one using a script-based, automated approach through HydroModPy and MODFLOW-NWT. A third approach using FloPy is also briefly discussed to highlight its role of Python-based groundwater modelling tools.

2.1 Model Development Using Graphical User Interface

ModelMuse is the GUI used in this study as the reference modelling environment. It is based on a "design by objects" approach, which allows model features to be defined conceptually and independently from the computational grid, increasing flexibility in early model stages and reducing the risk of input errors (Basharat and Jawad (2023) and Winston (2019)).

The modelling process begins with project initialization, where a solver is selected. The model domain is then defined either by importing spatial data (e.g., shapefiles) or by manually sketching boundaries within the interface. A structured grid is generated, and vertical discretization is specified through the number of layers and their corresponding elevations, which can be derived from raster datasets like digital elevation models (DEMs). Figure 2.1 illustrates an example of a structured model grid.

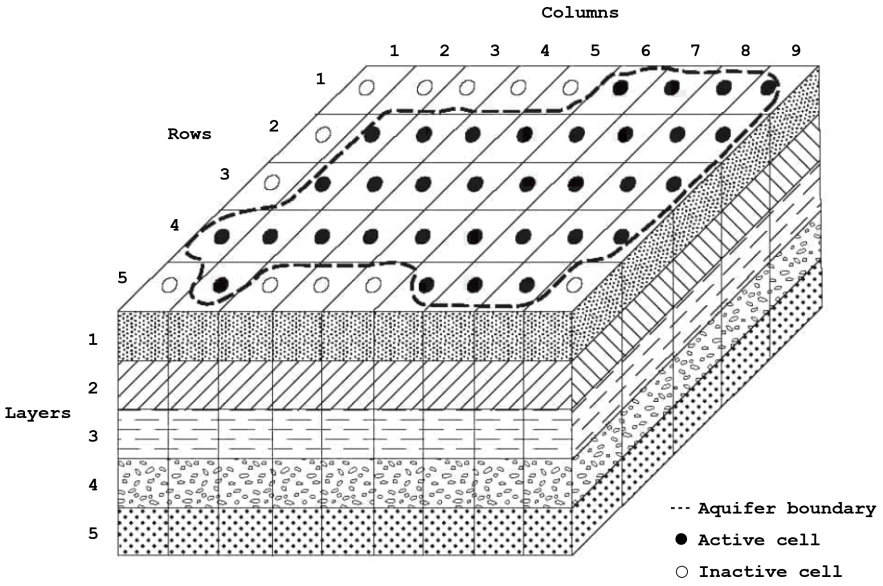


Figure 2.1: An illustration of an discretized artificial aquifer (Water Resources Mission Area (USGS), 2018).

Subsequently, geological and hydrological inputs are assigned, including the hydrostratigraphy, which may be based on either raster or vector data. In ModelMuse, spatial data can be imported directly as shapefiles, raster grids (e.g., GeoTIFF), or manually drawn features. The software supports layering and attribute-based zoning by allowing the user to define hydrogeological units based on map geometry and assign properties such as hydraulic conductivity, specific yield, or storage coefficients through logical expressions or spatial conditions (Basharat and Jawad, 2023).This enables fine-grained control, for example, different conductivity values can be assigned to the same soil type depending on location, depth, or layer number. Zones can also overlap or be prioritized based on order, giving the modeller full flexibility when handling geological complexity. Hydraulic properties, such as hydraulic conductivity, specific yield, and storage coefficient, are attributed to each layer or material zone. Recharge zones and rates are defined spatially, often using map-based inputs. Recharge zones and rates are also defined spatially, either as uniform values or as raster-based distributed inputs. These features allow ModelMuse to handle complex geological and hydrological environments in a flexible and transparent way, particularly during early conceptualization and calibration stages.

Boundary conditions in MODFLOW are configured using standardized packages, regardless of the interface or tool used. These packages form the core structure of all MODFLOW models, whether built using a GUI like ModelMuse or programmatically using FloPy or HydroModPy. For example, the Basic Package (BAS) defines active and inactive cells and initial heads; the Discretization Package (DIS or DISV) handles spatial and temporal resolution; and the Newton Solver Package (NWT) is used to address non-linearities in unconfined conditions. Boundary-specific packages include the Constant Head Package (CHD), often used to simulate lakes or

coastal interfaces; the Drain Package (DRN), which models surface water discharge via streams or ditches; and the Recharge Package (RCH), used to apply spatially distributed or uniform groundwater recharge (Anderson et al., 2015;Hughes, Langevin, and Banta, 2024). These components are implemented consistently across different modelling environments but are set up through different interfaces depending on the tool used.

Model calibration is performed by inputting observed groundwater levels and adjusting parameters to minimize discrepancies between simulated and measured data. Once the setup is complete, ModelMuse exports the MODFLOW input files and launches the MODFLOW engine as an external process. Although the simulation itself is handled by the MODFLOW executable, ModelMuse integrates the process, allowing the user to monitor convergence and directly postprocess results such as hydraulic heads, drawdowns, and flow paths within the interface (Winston, 2022). Outputs can also be exported for further analysis.

The GUI streamlines data management and ensures consistency between model inputs and geometry. Both GUI, and script-based MODFLOW workflows support scenario testing, for example, evaluating groundwater responses to stressors such as pumping, infrastructure development, or climate variability. ModelMuse facilitates this process through its visual interface, which can make defining and managing multiple scenarios more intuitive, especially during early project phases (Leaf and Fienen, 2022).

2.2 Script-Based Model Development

The methodology described in this section is based on Gauvain et al. (2025), which outlines the HydroModPy framework for constructing watershed-scale groundwater models. FloPy was not used directly in this thesis, but serves as the computational foundation for HydroModPy. It is a Python package developed by the USGS that enables full scripting of MODFLOW models. HydroModPy builds on top of FloPy and automates many of the steps required to construct a working model.

This section provides a brief background on FloPy to clarify what functionality HydroModPy inherits and where the two tools differ.

2.2.1 Modelling with HydroModPy Framework

There are various modelling tools available to provide frameworks for developing hydrogeological models. However, accurately adapting them to a specific site remains complex, requiring careful consideration of assumptions, data, and methods (Gauvain et al., 2025). HydroModPy was initiated in 2018, and its development was driven by two primary objectives:

1. Automating the extraction and discretization of watersheds from DEMs, while simultaneously integrating relevant data (e.g., geology, hydrography, piezom-

etry) from available databases.

2. Facilitating the visualisation and comparison of results from different modelling programs within a single platform.

The workflow and code development are structured around five main components: (1) extraction of the watershed to define the model domain, (2) development of a conceptual model and retrieval of public and/or private data, (3) parametrization of the hydrogeological model, (4) simulation of groundwater flow and particle tracking, and (5) computation and visualisation of outputs (Gauvain et al., 2025). The stages of this workflow are illustrated in Figure 2.2.

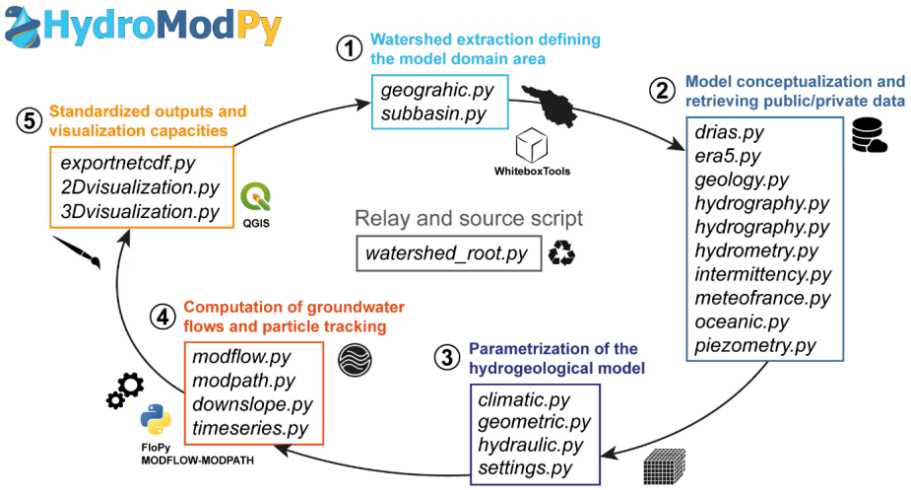


Figure 2.2: Global workflow of HydroModPy (Gauvain et al., 2025).

HydroModPy allows users to incorporate additional layers of hydrogeological information through a suite of dedicated modules. The model domain is first defined using the Geographic module, where a watershed or catchment area is extracted from a DEM. The input DEM file, typically in .tif format, determines the spatial resolution of the model, as each grid cell corresponds to a pixel defined by the DEMs cell size.

Following domain definition, the model is conceptualized by assigning input data across various thematic modules, as illustrated in Figure 2.2 (step 2). The Geology module rasterizes lithological units and assigns hydrostratigraphic codes across the model grid, enabling the differentiation of subsurface properties. Surface water features such as rivers, lakes, and wetlands are incorporated through the Hydrography module; these features are primarily used to assign drain (DRN) boundary conditions based on hydrological characteristics, while constant head (CHD) boundaries, such as sea-level interfaces, are handled separately.

Streamflow rates and stream intermittency observations are integrated through the Hydrometry and Intermittency modules. These inputs allow for the calibration of surface-groundwater interactions and provide additional constraints on model behaviour. Groundwater recharge and run off conditions are assigned via the Climatic

module, which supports both uniform and spatially distributed recharge, as well as time-variable inputs for transient simulations.

The Oceanic module facilitates the imposition of sea-level or other constant head boundaries, initializing the CHD package particularly for coastal areas. Observed groundwater levels (piezometric data) can be incorporated through the Piezometry module, allowing for model calibration against measured heads.

Hydraulic properties are parametrized through the Hydraulic module, where hydraulic conductivity, specific yield, specific storage, and vertical anisotropy can be assigned. HydroModPy supports spatial heterogeneity and vertical variations in hydraulic properties, including options for applying depth-dependent decay functions to better simulate realistic aquifer behaviour.

After the conceptual model is fully defined, the MODFLOW module assembles all model components into a complete MODFLOW-NWT or MODFLOW-2005 simulation setup. This includes grid discretization, initial heads, boundary conditions (DRN, RCH, CHD), stress period definitions, and solver settings. Users can configure both steady-state and transient simulations within the same framework, with model input files generated automatically based on the prior modules.

Finally, HydroModPy provides visualization tools, enabling users to plot DEMs, geology, boundary conditions, groundwater heads, and flow directions.

2.2.2 Model Development using FloPy

FloPy is a Python library developed by the USGS for programmatic construction and execution of MODFLOW models (Hughes, Langevin, Paulinski, et al., 2024). While FloPy was not used directly in this study, it plays a key role in the modelling workflow, as HydroModPy relies on FloPy as its computational engine. Understanding the functionality of FloPy is important for assessing what HydroModPy automates, and what limitations that may arise.

FloPy enables users to define model geometry, discretization, and all boundary and hydraulic parameters directly in Python code. It supports both structured (DIS) and unstructured (DISV) grids, allowing flexible spatial resolution. Model development typically begins with defining the domain, often using DEMs to set the top surface and geological data to determine bottom elevations. In settings with complex topography or subsurface conditions, unstructured grids may be used to better represent spatial heterogeneity (Hughes, Langevin, Paulinski, et al., 2024).

Spatial and temporal data are assigned via arrays that match the model grid. These inputs include surface elevation, stratigraphy, recharge rates, hydrographic features, pumping schedules, and well locations. Temporal variation is handled through stress period utilities that allow configuration of both steady-state and transient simulations.

Hydraulic properties, such as, hydraulic conductivity, specific yield, specific storage,

and anisotropy, are also input as spatially distributed arrays. FloPy supports both homogeneous and heterogeneous assignment, enabling realistic aquifer representation.

Boundary conditions are applied using standard MODFLOW packages. Common examples include the Constant Head Package (CHD), Drain Package (DRN), General Head Boundary (GHB), and Recharge Package (RCH). Initial heads are based on observed or interpolated data.

Solver configuration is done through MODFLOW classes such as MODFLOW-NWT, PCG, or SMS, with options to set convergence criteria, iteration limits, damping factors, and more. Once the model is assembled, FloPy generates the required input files and can launch the simulation directly from the Python environment.

Although not used directly in this thesis, the scripting structure of FloPy supports reproducibility and model transparency. Extensions such as MODFLOW-setup, allow users to define entire configurations using structured YAML or JSON files, enabling automated model generation and version-controlled development (Leaf and Fienen, 2022). However, FloPy does not offer built-in tools for geospatial preprocessing. Tasks such as raster resampling, shapefile processing, or 3D grid construction must be handled externally using packages like GDAL, rasterio, or pandas.

3

Case Study Implementation

This section describes the development of a groundwater flow model for a real-world infrastructure case study using the HydroModPy framework. The modelling process followed the general workflow established by (Gauvain et al., 2025), but was adapted to accommodate site-specific data characteristics and project requirements. As a reference for comparison, an existing baseline model of the same area, developed using MODFLOW 6 in ModelMuse is also presented (Sweco, 2024).

3.1 Case Description & Study Area

Trafikverket (Swedish Transport Administration) has begun building a new high speed railway. Starting with, Ostlänken, a double-track high-speed railway stretching approximately 160 kilometers between Järna, south of Stockholm, and Linköping in Sweden (Länsstyrelsen Östergötland, 2024). The Stavsjö-Loddby section of the Ostlänken, high-speed railway project in Sweden spans approximately 15 kilometres, running from Södermanland county border to the northern entrance of Norrköping. This route crosses the Kolmården forest and nature reserve, a region of high ecological and recreational value. A major component of this section is the Kolmården tunnel, an 8-km long underground passage designed to minimize surface disruption, see Figure 3.1 (Länsstyrelsen Östergötland, 2024).

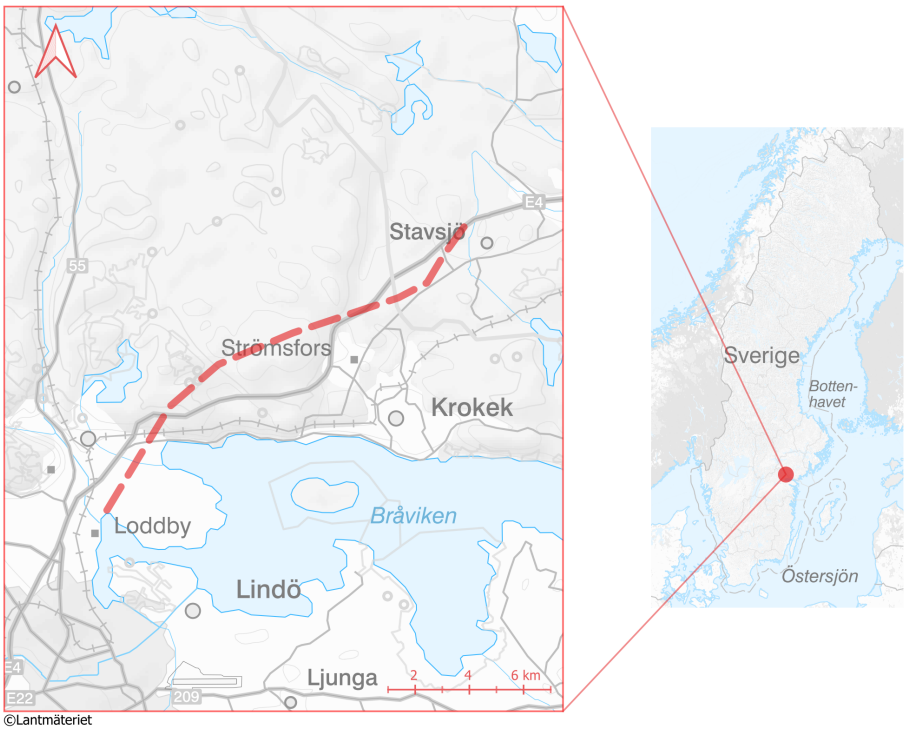


Figure 3.1: Map of the Stavsjö–Loddby section.

Despite mitigation measures, the project presents significant environmental challenges. The tunnel which will be situated 80 to 146 meters below ground level, cannot be entirely sealed against groundwater inflow. This may result in leakage that reduces local groundwater levels and impacts sensitive water-dependent environments such as Getå stream and Lake Skiren (Trafikverket, 2023b). These systems rely on continuous groundwater input to sustain regular flow and water levels. Leakage from the tunnel could reduce median flow and potentially cause local drawdown of the groundwater level in this area.

The Kolmården area is topographically diverse, with elevations ranging from 100 to 155 meters above sea level (Sweco, 2024). The landscape is dominated by hills and shallow bedrock outcrops, generally with only thin layers of overlying soil such as sand, moraine, peat, or mixed sediments. Agricultural land, residential areas and minor roads are typically located in the low-lying valleys between bedrock ridges.

A major geological feature is the Getå ravine, a deep valley carved into a fault zone with soil depths of up to 30 meters. Here, the valley fill consists primarily of silt and clay, while glaciofluvial sediments exposed along the stream base facilitate substantial interaction between groundwater and surface water. This highly conductive material acts as a local aquifer. The underlying bedrock consists mainly of gneiss and granite, while the ravine itself is influenced by sedimentary rocks such as vacka (Tranviks Udde Fastigheter AB, 2023).

Recharge primarily occurs in the northern part of the Getå valley where coarse sediments are present, and also through fractured zones in the bedrock. Figure 3.2

illustrates a cross-sectional geological interpretation of the area.

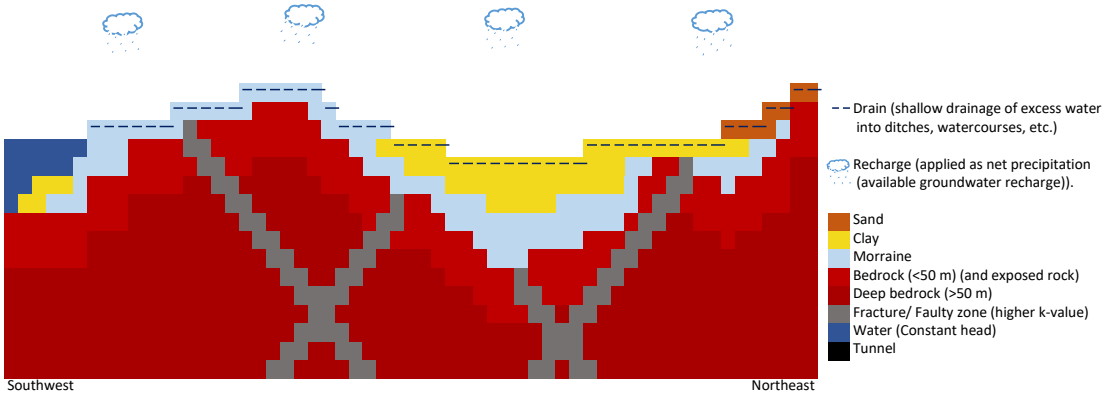


Figure 3.2: A cross-sectional conceptual model illustrating the geology in the area.

3.2 Base Model Setup in ModelMuse

The base groundwater flow model was constructed and simulated using the numerical code MODFLOW 6 (Hughes, Langevin, and Banta, 2024) in the graphical user interface ModelMuse 5.0 (Winston, 2022). The GUI is supported with the latest version of MODFLOW and utilizes a generalized finite-difference method with controlled flow volume discretization (CFVD). The model was constructed with locally refined grids to increase spatial resolution in areas of particular interest, such as around the tunnel, lakes, and fault zones. This approach allows finer discretization near key features without increasing the computational cost of the entire model domain. Grid refinement was implemented manually in the GUI by assigning smaller cell sizes in zones with greater expected hydraulic gradients or data density.

3.2.1 Hydraulic Boundary Conditions

Large lakes in the area were assumed to function as positive hydraulic boundaries, meaning full hydraulic contact between the bedrock and the lake, and were simulated using the "CHD" boundary condition. Water levels in the lakes were set to the mean groundwater level based on open data from SMHI, see Table 3.1 (SMHI, n.d.).

Table 3.1: Constant head boundaries applied to lakes and sea (SMHI, n.d.).

Water Body	Head Level (m, RH2000)
Bärsjön	87
Bråviken	0 (assumed sea level)
Lilla Älgsjön	118
Nedre Glottern	80
Torsjön	68

A general "DRN" boundary condition was applied and set to 0.5 meters below ground level to simulate shallow drainage of precipitation into ditches, streams, and lakes. The purpose of this general drain is to handle discharge in outflow zones where groundwater recharge is expected to be negligible, and the groundwater table reaches the ground surface. This boundary condition consists of a specified drainage elevation and a conductance value. Conductance numerically represents the flow resistance between the drain and surrounding model cells. In this case, a relatively high conductance value was assigned to ensure that the flow resistance does not limit the ability of water to exit the system via the drain (Sweco, 2024).

In the Getå Ravine, the Getå stream was implemented as a "DRN" boundary condition at the base of the ravine. The boundary was applied to a presumed lower aquifer, as observations in the area indicate that the stream is in hydraulic contact with the friction soil. The drainage level was set to +84.4 meters, corresponding to the estimated mean water level in Lake Skiren, and a high conductance value was applied, as with the general drainage boundary. The "DRN" boundary conditions are summarised in Table 3.2.

Table 3.2: Overview of drain boundary conditions used in the model.

Drain Type	**Elevation Description**	**Conductance**
General Drainage	Ground surface −0.5 m	8.64 m^2/day
Lake Skiren	Equal to surface elevation	86.4 m^2/day
Getå Stream	Model top −7 m (adjusted to observations)	8.64 m^2/day
Tunnel Drainage	Varies along tunnel alignment	86.4 m^2/day

A "RCH" boundary condition was applied to the top layer to simulate groundwater recharge across the model area. Groundwater recharge was estimated as 200 mm/year, derived from long-term net precipitation data provided by the Swedish Meteorological and Hydrological Institute (SMHI) (SMHI, n.d.). This boundary condition applies a constant inflow to the designated cells. The actual infiltration of recharge into the model depends on the hydraulic conductivity of individual cells, the presence of drainage conditions, and topography. This boundary condition is not applied to cells that already have a "CHD" condition assigned, as this is not permitted in the MODFLOW code.

The models outer boundaries are represented, where possible, by no-flow boundaries, aligned with topographical groundwater divides. "CHD" boundaries were used along lakes and streams. The bottom of the model is defined as a no-flow boundary, meaning that no flow is allowed into or out of the model from the bottom layer. The model domain, together with corresponding boundary conditions are illustrated in Figure 3.3.

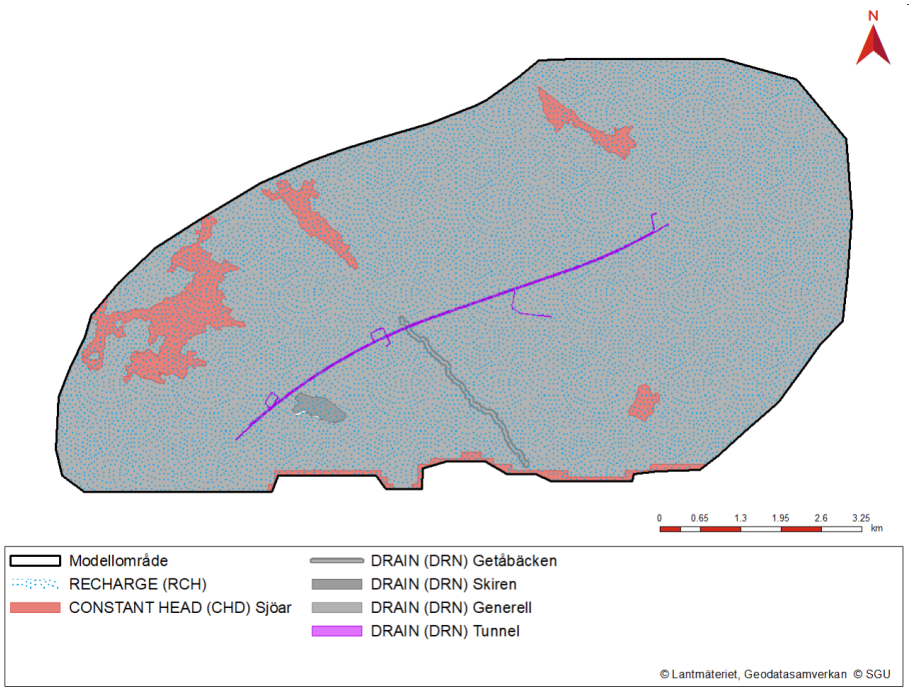


Figure 3.3: Overview of the model domain together with active boundary conditions (Sweco, 2024).

3.2.2 Model Discretization

The model projection used was SWEREF 99 TM (EPSG:3010). The model was constructed as a three-dimensional matrix consisting of three-dimensional cells, or voxels, with varying sizes. The voxel size depends on the distance to the tunnel and proximity to hydraulic boundary conditions. Closest to the tunnel and access shafts, the cell size is approximately 10 meters × 10 meters. Cells near lakes and fault zones are set to 20 meters × 20 meters, while the remaining parts of the model use a coarser resolution of 40 meters × 40 meters. This type of locally refined gridding follows a quadtree refinement approach. This is used when a higher resolution is needed in particularly interesting areas without risking disruption in overall model performance, see Figure 3.4.

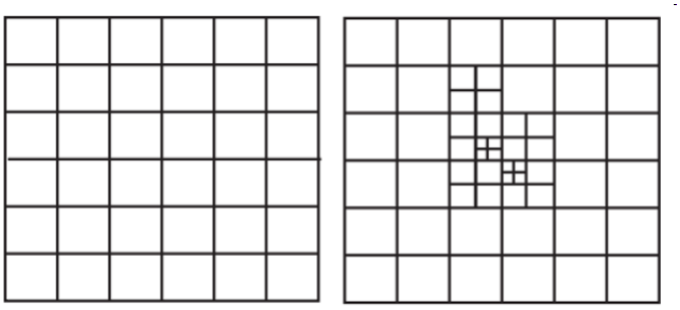


Figure 3.4: Left: a uniformly structured grid. Right: a locally refined grid (Sweco, 2024).

The total model domain covers an area of approximately 67 km^2. The top surface

of the model is represented by high-resolution elevation data from Lantmäteriet (the Swedish mapping, cadastral and land registration authority). The DEM was resampled to 10 meters × 10 meters to match the smallest cell size in the model grid.

The bottom boundary of the model consists entirely of bedrock, set to a uniform elevation of -150 meters (RH2000).

In this model setup, hydraulic conductivity in the bedrock is assigned based on a statistically derived isotropic mean value, representing a simplified but representative estimate of hydraulic properties in the rock.

Vertically, the model consists of four layers: soil (upper aquifer), soil (lower aquifer), bedrock with higher hydraulic conductivity, and bedrock with lower hydraulic conductivity. Each layer is assigned distinct hydraulic parameters to reflect varying subsurface conditions. These layers enable a conceptual distinction between surficial deposits, fractured bedrock zones, and deeper, less permeable bedrock.

3.2.3 Hydraulic Properties

Hydraulic conductivity values were assigned to represent the heterogeneity of the subsurface. Soil and geological maps were used to link hydraulic properties to spatial zones within the model. Separate values were used for unconsolidated sediments and bedrock, as shown in Tables 3.3 and 3.4.

Table 3.3: Hydraulic conductivity (K_{soil}) values for different soil types.

Soil Type	K_{soil} **(m/s)**
Clay	1×10^{-8}
Sand (and gravel)	1×10^{-5}
Till	5×10^{-7}
Exposed bedrock	2×10^{-7}

Table 3.4: Hydraulic conductivity (K_{rock}) for bedrock depending on depth.

Depth Range	K_{rock} **(m/s)**
$<$ 50 meters depth	2.50×10^{-8}
$>$ 50 meters depth	2.90×10^{-9}

The values for bedrock were applied using a depth-dependent function, to reflect decreasing permeability with depth due to fewer fractures and lower porosity in deeper rock layers.

3.2.4 Solver Configuration and Numerical Settings

The numerical solution was configured to ensure stability and convergence for the complex hydrogeological setting. The recharge was applied only to the top most active cells in accordance with MODFLOW 6 requirements. Solver settings are listed in Table 3.5.

Table 3.5: Solver settings for MODFLOW 6 simulation.

Setting	Value	Comment
Solution scheme	Complex	
Nonlinearity formulation	Newton	
Under-relaxation	Enabled	Dampens oscillations in the iterative process; stabilizes convergence
Outer iterations	50	
Outer convergence criterion	0.001	
Inner iterations	600	
Inner convergence criterion	0.0001	
Linear solver	BICGSTAB	Required for MODFLOW 6 compatibility
Cell_Type	0	

3.3 Model Setup in HydroModPy

The groundwater model was constructed using HydroModPy to evaluate hydrogeological responses, and to replicate the ModelMuse model, described above. The model was developed using MODFLOW-NWT as the numerical solver for steady-state simulation. Model development was structured according to HydroModPy's modular architecture and involved combination of automated routines that the package allows for and some customised scripting.

3.3.1 Hydraulic Boundary Conditions

The model was constructed using HydroModPy's modular architecture, where thematic input modules are used to define spatial and hydrogeological properties. The process began by structuring the project environment into separate folders for input data, outputs, and MODFLOW configuration files.

The Geographic module was used to define the model domain, based on a high-resolution DEM from Lantmäteriet and a shapefile outlining the outer watershed boundary. Geological information, including lithological types and fracture zones, was then imported and rasterized using the Geology module. Fracture zones were incorporated by assigning distinct hydrostratigraphic codes within the bedrock raster layer.

The Hydrographic module was used to define linear and polygonal surface water features, such as lakes and streams. These features were then used to generate MODFLOW boundary conditions: constant head boundaries (CHD) for hydraulically connected lakes, and drain (DRN) boundaries for shallow discharge zones and streams. Water levels for lakes were assigned based on available data (see Table 3.1).

Groundwater recharge was applied using the Climatic module, with a spatially uniform recharge rate of 200 mm/year. To avoid conflicts with CHD boundaries, a masking operation was applied to ensure that recharge was not assigned to CHD-

designated cells, as required by MODFLOW logic.

Drain conditions were specified for several zones, each with defined conductance values and elevations based on geological setting and target feature type (e.g., general drainage, Getå stream, Lake Skiren). These were assigned through a combination of HydroModPy's interface and manually adjusted input layers (Table 3.2).

3.3.2 Model Discretization and Hydraulic Parameters

All spatial data were projected to SWEREF 99 TM (EPSG:3010) to ensure consistency. The grid was constructed using a structured setup in HydroModPy. Due to performance constraints, the horizontal cell size was initially set to 10×10 meters near key infrastructure (e.g., tunnel, lakes) but was later coarsened to 50×50 meters across the domain.

Vertical discretization was defined using a custom bottom elevation array, developed manually by combining DEM data with borehole-derived depth-to-bedrock estimates. This array was constructed outside HydroModPy and then imported to overwrite the default layer generation.

Layer thicknesses were determined using a combination of fixed thresholds and exponential spacing logic. Five primary zones were defined, from shallow overburden to deep bedrock. The full logic for bottom elevation calculation is included in Appendix X.

Hydraulic properties were assigned using the Hydraulic module. These were spatially varied using rasterized geological inputs and included a depth-dependent decay function for hydraulic conductivity in the fractured bedrock. This decay function was implemented manually via custom scripts that updated the conductivity array prior to model execution.

3.3.3 Solver Configuration and Numerical Settings

HydroModPy configures a MODFLOW-NWT model using the Newton-Raphson formulation. This approach is well-suited for non-linear systems involving water table movement and dry cell handling. The COMPLEX setting is enabled to activate adaptive under-relaxation and damping, improving convergence behaviour in heterogenous and steep-gradient settings. Solver iterations in MODFLOW-NWT are controlled by the MAXITEROUT parameter, together with residual based stopping criteria using STOPTOL and tolerances for head and flow imbalances (headtol and fluxtol). The linear system is solved using the GMRES (Generalized Minimal Residual) method, which is robust for large systems in 3D groundwater models. See Table 3.6.

Table 3.6: Solver settings for MODFLOW-NWT simulation with HydroModPy.

Setting	**Value**	**Comment**
Solution scheme	Complex	
Nonlinearity formulation	Newton	
Under-relaxation	Enabled	Dampens oscillations in the iterative process; stabilizes convergence
Max outer iterations	5000	Ensures convergence in difficult cases
Head tolerance	1e-5 m	Convergence threshold for head change
Flux tolerance	100 m^3/day	Convergence threshold for flow imbalance
Linear solver	GMRES	Generalized minimal residual method
Dry cell handling	IBOTAV=1	Averaged head values in dry cells
Stop criteria	STOPTOL=1e-10	Ensures early termination when residual is small

While both model solvers share the same core goal, stabilizing and accelerating convergence in non-linear flow conditions, there are key differences that separate them:

- Iteration Handling: MODFLOW-NWT uses a combined approach with MAXITEROUT and global residual control, whereas MODFLOW 6 separates outer and inner iterations for finer control.

- Linear Solvers: GMRES (HydroModPy/MODFLOW-NWT) vs BICGSTAB (MODFLOW 6), both iterative Krylov methods, each with distinct convergence characteristics.

- Modularity: MODFLOW 6 allows for greater modularization and solver coupling, supporting multiple models (e.g., GWF-GWT), while HydroModPy is tailored to streamline the workflow for single-model groundwater simulations.

Despite these differences, the solver configuration in HydroModPy has been designed to be functionally compatible with MODFLOW 6 standards where possible, ensuring robust performance for unconfined, recharge-driven systems.

The reference model developed in ModelMuse (MODFLOW 6) and the HydroModPy model (MODFLOW-NWT) were evaluated and compared based on convergence, mass balance, head distribution, and boundary condition performance, using data from .cbc, .hds, and .lst files. The calibration results were further evaluated using the Normalized Root Mean Square Error (NRMSE), with detailed calculations

provided in Appendices B and C.

3.3.4 Key Differences Between the Models

Although both models were constructed using the MODFLOW platform, there are fundamental differences in how the modelling components were implemented. The reference model was developed using ModelMuse (MODFLOW 6), whereas the comparison model was constructed using HydroModPy (MODFLOW-NWT). This section summarises the main differences between the two approaches.

Table 3.7: Overview of key differences between the reference and HydroModPy models

Model Component	**Reference Model (ModelMuse)**	**HydroModPy Model**
MODFLOW version	MODFLOW 6	MODFLOW-NWT
Solver	BICGSTAB, Newton method	GMRES, Newton method
Interface	GUI-based (ModelMuse)	Script-based (Python)
Grid resolution	Locally refined (10–40 m)	Uniform/coarse (50 m)
Vertical discretization	17 layers	17 layers with custom bottom array
Geological input	Layer-by-layer manual assignment from shapefiles	Rasterized via "Geology" module
Fracture zones	Manual zones with high-K values	Assigned via hydrostratigraphic codes
Boundary conditions	Interactive assignment with visual QA	Automated via modules and shapefiles
Recharge assignment	Manual raster input, filtered with GUI logic	Uniform assignment with CHD masking in code
Drainage structure	Manually assigned via interface	Automated + scripted logic per drain type
Calibration approach	Manual adjustment in GUI	Python scripts with batch reruns

As seen in Table 3.7, the main contrasts lie in how spatial data were processed and assigned, how model geometry was built, and how reproducibility and flexibility were balanced. While the GUI-based model allowed fine-grained visual control and detailed local refinement, the HydroModPy model prioritised automation and code-driven transparency at the cost of some flexibility.

4

Results & Discussion

4.1 HydroModPy Model Implementation

The HydroModPy model was implemented following the framework's modular workflow, with adjustments made to accommodate Swedish data formats and site-specific modelling requirements. A structured project folder was created to manage inputs, simulation files, and outputs. The model domain was defined using the "Geographic" module, based on a high-resolution DEM from Lantmäteriet and a shapefile outlining the topographic watershed boundary.

All spatial datasets were projected to SWEREF 99 TM (EPSG:3010). The grid was initially defined with 10 × 10 m cell resolution near key infrastructure, but due to performance limitations, the resolution was upscaled to 50 × 50 m across the domain.

Geological units were rasterized from vector maps and assigned hydrostratigraphic codes using the "Geology" module. Fracture zones were integrated into the bedrock representation as distinct units. Surface water features such as lakes and streams were imported using the "Hydrographic" module and used to generate boundary condition layers. CHD boundaries were assigned to hydraulically connected lakes, and DRN boundaries were used to simulate general drainage, streams, and Lake Skiren (see Table 3.2).

Recharge was applied uniformly across the model domain using the "Climatic" module, at a rate of 200 mm/year (equivalent to 6.34×10^{-9} m/s). Cells already assigned CHD boundaries were masked to prevent recharge conflicts, in line with MODFLOW logic.

Vertically, the model consists of 17 layers. A custom bottom elevation array was constructed using DEM and borehole-based depth-to-bedrock data, see Equation 4.1. This array was imported to overwrite HydroModPy's standard bottom assignment functionality and ensure better representation of the stratigraphy.

$$\text{Model\_Top} = \text{Surface elevation (DEM)}$$

$$\text{L1\_Bottom} = \begin{cases} \text{Model\_Top} - 5, & \text{If Borehole data} < 5 \\ \text{Model\_Top} - \text{Borehole data}, & \text{otherwise} \end{cases}$$

$$\text{L11\_Bottom} = \begin{cases} \text{L1\_Bottom} - 5, & \text{If (L1\_Bottom} - 5) < 90 \\ 90, & \text{otherwise} \end{cases}$$

$$\text{L2\_Bottom} = \begin{cases} \text{L11\_Bottom} - 5, & \text{If (L11\_Bottom} - 5) < 25 \\ 25, & \text{otherwise} \end{cases} \tag{4.1}$$

$$\text{L3\_Bottom} = \begin{cases} \text{L2\_Bottom} - 5, & \text{If (L2\_Bottom} - 5) < -25 \\ -25, & \text{otherwise} \end{cases}$$

$$\text{L4\_Bottom} = -100$$

$$\text{L5\_Bottom} = -150$$

Hydraulic properties were assigned using the "Hydraulic" module. A depth-dependent decay function (Equation 4.2), was applied to represent the reduction in hydraulic conductivity with depth due to fracture closure:

$$K = \begin{cases} 1 \times 10^{-5} & \text{if layer} < 3 \\ 2.9 \times 10^{-9} & \text{if } Z < (\text{L1\_Bottom} - 45) \\ 2.5 \times 10^{-8} & \text{otherwise} \end{cases} \tag{4.2}$$

Custom Python scripts were used to link rasterized geological data with conductivity zones and to assign boundary condition parameters to the correct layers. These adjustments were necessary due to HydroModPy's limited support for assigning overlapping or prioritized zones in complex geological settings.

4.2 Simulation Performance and Results

Both the ModelMuse and HydroModPy models converged successfully, indicating that they were numerically consistent and stable. Simulated heads in the upper aquifer ranged from +90 to +117 meters in the ModelMuse model, and from +92 to +118 meters in the HydroModPy model. Both results reflect the general topographic gradient in the area showing consistency between the two models.

All flows have been converted to L/s based on simulation time step of 86400 seconds (1 day). A summary of the key flow components are presented in Table 4.1.

Table 4.1: Comparison of boundary flows and mass balance between the two models

Metric	**ModelMuse (L/s)**	**HydroModPy (L/s)**	**Rel. diff. (%)**	**Abs. diff. (L/s)**
CHD inflow	2.56×10^{-3}	2.00×10^{-3}	-21.9	5.6×10^{-4}
RCH inflow	4.20×10^{-1}	5.44×10^{-1}	$+29.5$	1.24×10^{-1}
DRN outflow	3.82×10^{-1}	5.32×10^{-1}	$+39.3$	1.50×10^{-1}
CHD outflow	4.06×10^{-2}	2.50×10^{-2}	-38.4	1.56×10^{-2}
Total inflow	4.22×10^{-1}	5.46×10^{-1}	$+29.4$	1.24×10^{-1}
Total outflow	4.22×10^{-1}	5.57×10^{-1}	$+32.0$	1.35×10^{-1}
Mass balance error	0.00 %	-1.94 %	-	-

The ModelMuse model shows a perfect mass balance, with a percent discrepancy of 0.00%. The HydroModPy model, however, has a mass balance error of 1.94%. This exceeds the commonly recommended threshold of 1%, which is generally considered acceptable depending on the model's complexity (Ahlfeld and Hoque, 2008). As discussed by Ahlfeld and Hoque (2008), such a discrepancy may be acceptable during early stages of model development or in particularly complex systems. However, efforts should be made to reduce this value where possible.

4.2.1 Calibration

Both models were calibrated using data from 22 observation points (groundwater boreholes located in both soil and rock). Model performance was assessed using the normalized root mean square error (NRMSE), a metric that evaluates the squared differences between observed and simulated groundwater levels, normalized by the mean of the observed values (Anderson et al., 2015).

The reference model achieved an NRMSE of 2.5%, indicating a generally low deviation between observed and simulated groundwater levels (Figure 4.1).

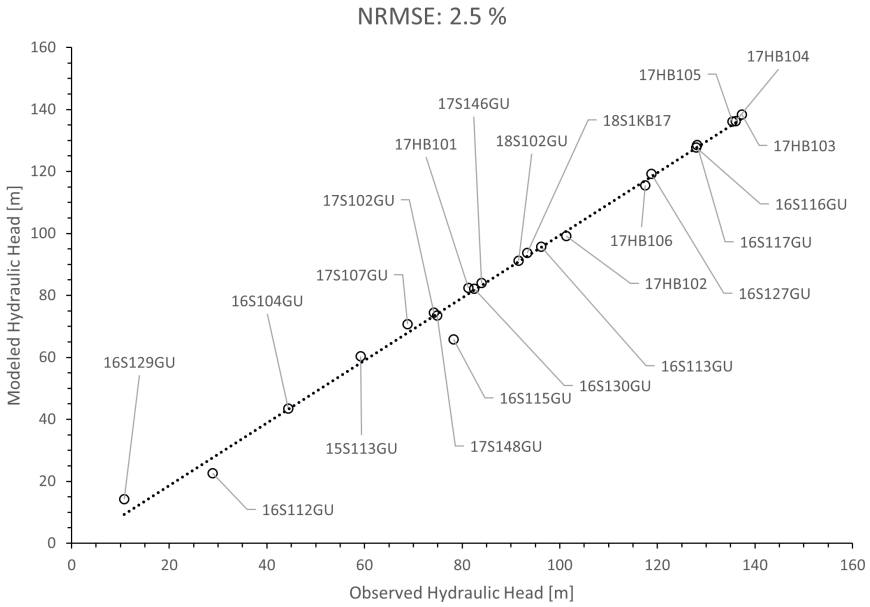


Figure 4.1: Calibration of the base scenario with NRMSE = 2.5% (Sweco, 2024).

In contrast, the HydroModPy model initially resulted in an NRMSE of 64.3%, indicating a poor fit between simulated and observed groundwater levels (Figure 4.2). Boreholes 16S112GU, 16S115GU, 17HB102, and 17HB103 showed the largest deviations from observed values.

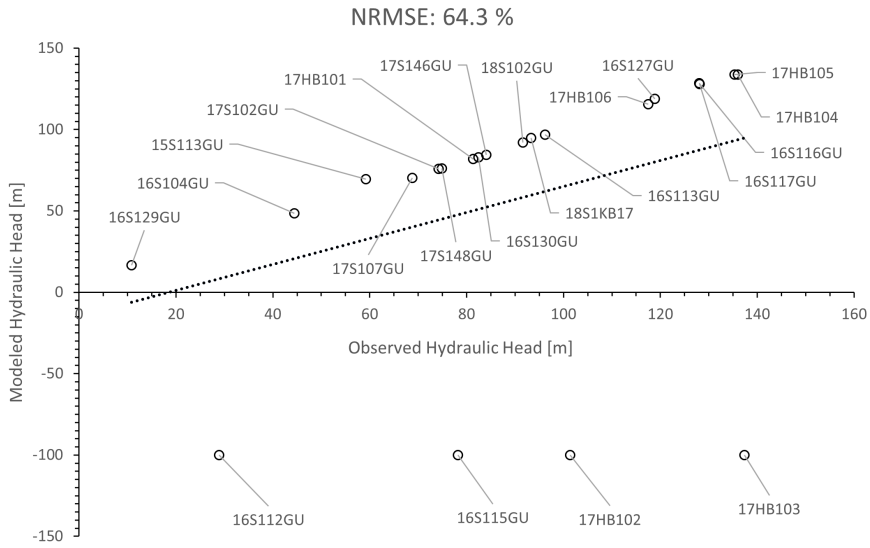


Figure 4.2: Calibration of the HydroModPy model with NRMSE = 64.3%.

To further investigate the nature of these deviations, a scatter plot was produced comparing observed heads to those simulated by both models at all 22 locations

(Figure 4.3). The reference model aligns closely with the trend line, while the HydroModPy model exhibits systematic underestimation at the four deviating points.

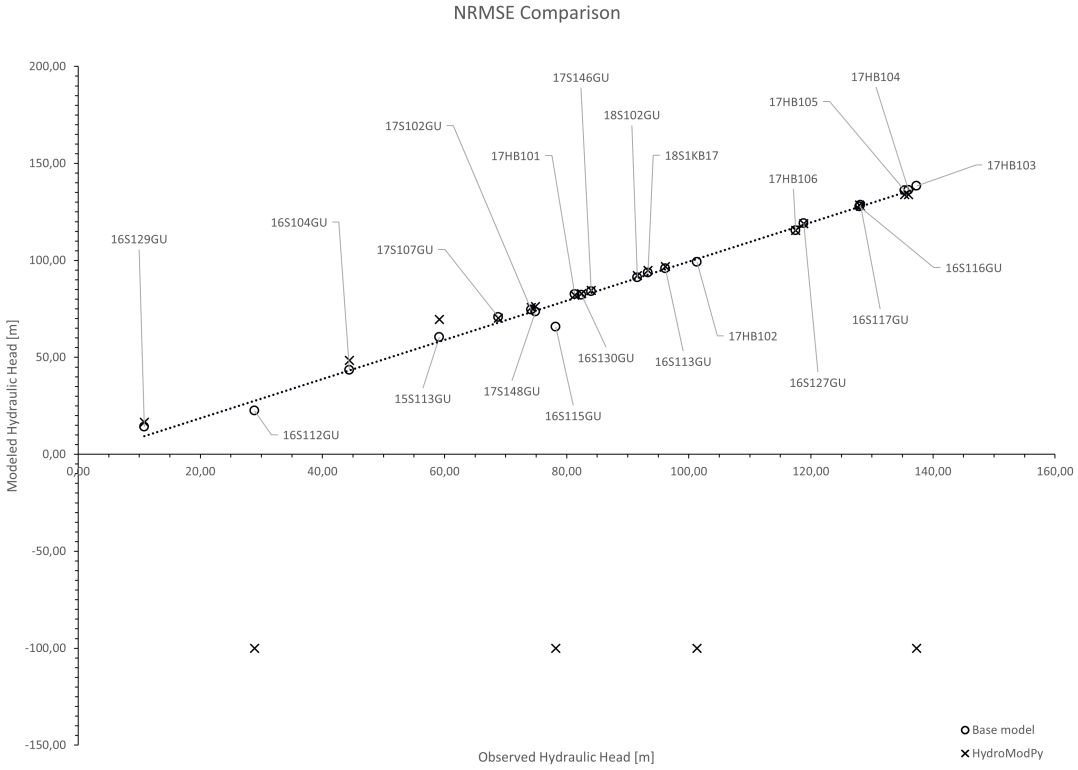


Figure 4.3: Comparison of simulated and observed hydraulic heads at 22 observation wells. Open circles represent the reference model (ModelMuse); crosses represent the HydroModPy model.

Boreholes 16S112GU and 16S115GU are located in soil, while 17HB102 and 17HB103 are located in rock. After analysing these points in relation to the surrounding geological conditions, no clear explanation for the deviations could be identified. One potential hypothesis is that the discrepancies are caused by unresolved local variations in the geological model. These may include thin high- or low-conductivity layers, bedrock fractures, or local variations in recharge or boundary conditions that are not captured at the current model resolution. In particular, the rock boreholes may be affected by discrete fracture zones, whereas the soil boreholes could reflect influences from fill material or perched groundwater.

The evaluation of the observation wells with large deviations reveals important conceptual differences between the HydroModPy model and the reference model. In the HydroModPy setup, a coarser horizontal resolution was used, resulting in a loss of detail compared to the locally refined grid in the reference model. Additionally, the assignment of hydrostratigraphic properties in HydroModPy was more generalized, whereas the reference model benefited from site-specific geological data with higher spatial precision.

Boundary conditions in HydroModPy were implemented through automated routines which, while efficient, may have introduced less precise boundary representations near sensitive wells. Furthermore, the HydroModPy model applied a global depth-decay function to simulate hydraulic conductivity in the bedrock. This approach potentially oversimplified fracture-dominated flow conditions, especially in rock wells where localized variations in fracture permeability are significant.

These differences in grid resolution, hydrogeological parametrization, and boundary condition accuracy help explain the higher simulation errors observed at specific wells in the HydroModPy model. The findings highlight the challenges of using automated model setups for complex geological environments where local heterogeneity plays a critical role.

4.3 Workflow & Usability Evaluation

Numerous challenges occurred during the implementation of the HydroModPy model. The primary technical difficulties were related to limitations in the framework when applied to a complex environment.

- Internal constraints in HydroModPy made it difficult to override defaults for vertical layering, raster linking, and shapefile reuse. For example, implementing custom vertical discretization logic initially conflicted with hard-coded assumptions in the framework, which had to be manually removed or rewritten to enable the desired setup.

- Visualization and preprocessing tools frequently failed due to hard-coded file paths and reliance on attribute naming conventions that did not match Swedish datasets. For example, geological shapefiles containing soil layering information from Swedish sources lacked required attribute fields such as #hex, C-Fond, N-Fond, and J-Fond. This mismatch led to repeated errors when using HydroModPy's built-in visualization tools, which depend on these specific attributes to render geological units correctly.

- Drainage elevations required extensive post-processing to correctly assign them to layers and avoid unrealistic configurations. Although the "Hydrographic" module automates some parts of the DRN setup, it does not ensure that drainage elevations align properly with underlying stratigraphy. Custom scripts were therefore developed to align drainage elevations with appropriate layers.

- Initial attempts to replicate the reference model's grid resolution of 10 meters $\times$ 10 meters in HydroModPy led to excessive runtimes and non-convergence. While longer runtimes may be acceptable in principle for detailed models, the fine-resolution setup could not be made to run successfully during the project's testing period. To enable simulations to complete within a reasonable time frame, the grid had to be upscaled to a coarser resolution of 50 meters $\times$ 50 meters. One key limitation encountered during this process was HydroModPy's lack of support for locally refined grids, which made it impossible to repro-

duce the fine spatial detail around the tunnel, fault zones, and lakes present in the expert-built model. This limitation likely contributed to observed discrepancies in mass balance and groundwater head calibration, particularly in areas where local geological heterogeneity plays a critical role. For example, the largest calibration errors in the HydroModPy model occurred at boreholes 16S112GU, 16S115GU, 17HB102 and 17HB103, located in soil and in fractured bedrock near the tunnel. These areas are represented with higher resolution in the reference model using local grid refinement, allowing for more accurate simulation of complex flow paths and vertical gradients that were not captured in the coarser HydroModPy grid.

- Some HydroModPy modules are hardcoded to work with French data formats and naming conventions. For example, the "Intermittency" class expects attributes that are not found in data provided by the Swedish Geological Survey (SGU), and the "Piezometry" class assumes French borehole naming conventions (e.g., BSS codes). To enable integration, SGU data had to be manually restructured - columns were renamed to match expected field names, additional fields were synthesized (e.g., borehole type and observation depth), and coordinate formats were converted to match HydroModPy's internal logic. This restructuring was done in a preprocessing script to allow import via the standard module interface.

In addition to technical issues, several organizational challenges affected the project:

- Time and resource estimation was overly optimistic. The expectation that one person could handle data preparation, geological interpretation, model construction, simulation, and comparison proved unrealistic. Although a functional model was produced, it did not reach the level of accuracy or detail that was originally hoped for. This highlights the need for more realistic planning and iterative development, particularly when working with new tools and unfamiliar datasets.

- Lack of continuous expert support made it difficult to validate assumptions early. While expert feedback was received, it came late in the project and was only partially implemented. As a result, several modelling choices (e.g., layer definitions, recharge distribution, and fracture representation) had to be made without guidance, leading to repeated trial-and-error iterations. Earlier expert input, particularly during the geological interpretation and parameterization phases, would likely have improved the initial model design and reduced unnecessary rework.

Although the goal of fully replicating the reference model in HydroModPy was not achieved, several key elements from the reference setup could not be implemented. These include the use of locally refined grid resolution near sensitive features, detailed stratigraphic layering based on borehole interpretation, and the full representation of site-specific boundary conditions such as head-dependent drains. These elements were excluded due to structural limitations in HydroModPy's framework

and the time required to adapt the tool to Swedish data formats and modelling conventions. Despite these constraints, the project provided valuable insights into the framework's capabilities and limitations in a Swedish hydrogeological context.

HydroModPy offers a functional framework for automated groundwater model setups. However, reproducing highly detailed and site-specific models requires significant modifications to its configuration and code. Based on the observed limitations, it can be concluded that HydroModPy, in its current form, is not yet well suited for building complex, high-resolution, and calibrated models where topography, layering, and boundary conditions need to be carefully designed and refined. However, the custom scripts developed as part of this project, such as those for bottom layer discretization, geological raster conversion, and and spatially distribute hydraulic properties, demonstrate that HydroModPy can be extended to support more advanced use cases. With further refinement and integration, these scripts could serve as the basis for a Swedish workflow within HydroModPy, especially for automated model development.

While HydroModPy builds upon standard MODFLOW packages, its codebase is streamlined and oriented toward workflow automation, consistent with how the framework is described in its documentation. Rather than offering a wide range of advanced modelling features, HydroModPy focuses on automating common preprocessing steps such as grid setup, geological attribution, and boundary condition assignment. Compared to more mature tools, it offers less flexibility for detailed customization, which can limit its applicability in high-resolution or site-specific modelling projects.

Implementing complex groundwater models using HydroModPy remains challenging, primarily due to the structural demands of such models rather than fundamental flaws in the framework itself. These challenges include the need for detailed vertical layering based on borehole data, fine horizontal resolution in sensitive areas, and the integration of spatially heterogeneous boundary conditions such as lakes and fault zones. Although HydroModPy offers useful automation for standard setups, its limited interface options, lack of graphical support, and sparse documentation constrain its practical usability in projects that require high degrees of customization or precision.

Furthermore, HydroModPy currently does not support MODFLOW 6, which includes many recent developments such as unstructured grids (DISV), improved lake and stream packages, and coupled groundwater transport simulations (USGS, 2024). If the framework is not updated to accommodate these features, it may become less compatible with emerging modelling practices. This limitation should be carefully considered when evaluating HydroModPy's suitability for future projects, particularly those involving complex boundary conditions, multi-package interactions, or high-resolution domains.

4.4 Recommendations for Future Work

While HydroModPy shows potential as a tool for automating model setup, it is currently best suited for prototyping or educational purposes. For building detailed groundwater models for complex infrastructure projects, where precise control over geometry, boundary conditions, and grid refinement is required, more mature frameworks such as FloPy with MODFLOW 6 or GUI-based environments like ModelMuse provide greater flexibility and reliability during the model development phase.

Given the author's limited prior experience in hydrogeological modelling, several decisions in this project had to be made independently, without established workflows or early expert guidance. The project would likely have benefited from earlier involvement of experts, particularly during geological interpretation and model setup. For future student-led modelling projects, it is recommended to ensure close collaboration with experienced professionals throughout the process, from initial assumptions to evaluation of simulation results. A great understanding of coding and scripting is also essential to properly evaluate HydroModPy's suitability for Swedish groundwater modelling projects.

Another suggestion for future work is to explore, depending on the evolving demand for automation in groundwater modelling, the potential for a PhD project focused on extending HydroModPy to automatically retrieve data from Swedish databases, such as SGU, SMHI, and Lantmäteriet - similar to how it currently interfaces with French data sources.

5

Conclusion

This study set out to explore (1) how does HydroModPy compare to a traditional GUI-based setup in groundwater modelling, and (2) what the framework's strengths and limitations are when applied to Swedish hydrogeological conditions.

In addressing the first question, the results show that HydroModPy can reproduce groundwater head levels comparable to those in an existing MODFLOW 6 model at most observation points. However, achieving this required modifications in the HydroModPy code to define a detailed layer structure, resolution, and boundary setup. This suggests that HydroModPy is currently less suited for large and high-resolution models where fine-scale spatial accuracy is required.

Regarding the second question, HydroModPy demonstrated several strengths, particularly in its integration with GIS, automation of repetitive tasks, and transparent, modular workflow structure. However, the framework also revealed important limitations in flexibility, error handling, and support for advanced features, especially when applied to Swedish modelling contexts. For example, arose when using geological shapefiles that lacked expected attributes (e.g., #hex, C-Fond), and when borehole identifiers did not match the French BSS code format assumed by the framework. These issues required extensive manual preprocessing and custom code modifications. In addition, HydroModPy's lack of compatibility with MODFLOW 6 and its reliance on Python scripting increase the technical threshold for effective use in professional modelling projects.

Therefore, HydroModPy should be viewed not as a replacement but as a complement to traditional modelling tools, particularly useful for research, education, and fast pre-processing of input data.

Bibliography

Ahlfeld, D. P., & Hoque, Y. (2008, September). *Impact of simulation model solver performance on ground water management problems* (tech. rep.). https://doi.org/10.1111/j.1745-6584.2008.00454.x

Anderson, M. P., Woessner, W. W., & Hunt, R. J. (2015). *Applied groundwater modeling : simulation of flow and advective transport*. Academic Press.

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*, 733–739. https://doi.org/10.1111/gwat.12413

Barlow, P. M., & Harbaugh, A. W. (2006). USGS directions in MODFLOW development. *Ground Water*, *44*, 771–774. https://doi.org/10.1111/j.1745-6584.2006.00260.x

Basharat, M., & Jawad, M. (2023, August). Building MODFLOW Model in ModelMuse GUI For Bari Doab. https://doi.org/10.21203/rs.3.rs-3217168/v1

Gauvain, A., Abhervé, R., Coche, A., Le Mesnil, M., Roques, C., Bouchez, C., Marçais, J., Leray, S., Marti, E., Figueroa, R., Bresciani, E., Vautier, C., Boivin, B., Sallou, J., Bourcier, J., Combemale, B., Brunner, P., Longuevergne, L., Aquilina, L., & de Dreuzy, J.-R. (2025, January). HydroModPy: A Python toolbox for deploying catchment-scale shallow groundwater models. https://doi.org/10.5194/egusphere-2024-3962

Höglund, N. B., Sparrenbom, C., Barthel, R., & Haraldsson, E. (2024). Groundwater modelling for decision-support in practice: Insights from Sweden. *Ambio*. https://doi.org/10.1007/s13280-024-02068-7

Hughes, J. D., Langevin, C. D., & Banta, E. R. (2024, December). *MODFLOW 6: USGS Modular Hydrologic Model* (tech. rep.). https://doi.org/10.3133/tm6A57

Hughes, J. D., Langevin, C. D., Paulinski, S. R., Larsen, J. D., & Brakenhoff, D. (2024). FloPy Workflows for Creating Structured and Unstructured MODFLOW Models. *Groundwater*, *62*, 124–139. https://doi.org/10.1111/gwat.13327

Länsstyrelsen Östergötland. (2024, June). Länsstyrelsen tillstyrker järnvägsplanen och lämnar artskyddsdispens för Ostlänken Stavsjö-Loddby.

Leaf, A. T., & Fienen, M. N. (2022). Modflow-setup: Robust automation of groundwater model construction. *Frontiers in Earth Science*, *10*. https://doi.org/10.3389/feart.2022.903965

SMHI. (n.d.). Ladda ner observationer från sjöar och vattendrag. https://www.smhi.se/data/hitta-data-for-en-plats/ladda-ner-observationer-fran-sjoar-och-vattendrag/waterdischarge15min

Sweco. (2024, April). *Beräkningar, Delområde Stavsjö-Böksjö, Kolmårdstunneln och Malmölandet-Loddby* (tech. rep.). Trafikverket.

Trafikverket. (2023a, May). *Ostlänken, järnvägsplan delen Stavsjö-Loddby* (tech. rep.). Trafikverket.

Trafikverket. (2023b, May). *PM miljökvalitetsnormer för vatten Ostlänken delen Stavsjö-Loddby Norrköpings kommun, Östergötlands län Bilaga till Miljökonsekvensbeskrivning* (tech. rep.). Trafikverket.

Trafikverket. (2024, April). *PM Yt- och grundvatten, delområde STavsjö-Böksjö, Kolmårdstunneln och Malmölandet-Loddby* (tech. rep.).

Tranviks Udde Fastigheter AB. (2023, October). *B-PM Bergteknik* (tech. rep.). The Stationery Office.

USGS. (2024). MODFLOW 6: USGS Modular Hydrologic Model. https://doi.org/10.5066/F76Q1VQV

Water Resources Mission Area (USGS). (2018, August). MODFLOW-2005 Hypothetical Aquifer System [Illustration]. https://www.usgs.gov/media/images/modflow-2005-hypothetical-aquifer-system

Winston, R. B. (2019). *ModelMuse Version 4: A graphical user interface for MODFLOW 6* (tech. rep.). https://doi.org/10.3133/sir20195036

Winston, R. B. (2022). ModelMuse Version 5.0: Software Release. *USGS Asset Identifier Service (AIS)*. https://doi.org/10.5066/P9JMSQ2M

A

Appendix 1 - Python Script

The model setup follows the HydroModPy example provided at https://hydromod.readthedocs.io. The code environment can be provided upon request by contacting Ezra Haaf (ezra.haaf@chalmers.se) or Lars Rosén (lars.rosen@chalmers.se).

AI was used during the development phase to assist with code debugging and problem solving. All scientific content, analyses, and conclusions were independently developed by the author.

B

Appendix 2 - NRMSE Calculation on ModelMuse Model

ID	X	Y	Z	Observed	Simulated	Residual	$(R_i-R_m)^2$	R_i^2	Obs^2	X_{sim} - μ_{sim}	X_{obs} - μ_{obs}	(Xsim - μsim)*(Xobs - μobs)	$(Xi - \mu_{sim})^2$	$(Xi - \mu_{obs})^2$
15S113GU	136393,90	6507671,74	72,34	59,17	60,33	-1,16	3,56	1,35	3501,09	-28,52	-30,41	867,18	813,39	924,53
16S104GU	133613,43	6506023,88	44,57	44,39	43,47	0,92	0,04	0,85	1970,47	-45,38	-45,19	2050,53	2059,31	2041,78
16S112GU	134808,28	6506943,77	96,39	28,88	22,54	6,34	31,56	40,24	834,05	-66,31	-60,70	4024,99	4397,53	3684,01
16S113GU	136439,61	6507824,98	78,75	96,18	95,75	0,43	0,09	0,19	9250,59	6,90	6,60	45,56	47,60	43,61
16S115GU	137733,60	6508220,19	128,44	78,23	65,80	12,43	136,95	154,47	6119,93	-23,05	-11,35	261,51	531,23	128,73
16S116GU	138118,94	6508411,53	128,93	127,99	127,67	0,32	0,16	0,11	16381,44	38,82	38,41	1491,06	1506,64	1475,63
16S117GU	136792,40	6507898,04	118,77	128,13	128,49	-0,36	1,18	0,13	16417,30	39,64	38,55	1528,34	1571,45	1486,41
16S127GU	133242,05	6505730,71	14,71	118,81	119,17	-0,36	1,18	0,13	14115,82	30,32	29,23	886,33	919,20	854,62
16S129GU	135265,33	6507276,23	82,63	10,80	14,20	-3,40	17,04	11,57	116,64	-74,65	-78,78	5880,46	5572,30	6205,67
16S130GU	140597,56	6509513,07	72,45	82,44	82,12	0,32	0,16	0,10	6796,35	-6,73	-7,14	48,05	45,34	50,92
17HB101	134192,76	6506303,06	75,91	81,31	82,45	-1,14	3,49	1,30	6610,50	-6,40	-8,27	52,96	41,00	68,41
17HB102	134336,27	6506610,66	92,27	101,36	99,15	2,20	2,19	4,86	10272,84	10,30	11,78	121,33	106,09	138,74
17HB103	141908,53	6510445,89	76,41	137,33	138,35	-1,02	3,04	1,04	18859,53	49,50	47,75	2363,75	2450,11	2280,44
17HB104	134310,71	6506415,15	85,91	136,02	136,25	-0,23	0,92	0,05	18501,17	47,40	46,44	2201,47	2246,92	2156,95
17HB105	133397,79	6505903,53	31,15	135,35	136,07	-0,72	2,09	0,52	18320,16	47,22	45,78	2161,62	2229,88	2095,44
17HB106	133994,76	6506499,51	2,90	117,52	115,45	2,07	1,80	4,28	13811,42	26,60	27,95	743,46	707,74	780,98
17S102GU	134360,29	6506779,64	20,30	74,20	74,48	-0,28	1,01	0,08	5505,64	-14,37	-15,38	221,00	206,58	236,42
17S107GU	137651,19	6508107,75	39,00	68,79	70,69	-1,90	6,88	3,59	4732,06	-18,16	-20,79	377,56	329,93	432,06
17S146GU	138365,90	6508429,82	47,90	84,02	83,95	0,07	0,43	0,01	7059,36	-4,90	-5,56	27,25	24,05	30,87
17S148GU	139221,58	6508724,81	67,50	74,88	73,49	1,39	0,44	1,93	5607,01	-15,36	-14,70	225,71	235,90	215,97
18S102GU	140097,85	6509418,52	1,40	91,58	91,12	0,46	0,07	0,21	8386,90	2,27	2,00	4,54	5,14	4,02
18S1KB17	134274,37	6506644,08	-14,90	93,30	93,73	-0,43	1,34	0,18	8704,89	4,88	3,72	18,17	23,81	13,87

Residual mean	0,726
Abs, Residual mean	0,726
SEE	0,68
RMS	3,214
NRMSE:	2,54%
Correlation coeff,	0,9959
Max Residual	12,43
Min Residual	-3,40
Cov	1163,765
μ_{sim}	88,85003
μ_{obs}	89,57605
σ_{sim}	34,4246
σ_{obs}	33,9452

C

Appendix 3 - NRMSE Calculation on HydroModPy Model 1.0

ID	X	Y	Z	Observed	Simulated	Residual	$(R_i-R_m)^2$	R_i^2	Obs^2	X_{sim} - μ_{Sim}	X_{obs} - μ_{obs}	(Xsim - μsim)*(Xobs - μobs)	$(Xi - \mu_{Sim})^2$	$(Xi - \mu_{obs})^2$
15S113GU	136393,90	6507671,74	72,34	59,17	69,57	-10,40	1869,29	108,16	3501,09	12,83	-30,41	-390,08	164,58	924,55
16S104GU	133613,43	6506023,88	44,57	44,39	48,44	-4,05	1360,68	16,42	1970,47	-8,30	-45,19	375,00	68,87	2041,81
16S112GU	134808,28	6506943,77	96,39	28,88	-100,00	128,88	9224,62	16610,05	834,05	-156,74	-60,70	9513,62	24567,81	3684,05
16S113GU	136439,61	6507824,98	78,75	96,18	96,81	-0,63	1119,67	0,39	9250,59	40,07	6,60	264,58	1605,21	43,61
16S115GU	137733,60	6508220,19	128,44	78,23	-100,00	178,23	21139,67	31765,93	6119,93	-156,74	-11,35	1778,44	24567,81	128,74
16S116GU	137733,60	6508220,19	128,44	127,99	128,57	-0,58	1116,51	0,34	16381,44	71,83	38,41	2759,17	5159,23	1475,61
16S117GU	138118,94	6508411,53	128,93	128,13	127,87	0,26	1061,17	0,07	16417,30	71,13	38,55	2742,29	5059,38	1486,38
16S127GU	136792,40	6507898,04	118,77	118,81	118,93	-0,12	1086,23	0,02	14115,82	62,19	29,23	1818,09	3867,80	854,61
16S129GU	133242,05	6505730,71	14,71	10,80	16,56	-5,76	1489,21	33,12	116,64	-40,19	-78,78	3165,71	1614,92	6205,72
16S130GU	135265,33	6507276,23	82,63	82,44	82,83	-0,39	1103,89	0,15	6796,35	26,09	-7,14	-186,18	680,61	50,93
17HB101	140597,56	6509513,07	72,45	81,31	82,03	-0,72	1125,64	0,51	6611,32	25,28	-8,27	-209,01	639,29	68,33
17HB102	134192,76	6506303,06	75,91	101,36	-100,00	201,36	28400,63	40545,85	10273,85	-156,74	11,78	-1846,98	24567,81	138,85
17HB103	134336,27	6506610,66	92,27	137,33	-100,00	237,33	41818,15	56325,53	18859,53	-156,74	47,75	-7484,96	24567,81	2280,41
17HB104	141908,53	6510445,89	76,41	136,02	133,85	2,17	940,55	4,69	18501,44	77,11	46,44	3581,36	5946,26	2157,01
17HB105	134310,71	6506415,15	85,91	135,35	133,83	1,52	980,40	2,32	18319,62	77,08	45,77	3528,46	5942,09	2095,23
17HB106	133397,79	6505903,53	31,15	117,52	115,48	2,04	948,30	4,16	13810,95	58,74	27,94	1641,36	3450,17	780,85
17S102GU	133994,76	6506499,51	2,90	74,20	75,77	-1,57	1183,68	2,46	5505,64	19,03	-15,38	-292,59	362,07	236,43
17S107GU	134360,29	6506779,64	20,30	68,79	70,20	-1,41	1172,41	1,98	4732,06	13,45	-20,79	-279,66	181,01	432,07
17S146GU	137651,19	6508107,75	39,00	84,02	84,52	-0,50	1111,43	0,25	7059,36	27,78	-5,56	-154,37	771,83	30,87
17S148GU	138365,90	6508429,82	47,90	74,88	76,13	-1,25	1161,96	1,57	5607,01	19,39	-14,70	-284,98	376,02	215,98
18S102GU	139221,58	6508724,81	67,50	91,58	92,08	-0,50	1111,14	0,25	8386,90	35,34	2,00	70,80	1248,73	4,01
18S1KB17	140097,85	6509418,52	1,40	93,30	94,85	-1,55	1182,19	2,40	8704,89	38,11	3,72	141,90	1452,11	13,87

Residual mean	32,835	
Abs, Residual mean	32,835	
SEE	16,23	
RMS	81,304	
NRMSE:	64,26%	
Correlation coeff,	0,3438	
Max Residual	237,33	
Min Residual	-10,40	
Cov	920,5439	
μ_{sim}	56,74123	
μ_{obs}	89,57636	
σ_{sim}	78,87315	
σ_{obs}	33,94508	

DEPARTMENT OF ARCHITECTURE AND CIVIL ENGINEERING
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY