Master Thesis Report

Development of a PYTHON module for streamflow routing and coupling with ground water model MODFLOW

by Oscar Cardenas

Academic Tutor: Dr. Frank Molkenthin,
Brandenburg University of Technology Cottbus-Senftenberg
Institutional Tutor: Erik Nixdorf
Host institutio: Bundesanstalt für Geowissenschaften und Rohstoffe

15 August 2023

Declaration

I hereby declare that this work was written independently without any unauthorized assistance or sources not given credit within the work. All words, phrases, passages, and data taken from other sources have been properly cited. No parts of this work in the same form or a similar form have ever been previously handed in to fulfill an examination.

Oscar David Cardenas Garnica

BTU, 15/08/2023
Location, Date — Author's signature

Acknowledgement

I would like to say thanks to my advisors Erik Nixdorf (BGR) and Prof. Frank Molkenthin (BTU) for all the guidance and help in the elaboration of this project. Also, to my family and friends for all the support through the last two years.

Contents

Table index

Figure index

Abstract

Nowadays the concentration of the population in the cities and growing industrial activities have increased the stress and pollution of water resources. As a result, models that integrate several hydrological processes in one model are needed to achieve better water management. To answer that need, the main objective of this project was coupling the groundwater flow software MODFLOW through the development of a stream routing module using Python and GIS.

For the coupling, a loosely coupled was selected allowing the interaction of MODFLOW a 3D model, with the developed Python stream routing module that models rivers and channels in 1D. The results of the developed module were compared with the results of the SFR, which is a MODFLOW package to couple the stream routing model with the groundwater model through a fully coupled approach. Another goal of this project was to develop a module that gives more freedom to the user and that can be modified easily to be coupled with other groundwater software or by changing the calculation approach of the stream routing.

Two synthetic networks within a groundwater domain grid were created to couple and test the module. The smallest network was implemented to validate if the physical behavior of the model was correct, and the biggest network to perform a sensibility analysis of the Python module.

Keywords: Loosely couple, MODFLOW, SFR, ground water, surface water, fully couple, stream routing.

1 Introduction

Worldwide the extraction of water resources, pollution of rivers and aquifers, and climate change are increasing hydric stress. In Germany, more than 60% of the demand from households, the public sector, and small businesses, and 15% of the industrial demand is supplied using groundwater (Houben &Broda, n.d). But this percentage can also vary depending on the city, for instance, Berlin extracts 24.1% of the water for public supplies, while Brandenburg extracts 87.6% of the water from the ground (Houben &Broda, n.d).

In cases of federal states like Brandenburg, that rely heavily on the GW resources it is more important to protect the quality of the aquifers. Two common threads to the aquifer, the first thread comes from agriculture due to nitrogen content in fertilizers that is diluted with irrigation or rainfall water, and then infiltrates into the soil. The second thread is related to mining activities that can contaminate the aquifers with heavy metals from the extracted minerals. The main cause of this process is the weathering process of the rocks at atmospheric conditions that produce sulfuric acids and the mobilization of metal and metalloids (Gammons et al., 2010).

Currently, Germany is in the process of closing all the coal mines across the country by 2038 (BGR, 2O22). This has led to increasing research on the effects of mining activities on the environment. Currently, BGR is assessing the impact of the mining activities that take place in Brandenburg. This includes groundwater (GW) resources, as they can be polluted due to the infiltration of mining wastes and then spread to the rivers through the water exchange between GW and SW. For instance, iron-oxide deposits have been found in the Spree River and other rivers and might produce acidification of the water of the rivers and the aquifers (BGR, 2O22).

From a hydrological perspective, the pollution of a river or an aquifer is not an isolated event, and the pollutant can spread through a region as ground and rivers are connected. Depending on the soil and weather conditions, surface water (SW) can infiltrate to the soil, or flow from the soil to the river. To achieve a better comprehension of pollution phenomena, first, a model capable of representing the interaction between the GW and SW is needed.

Previously, several authors have developed tools to connect couple models. For instance, in 1996 the USGS created a coupled model using the river routing software BRANCH’ with the GW software MODFLOW. In order to perform the coupling and find the leakage flow exchange between both models, both models were run iteratively until the aquifer head and river stage converge (Swain & Wexler, 1996). The process from USGS is called loosely coupling and will be explained in further detail later in this document. Since 1996, MODFLOW has been constantly developed and has developed its GW and SW coupling model MODFLOW-SFR, which integrates a fully coupled approach.

Even though there are several tools and previous work in the coupling of GW and SW, most of the tools are not open source or require several steps for input preprocessing and creation of the model. Because of this, the goal of this project was to create and validate an open-source surface water routing code that can be coupled to MODFLOW or to other GW software with small changes through Python. Python was selected as it is a programming language that is broadly used in the scientific field and has several open-source libraries. For instance, there are GW libraries like OpenGeoSys and MODFLOW with the Flopy extension. It is important to mention that from the two GW alternatives mentioned, the code was created with the Flopy extension by using the loosely coupling approach.

2 Theoretical background

2.1.1 Surface water modelling conceptualization

The first step for the conceptualization of the routing module was defining the topology of the module. The selected approach was developing a 1D discretization of the rivers. This approach was selected as developing a grid is more complex and it would require more computational resources for the calculation of the model. In addition, having a 2D model requires more information for the input files which might impact the flexibility and usage of the module.

The routing model is composed of a river network that is divided by rivers in the GIS before the preprocessing stage. Those rivers are subdivided into reaches after the preprocessing, and they inherit the topological and hydraulic characteristics from the river in the original GIS attribute table. Each reach is composed of an initial and final node where the information of the stage is stored. In figure 1 it can be observed a schematical example of the discretization. Furthermore, there is a dependent relationship between the GW grid discretization and the SW discretization, as a reach will be created by splitting the river for every cell that is within a cell of the GW spatial discretization.

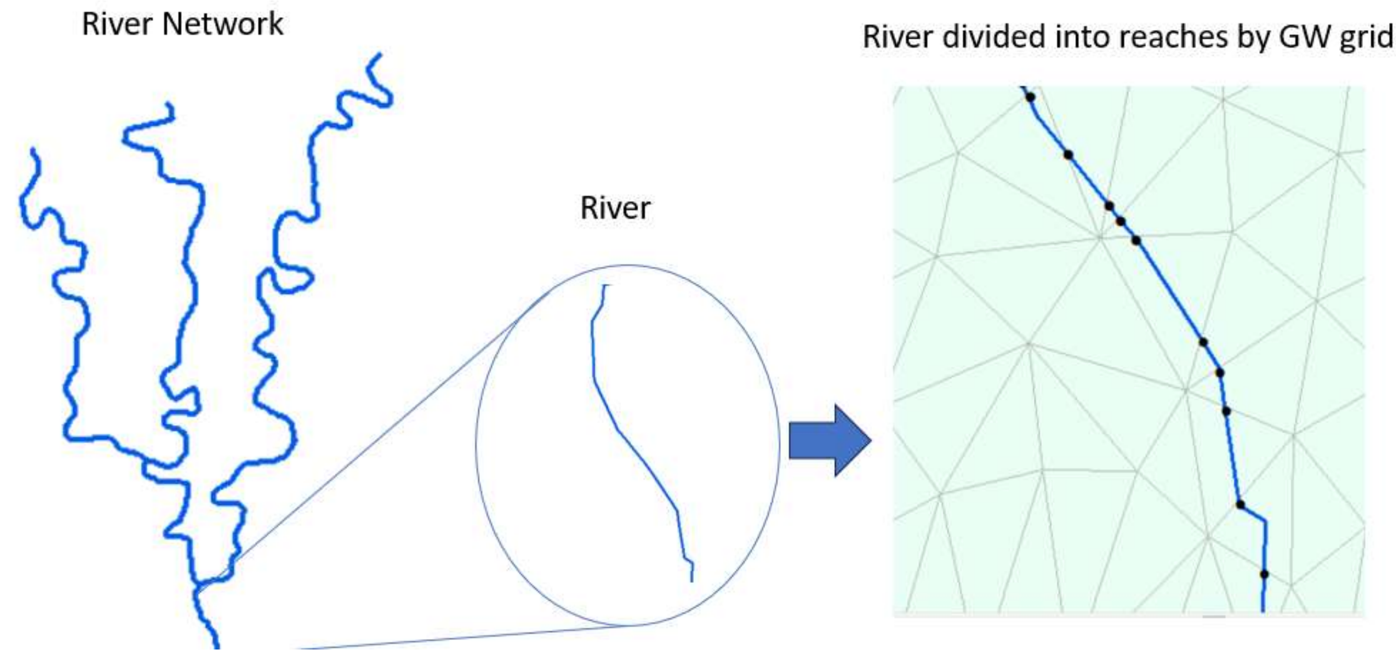

Figure 1. Schematic discretization of the river network.

The river network assignation to the GW spatial discretization might be a source of uncertainty as in the division of the river into reaches, a small reach can be assigned to a cell with a big area, attenuating the real impact of the reach in the GW model if the cell is too big. Nevertheless, the uncertainty can be attenuated through the refinement of the unstructured grid close to rivers to reduce the uncertainty. In those cases, is up to the user to develop a proper grid suitable for the river model.

In addition, the river cross section geometry is assumed as a square cross section and there is only exchange of water between the river and ground in the riverbed. As a result, the manning equation is simplified to equation 1, where H is the water depth, Q is the flow, n is the manning number, W is the width of the reach and S is the slope. In other equations there is a coefficient indicating the unit conversion, but for the proposed module it is assumed that the units are in the international system of units. Furthermore, to calculate the stage the river bottom elevation is added to the water depth.

$$H = \left(\frac{Q * n}{W * S^{\frac{1}{2}}}\right)^{\frac{3}{5}} (1)$$

The conductance estimation comes from Darcy's law where the water is infiltrating through the area of the riverbed and flowing across the riverbed. Moreover, as the water exchange will only take place at the river bottom, for the calculation of the conductance, only the topological variables river length and river width are necessary to determine the area where water is flowing into the soil as can be observed in equation 2. One source of uncertainty in equation 2, is the assignation of one homogeneous hydraulic conductivity for one reach in cases where the reach length is too long, or the riverbed soil is heterogenous across the reach. Even though it might be a source of uncertainty, if the groundwater cell size is small enough the error can be reduced, and several conductances can be assigned. In figure 2 we can observe a scheme of the required values and in equation 2 the equation for the calculation of the conductance, where the meaning of L, K, W, and b can be found in figure 2 and C is the conductance.

$$C = \frac{L * K * W}{B} \quad (2)$$

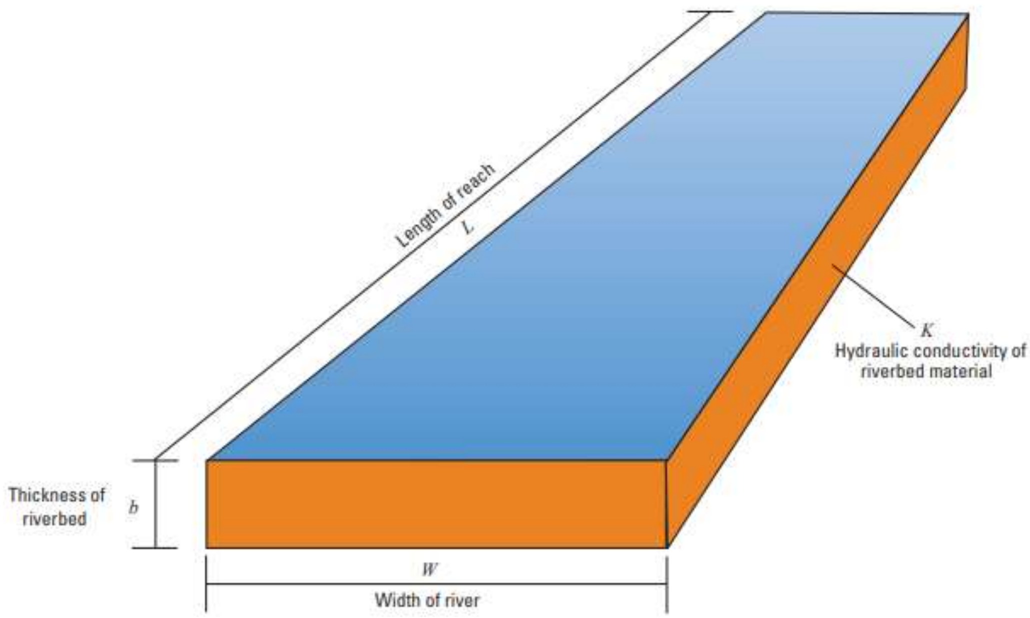

Figure 2. Schematic representation of the river water exchange zone with the aquifer. Source: (Langevin et al., 2017)

Once the equations for the calculation of the river water depth and the leakage flow are established, it is now time to consider the spatial position of the river and the connection between the river and the soil. In the case of MODFLOW, the calculation of the head and all the source terms are located at the centroid of the cells as will be explained in deeper detail later. As a result, when the Python module calculates the river stage and leakage across the reach, the leakage will be relocated into the ground at the centroid of the cell. In cases where there is more than one reach, the leakage will of all the rivers will be accumulated and then relocated to the centroid of the cell.

When the data comes from MODFLOW the value of the head of the centroid of the cell will be assumed as constant across the cell, allowing the calculation of only one homogeneous leakage value for the cell. Therefore, the user must use a proper cell size close to the river, because in cases where the cell is too big the spatial distribution of the cell will not be representative of the aquifer. In cases where the spatial distribution of the leakage or the river morphology is important is advised to use a 2D approach.

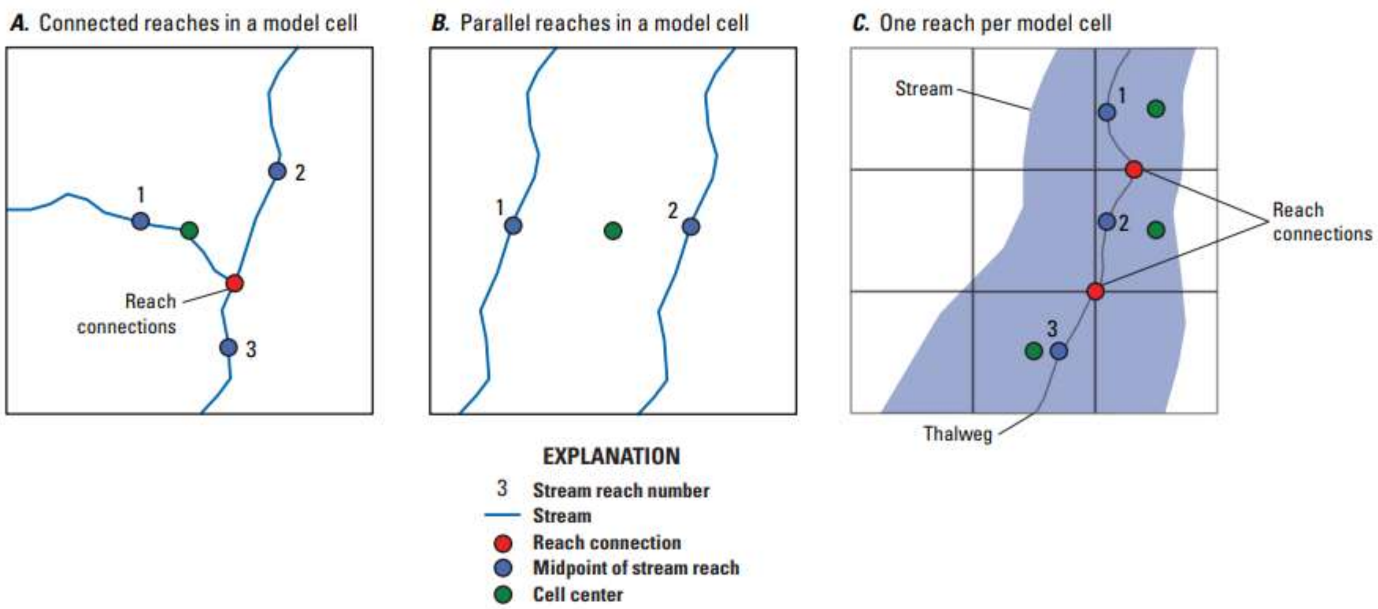

Figure 3. Discretization scheme of the groundwater cell and the reach. Source (Langevin et al., 2017)

2.1.2 Temporal conceptualization

In this section will be explained the temporal discretization of the module and MODFLOW. Due to the temporal discretization differences depending on the software that is being used, the developed time discretization of the module is heavily dependent on MODFLOW time discretization. Consequently, MODFLOW time discretization approach might reduce the flexibility of using other GW numerical models.

The processes in the GW domain and the SW domain will always have different time scales. The GW processes tend to take months to observe changes in the hydraulic head while the rivers and channels will change rapidly in the presence of precipitation. Consequently, the time scale of the available data for both domains might not be the same, creating different time intervals for both models. Nevertheless, the information from the SW with a higher time step can be implemented to improve the calculated leakage.

In MODLFOW TDIS package the time discretization is composed of three parameters. The first parameter is the stress period length (PERLEN), which gives the length of the time span that is being considered for a set of constant parameters. The second parameter is the number of time steps (NSTP), which will divide the stress period length into time steps, improving the calculation stability and accuracy. The third parameter is the time step multiplier that allows the calculation of non-constant time steps (TSMULT). No information about the initial and ending time is provided in MODFLWO through dates, instead the time units must be provided to the package.

In equations 3 and 4 we can observe equations presented in the MODFLOW 6 manual, describing how to calculate the time steps based on the multiplier value. These equations were implemented within the time manager class of the module to be able to use several flows with different time steps in the future. Furthermore, by knowing the time intervals of the module it is possible to access the head in each time step to improve the visualization of the groundwater leakage in time.

$$\begin{cases} \Delta t_1 = \text{PERLEN} * \left(\dfrac{\text{TSMULT} - 1}{\text{TSMULT}^{\text{NSTP}} - 1}\right) & \text{TSMULT} \neq 1 \\ \Delta t_1 = \dfrac{\text{PERLEN}}{\text{NSTP}} & \text{TSMULT} = 1 \end{cases} \quad (3)$$

$$\Delta t = \Delta t_{old} * \text{TSMULT} \ (4)$$

MODFLOW also has defined different time units for the input of the information that can be observed in table 1. In contrast, the valid units for the Python module are only seconds as the adopted units for the development of the module are the international system of units. Because of this, for the internal management of the time manager of the developed module, all the units are converted to seconds, by extracting the time units from the MODFLOW model. After calculating the leakage, the result is converted from m^3/s to the defined MODFLOW time unit as a sink or source term. Furthermore, in MODFLOW months were not implemented as it is variable depending on the number of days and for the years it is assumed that the year has 365 days all the time. In case the conversion factor of the year is a problem, it is possible to change this value in the time manager class contained in the Routin_Classes.py file.

Unit	Conversion factor
seconds	1
minutes	60
hours	3600
days	86400
years	31536000

Table 1. Module time units and conversion factor of the unit to seconds.

The current version of the module works like the SFR package where there is only one input at the inlets of the rivers per stress period. The current implementation is capable of iterating through all the stress periods, and stress period times with time the multiplier of the MODFLOW model. However, due to non-solved bugs in the current implementation, only the leakage is being solved for the first stress period, and the inlets of the river network are only given for one stress period. This decision was also taken as the current version needs more testing before adding more complexity to the calculation.

As the time discretization of the routing package requires more development, a proposal on how to use different inlets in the river network was created. In the current proposal, there is no time step, as the time information will come from a CSV with a time column, the reach id, and the flow at the inlets of the network must be provided. From this CSV a dictionary with the time as the key will be stored and as the value a second dictionary with a reach id as a key and the flow as the value. Then the leakage will be calculated using the heads from the MODFLOW time step while the SW time is less or equal to the MODFLOW time step. This approach assumes that there are no extreme changes in the GW head during the time intervals. These changes also require making modifications in the result storage approach, that have not been considered yet. The workflow diagram can be observed in figure 4,

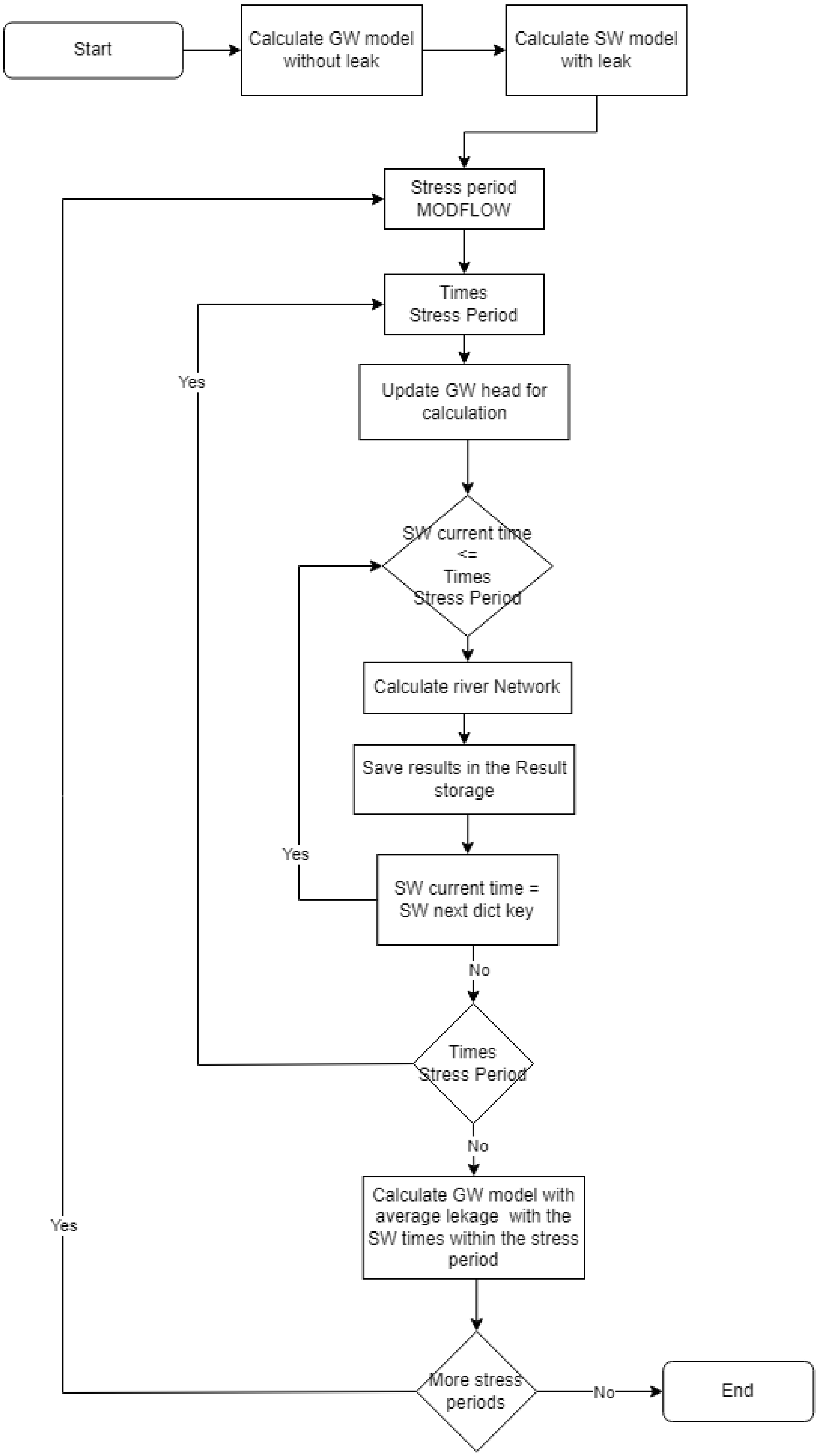

Figure 4. Proposed time workflow for further development.

2.1.3 Ground water and surface water exchange

The interaction between groundwater and a river can take place in three basic interactions, the river loose water to the groundwater, the river gains water from the groundwater or there is a mix of both interactions in different sections of the river (Winter et al., 1998). In figure 5 can be observed two of the basic interactions and how this process is regulated by the water table level, where a water table level above the river level allows water inflow from the ground, and a water table level below the stage generates a leakage from the river to the ground.

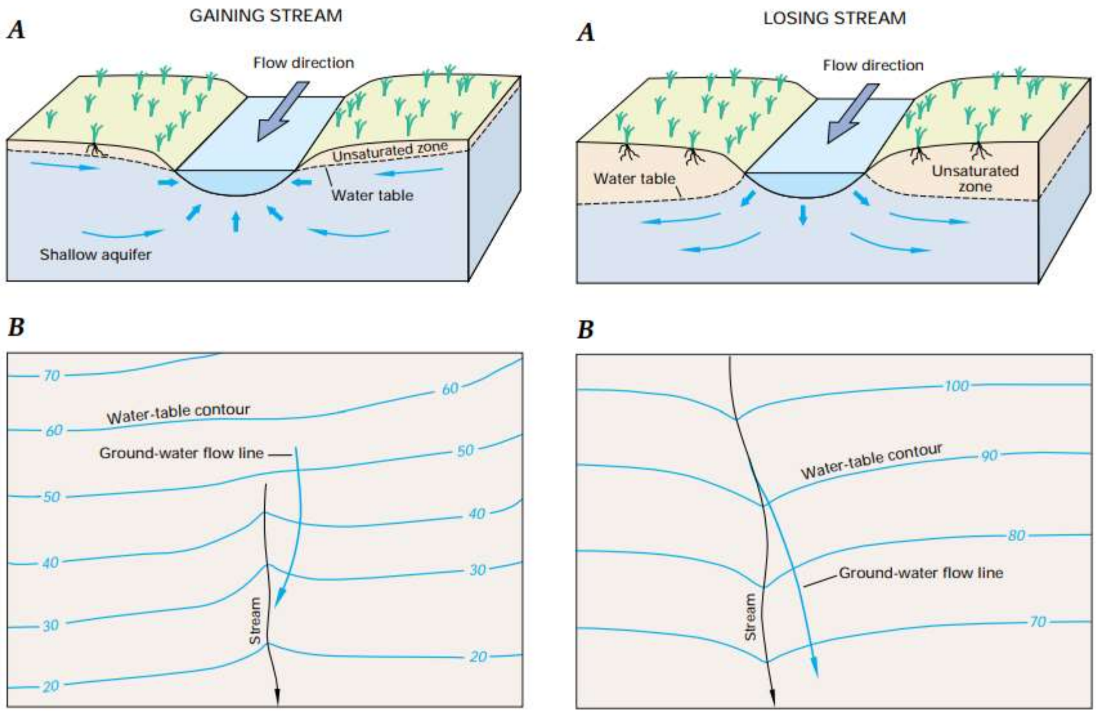

Figure 5. Ground water and surface water interaction illustration. Source: (Winter, et.al, 1998)

Riverbanks can also be important for the interaction between groundwater and surface water. As is shown in figure 6, when the river stage reaches the bank level, there is going to be a leakage from the river to the ground, but at the same time, the groundwater will go from the ground to the flooding area. This behavior can be relevant for flooding analysis as this might be an initial condition where part of the flooding plain is already flooded. Nevertheless, for this Python module, the main goal is to model normal conditions in the basin rather than flooding events.

Another relevant question that arises through the interaction of the floodplain and the groundwater is whether the flooding could be represented through a 2D or 1D approach. In case the 1D approach is selected, then it is not possible to model the interaction between the floodplain and the groundwater. Even though it is possible to add the riverbanks through the cross-section in software's like HEC-RAS, a 1D approach cannot consider the spatial distribution of the riverbank, add an independent saturation coefficient to the soil of the riverbank, and model the lateral leakage from the river to the soil. As the developed river routing module is a 1D model, the utilization of the module during flood events might lead to wrong results and a misrepresentation of reality.

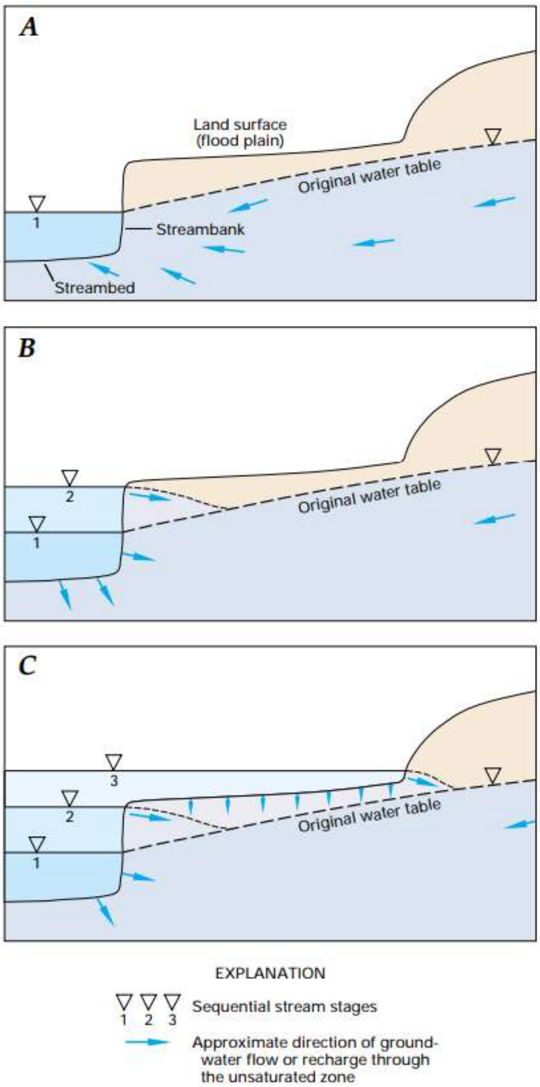

Figure 6. Illustration of flood area water recharge. Source: (Winter, et.al, 1998)

Once the physical interaction between groundwater and surface water is comprehended, it is necessary to translate that interaction into equations that describe the phenomena. For this purpose, the head-dependent approach described in the MODFLOW user manual described by Langevin et al. in 2017 was selected. Furthermore, using the same set of equations from MODFLOW SFR package will make it possible to compare the obtained results from models created using the Python module with the SFR package.

By combining equation 2 and Darcy's law it is possible to calculate the leakage that when there is GW head is higher than the riverbed bottom elevation. In this equation C is the conductance of the riverbed discussed previously, h_{stage} is the river stage, rbot is the riverbed bottom elevation as can be seen in figure 7, h_{gw} is the water table head, and Q_{leak} is the vertical flow exchange between the soil and the river. The calculated leak will be positive if the water is flowing from the soul to river and negative if the water is flowing from the river to the soil.

$$Q_{leak} = C * (h_{stage} - h_{gw}) \ if \ h_{GW} > rbot \ (5)$$

The second case can be modeled once again by using equation 2 and Darcy's law. In this case, the GW head is below the riverbed bottom, the reason why the leakage is only dependent of the river stage and the water can only flow in one direction. For this equation rev_elev is the riverbed top elevation and the other variables are the same as equation 5.

$$Q_{leak} = C * (h_{stage} - rbd_{elev}) \ if \ h_{GW} < rbot \ (6)$$

Figure 7. Schematic representation of the different components of the reaches

From equations 5 and 6, the greatest source of uncertainty is the conductance of the model. If the magnitude of the conductance is too high, there might be cases where the leakage is higher than the flow in the river. As can be seen in figure 8 the leakage will be constant unless the GW head exceeds the river bottom. Furthermore, this conceptual approach relies on the assumption that there are no considerable changes in the flow during the stress period (Langevin et al., 2017). In addition, the river cannot go dry or exceed the river flood banks for long periods of time and in case those events take place there should not affect the river-aquifer interaction (Langevin et al., 2017).

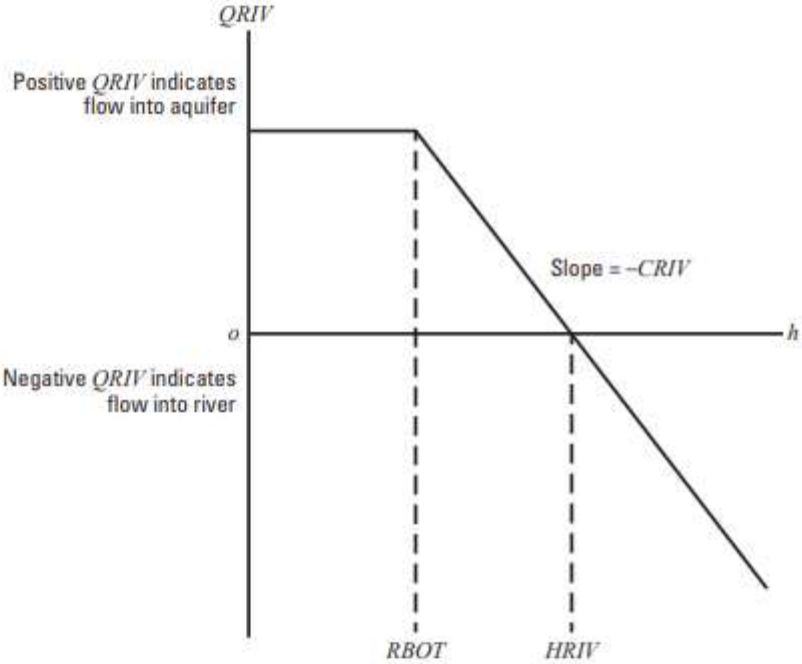

Figure 8. River leakage function plot. (Langevin et al., 2017)

3 GW-SW coupling approaches

Different coupling approaches for SW and GW models have been performed in other studies. In this section, some of those approaches will be reviewed. As it can be seen in figure 9 from the research of Haque there are two main divisions for coupling, detailed domain models and domain integrated models. The difference between detailed domain models and integrated models is that domain-integrated models are a simplification of the hydrological cycle to forecast water quality and quantity (Haque, et.al, 2021). In contrast, the detailed domain models are tools that were developed to model the interaction between GW and SW models (Haque, et.al, 2021).

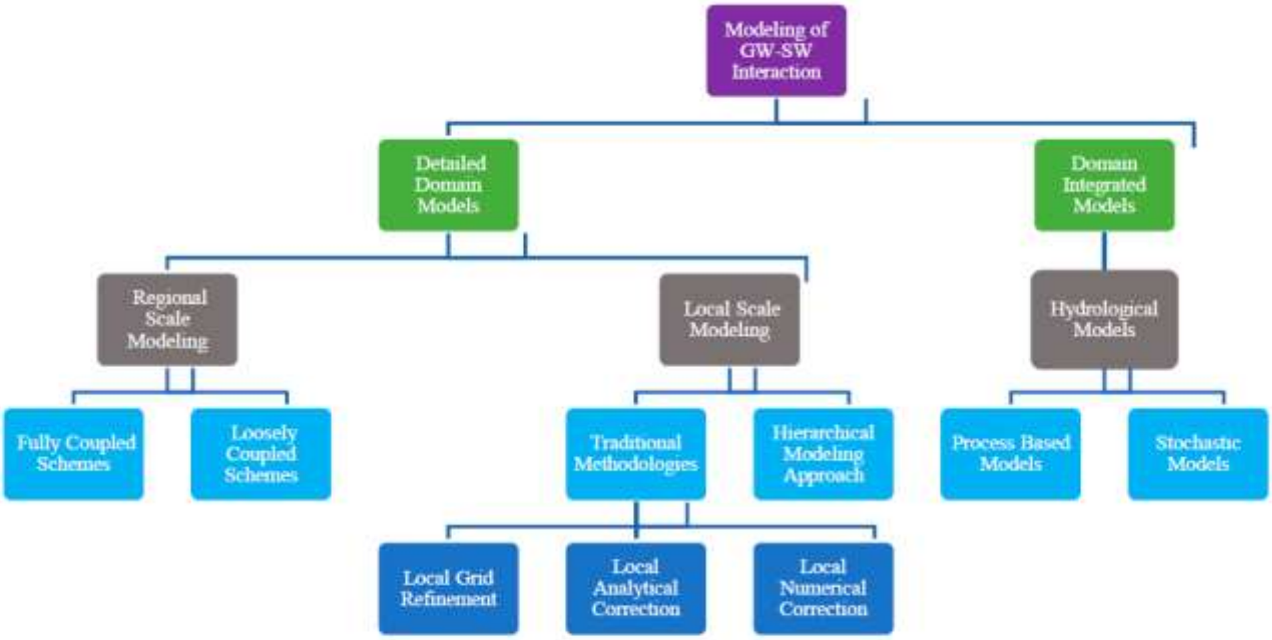

Figure 9. Graphic with different coupling methodologies for GW and SW. Source: (Haque, et.al, 2021)

Due to the creation of a module specifically for the interaction of GW and SW, there will be no in-depth explanation of the domain-integrated models. The focus of this section will be the explanation of large-scale modeling and regional-scale modeling and the reasons why regional modeling was selected for the development of the module.

3.1.1 Local Scale Modeling

While there is no specified definition of area for local or regional scale and in the literature, there is not a consistent definition of the different scales mentioned in the hydrology literature (Gleeson & Paszkowski, 2014). The selected definition for a local model was a small spatial extension of land, usually a town, a river, or a city. This small extension of land allows the introduction of more detailed characteristics of the study zone, like detailed land use, local precipitation information, and having a more refined grid in important zones like rivers. Consequently, a local model will allow us to have more accurate and representative results of the area that is being modeled. Nevertheless, the grade of detail of the model might require more input information to achieve a good model.

3.1.2 Regional Scale Modeling

In the case of regional scale modeling the goal of the model is to comprehend the hydrological and hydraulic processes on a broader scale. Consequently, the results obtained from a regional scale will not have the same accuracy from a local scale model (Langevin et al., 2017). As the scale increases the information available input data availability for the development of the model will decrease, especially in zones where there is no urbanization or any kind of data gathering.

Furthermore, having a detailed model like a local scale model with the same input variables will lead to higher execution times. As a consequence, the regional scale modeling requires a simplification of the different components of the region. For instance, at a large scale, the cell size will increase, and the selected land use will be an average of the different types of land use in the cell. For the SW model, the cross-section of the rivers might only allow one cross-section per river, or a uniform cross-section like in MODFLOW where the cross-section of the river is only considered rectangular. Is important to remember that this simplification will always lead to uncertainty in the results and that an exact result might not be possible.

The main reason why the regional scale approach was selected is the current context where BGR is performing an analysis of the mining zones of Brandenburg. Currently, GW regional models and tools are already developed, the reason why the selection of a regional scale approach is more appropriate for the module. Furthermore, the assumption of a rectangular cross-section will not be valid on the local scale where more detailed information of the morphology of the river is required. Therefore, the selection of a regional scale approach was the most suitable option from a goal perspective and from a limited-time implementation perspective.

3.2 Fully Coupling vs Loosely Coupling

The task of coupling groundwater and surface water models has been performed by several researchers with different software. In literature two main types of coupling technics can be found, fully coupled, and loosely coupled models, both having advantages and disadvantages (Haque, 2021). For instance, the fully coupled models combine the set of equations and solved simultaneously the modeled phenomena at the in time same time references (Tu, 2018). The main limitation when the fully coupled model is implemented is the lack of field data as the groundwater and surface water are modeled at the same time (Haque, 2021). Furthermore, the fully coupled model is more computationally expensive to solve due to a more complex set of equations, and the calibration is more complicated due to a higher number of parameters interacting with each other (Haque, 2021).

The loosely coupled models consist of two different topological discretization's where the models communicate through a constant exchange of information back and forth (Tu, 2018). Even though this coupling method can be convenient and provides flexibility to the user, the result will be inaccurate as it relies in the convergence of two different models, meaning that a certain tolerance must be fixed by the modeler (Tu, 2018).

In the case of this project, the selected coupling method was loosely coupling. One of the reasons for this decision is that making a fully coupled model requires developing the GW and river routing numerical model to solve both at the same time. This approach was not possible due to time constraints and because it was beyond the scope and the goal of the project of developing a streamflow routing model.

Once the method to couple the models was selected, the next stage was defining how both models will communicate between them. For the calculations made in the river routing module, the GW head will be obtained from MODFLOW, during this process the river stage and the leakage flow will be calculated. Then the leakage will be input into MODFLOW by assigning a well in the cell center when there is a presence of a reach.

In MODFLOW manual, it is mentioned that a positive flow indicates that water is being injected into the soil, and a negative flow indicates extraction of water. This sign convention can change depending on the type of package, like the SFR package, where the injection of water to the ground is negative because the river is losing water. The sign convention for the results of the module will be from the perspective of the river like in the SFR package.

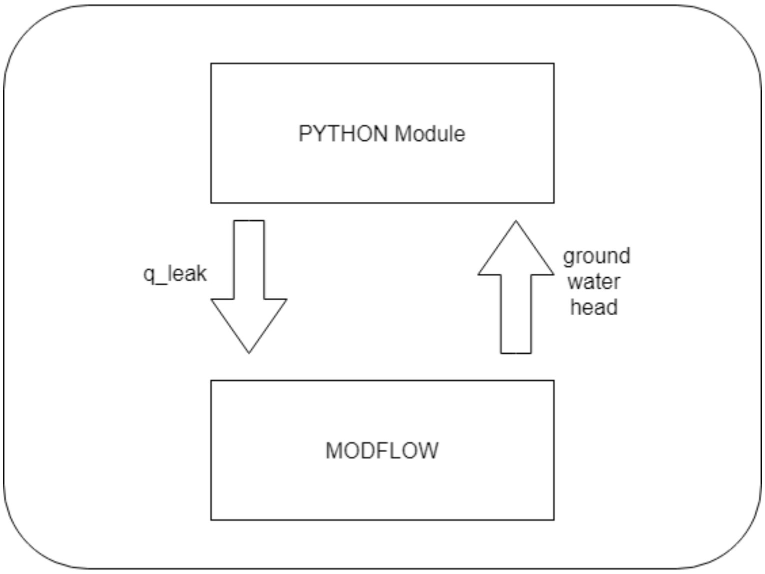

Figure 10. MODFLOW Python module information exchange diagram

4 Methods

In this section the most relevant tools that were necessary for the implementation of the module are described and explained. Furthermore, in this section can also be found the information about the implementation of the theoretical background described below. Finally, a step-by-step tutorial on how to utilize the developed is presented in this section.

4.1 MODFLOW and FloPy

MODFLOW is a software developed by the USGS since 1981. It has had several versions going from one of the oldest being MODFLOW-88 to the most recent version MODFLOW 6, released in 2017. Currently, MODFLOW 6 is still under development as new updates are being released on a regular basis. In addition to the development of MODFLOW, USGS has developed other tools to improve the user experience like ModelMuse a user interface to interact with different versions of MODFLOW, or FloPy a Python package to create and run models.

Due to the constant improvements and the useful packages to model different physical phenomena, MODFLOW was selected to make the coupling with the surface water module. Once the GW tool was selected it was necessary to comprehend MODFLOW functionality from physical equations and user functionality perspectives. The functionality will be boarded from a physical, temporal, and functional perspective as temporal functionality was explained before.

From a physical perspective, MODFLOW uses the control volume finite difference (CVFD) numerical method to solve Darcy's Law in three dimensions (Langevin, et.al, 2017). Because of this MODFLOW can calculate lateral and vertical flow through the different layers of a model. Furthermore, in MODFLOW the control volume is represented by a cell and the set of connected cells constitutes the grid of the model (Langevin, et.al, 2017). In addition, the calculation of the head in the cell is performed in the node of the cell that is in the center of the cell (Langevin, et.al, 2017).

In addition to the calculation of the GW model, MODFLOW offers other packages like the WEL package and the SFR package. As was explained before the connection of the coupling is performed by assuming an inflow in the center of the cell. In the case of the package, it can be used to add to or withdraw water from the soil at a constant rate specified by the user for a given stress period (Langevin et al., 2017). One of the main limitations of the implementation of this well in MODFLOW is that it is not possible to define a limit for the extraction of the GW. As a result of the lack of extracting limit, when the conductance of the riverbed has a high order of magnitude, more water than available might be extracted from the soil. To achieve an appropriate behavior, it is necessary to use small conductivity values.

From the user functionality perspective, MODFLOW FloPy extension is a library that reads inputs from ASCII or binary and writes the outputs as binary files (Bakker et al., 2016). Furthermore, FloPy has reduced the number of inputs for the packages, as it has assigned default values to almost all the input parameters of the different MODFLOW packages (Bakker et al., 2016). Whether an input ASCII file or binary file is used depends on the type of variables that are being stored. For instance, the time file will be stored in an ASCII file due to the small number of lines, while the unstructured grid will be stored in a binary file due to the size of the information.

In the case of the GW flow model (GWF), two outputs are created, a head file with the water table assigned based on the ID of the cells, and a cell budget file where the flows going through the cell is registered (Bakker et al., 2016). To read the files, it is not necessary to create a decoder as FloPy has implemented the reading of binary within its code. In addition, Flopy divides the information to access into two data structures, one for the information head and another one for the budget in the cell.

As the information on the groundwater head is stored in one independent class there is no need for a record name to access the information on the GW head, only the time interval is required. However, the information from the water cell budget is broader, the reason why is needed to specify the record name and the time interval of the information the user wants to access. The time interval is the accumulative reason why it is necessary to sum the previous stress period lengths to the time interval in the desired stress period the user wants to access.

In Appendix 1 a table with the record names of the GWF model binary file can be found. Other packages like the SFR package have an independent output structure with a different record name table that can also be found in appendix 1. This information is presented as the SFR package was used to contrast the results obtained with the developed module. In addition, while creating the MODLFOW model, the user must specify the name of the output file for every package and the amount of information that must be written in the output file. In the text below is shown an example o how to access the output information in FloPy.

```
Head_budget_information = sim.get_model().output.head().get_data(totim = Time_interval)
Cell_budget_information = sim.get_model().output.budget().get_data(text = Record_name, totim = Time_interval)
```

In addition to the functionality and data visualization is necessary to comprehend the functioning of FloPy and the possible errors that might take place. The major error found during the development of the module was that in cases where the assigned flows of the wells produce a numerical error in FloPy, MODFLOW will not converge and the MODFLOW model will be lost as the writing of the files will not be completed. Because of this, a backup of the model is created when the code is run and is highly advised to keep a copy of the module in a different folder than the working space.

4.2 SFR MODFLOW

SFR (Streamflow Routing) is a package developed by MODFLOW, to model rivers fully coupled with a MODFLOW GWF module. The SFR package solves the continuity equation in a steady state by assuming there are no changes in the inflow and outflow rate in discrete time intervals, in a uniform location that does not change with time (Langevin, et.al, 2017). However, it is still possible to add different conditions by assigning different inflows and outflows for each stress period to observe the evolution of the system through time.

Moreover, the MODFLOW-SFR package is a fully coupled model, the reason why the program will calculate a mathematical solution no matter if the parameters do not have a physical meaning. This behavior was tested with the synthetic network described later with hydraulic conductivities of 100 m/s that cannot be seen in natural soils. In contrast, the developed Python module will not converge making the model more sensible to instabilities. Nevertheless, as the MODFLOW-SFR package will always arrive at a solution, the reason why the user must be critical of the obtained results.

Another advantage of the SFR package is that the implementation of the source code is open source. This means that it is possible to check the implementation of the code to observe the calculation of the stream routing in cases that are not properly or not described at all in the code. One example of this is the smooth function implemented for cases when the stage of the river is drying, and the stage level is small. The flexibility and the availability of reviewing the implementation of the code is one of the reasons why this tool was selected to contrast and guide the developed module.

4.3 SFRmaker

SFRmaker is one of the most important steps to allow the interaction between GIS information and a brief introduction to this tool is necessary. This Python library was developed by the USGS to automatize the process of creating MODFLOW models where there is a presence of rivers. For this library, it is important to consider that there are no pip installation lines for the library, due to the need for specific versions of other libraries like NumPy and GeoPandas. Because of this is highly suggested to use a platform where it is possible to build Python environments.

Even though there are instructions about the installation of this library and the creation of the environment on the official website, some issues to be considered are presented here. The first issue was related to current problems with the Spyder debugger, as breakpoints are not working properly in the most recent versions. At the same time if the version of Spyder that is implemented is too old some compatibility issues might arise, and it will not be possible to install SFRmaker.

Another issue that was faced while installing sfrmaker was related to Mambaforge. When this path for installation is selected the user must replace the installation channel of Conda with Mambaforge as both are not compatible. The problem arises after changing the channels, due to after this change it was not possible to install packages anymore in Spyder, as the mamba command was not recognized. This experience is recorded in the report not to complain or criticize about any package. Instead, it is intended to help other users in case they face similar problems, or if they want to decide which version of Spyder to use.

The following path to succeed in the installation of sfrmaker was using the yml file provided in sfrmaker documentation. Furthermore, instead of using the suggested mamba command, the standard conda channel for installation was used. The downside of this process is that it takes more time to install all the packages. The upside of this process is that fewer modifications in Spyder are required.

Some errors and limitations can be found in the functionality of the library. For instance, the river network GIS polyline file must be created from upstream to downstream. This is because the library reads the polyline in the same order as the points were added. Furthermore, it is important to split the reaches when there is a tributary or a bifurcation. When this action is not performed, the library might create duplicated reach cell ID assignation and inner loops in the bifurcations. In figure 11 we can observe the correct discretization of the river, in which the river is sb-divided into two sub-rivers due to the presence of a tributary.

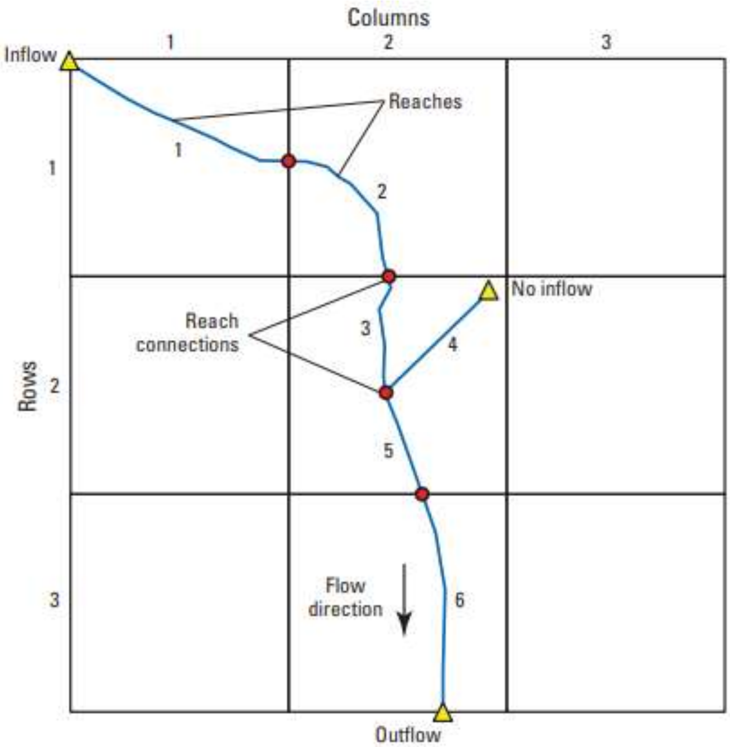

Figure 11. Schematic representation of the reach division when there is a reach connected to a tributary. Source: (Langevin et al., 2017)

4.4 Preprocessing

The first step for the development of the module was the creation of the preprocessing script. In this stage the user will give as an input a shapefile with the geometry of the rivers, a shapefile of the grid, or in case the user has a vtu file the code has a function that can make the conversion of vtu to shapefile. Then by using the library sfrmaker, the river reaches are assigned to the cells which result in four outputs.

Two of the outputs are intended for visual validation, the first one is the processed river shapefile where the rivers have been split into reaches as explained before. The second is a grid file containing only the cells of the grid where there is a reach. These two files are created for the user to perform the validation of the assignation process as there is no user interface implemented. The third file is a CSV containing the attribute table of the reach shapefile. This process described can be seen in figure 12, where there is a flow diagram of the preprocessing steps.

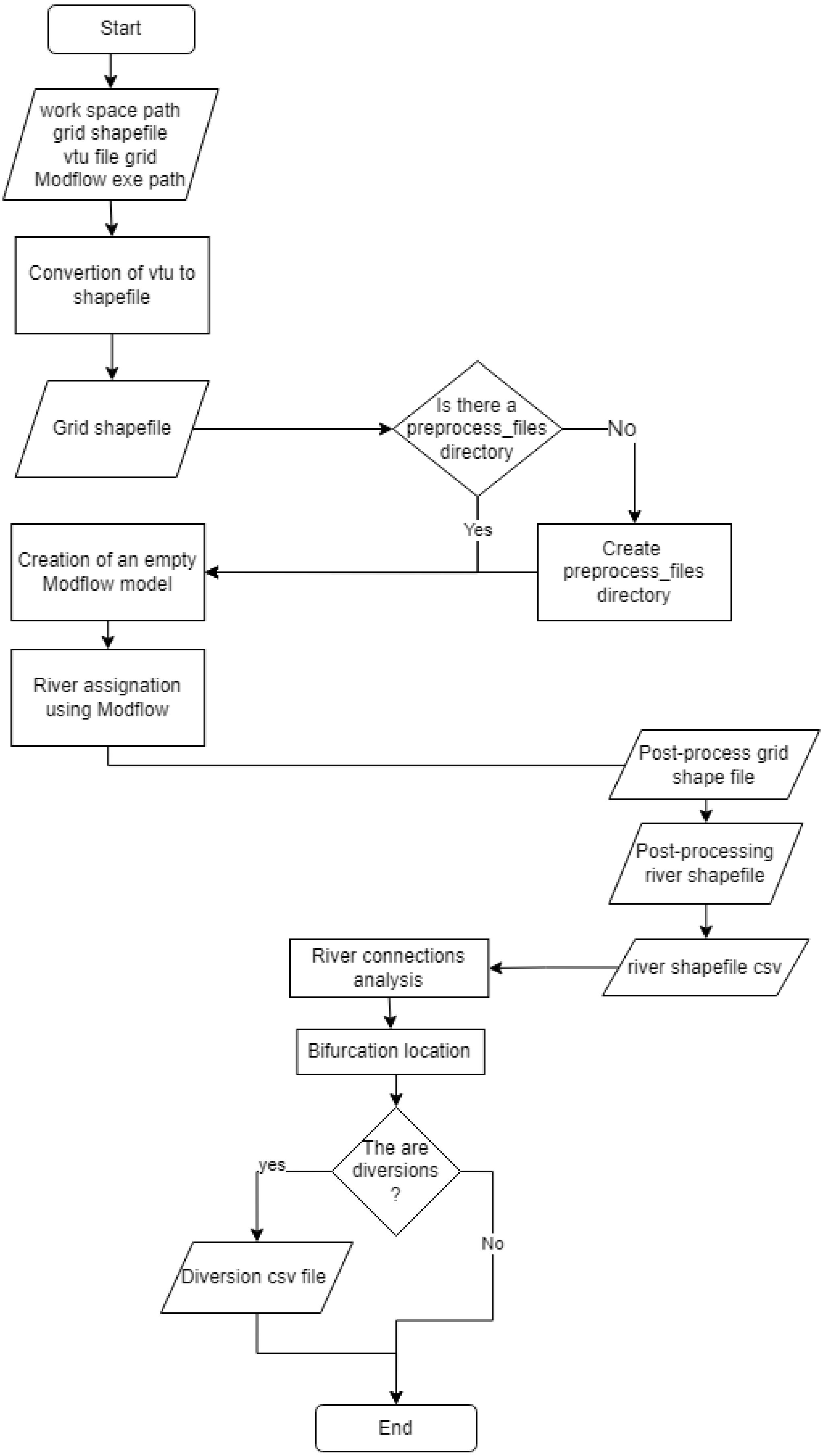

Figure 12. Preprocessing flow diagram

The last file that is created is the diversion CSV that must be filled out by the user. However, no diversion file will be created if there is diversion detected. This file is created using the connection function, which analyses the distance between start points and end points of the polyline with a tolerance of 10-6 meters to detect if there is a connection. The connection function identifies which reach is diverted and it will write one line in the csv file for every diversion that is detected.

Is the job of the user to fill the reach_type column that can be "**diversion**" or "**main**" and to specify the flow that is diverted into the diversion reach. It is important to mention that only one main reach can be defined, and the flow that goes through the main reach diversion downstream must be 0. For the remaining diverted reaches the user must specify which flow is going to pass through the diversions. The format of the CSV is presented in table 2.

diverted_reach	out_reach	reach_type	q
…	…	main/diversion	…

Table 2. Bifurcation CSV input scheme

4.5 Data structures

As can be seen in figure 14, four data structures were implemented for the elaboration of the module. The first structure is the River Network where there is a composition relationship with the reach section. In the river network, the reach sections are stored within a dictionary, where the key is the reach id assigned using SFRmaker and the value is the reach section class associated with that reach section. Another attribute of the river network is the solved path, where the order of calculation from the first iteration is stored to optimize the code, making the other runs of the network more efficient. In the river network the method to add the boundary conditions into the network was implemented. In this case, the boundary conditions must be a flow in m^3/s, as the unit conversion only takes place when the flow is input into the WEL package.

The second class that was implemented for the module is the Reach Section class. In this class, the information required to do the calculation of the stage for each reach is stored. One of the variables that is worthy of explaining is the variable rbd_elev, which is the riverbed elevation measured from the bottom of the GW model first layer, up to the riverbed top. This variable should not be confused with the rbot variable which is the rbd_elev minus the riverbed thickness. This is illustrated in figure 7 to make it more visual for the user.

Moreover, the method to add diversions is in the Reach Section class and it is called within the River Network constructor method. Because of this, the diversion information must be input when the River Network is created. The bifurcation information must be input as a Pandas data frame that is created using the CSV file that was explained before. Another important aspect related to the reach section is the cells that are over a head boundary condition. In case the reach is over a head boundary condition, no leakage will be calculated, and the Boolean attribute is_in_boundary will be changed from false to true. The user must not confuse is_in_boundary with the attribute is_boundary, as this attribute will be true if the node is identified as the inlet reach of the network.

The third class that is created is the Time Manager class. The parameters of this class were explained in the time discretization chapter, reason why further explanation will not be made. The fourth class that is created is the Result Manager, which stores the information at a stress period or time step level depending on the information of TDIS file.

The results are stored in a dictionary with three key levels. The first key is the stress period id starting the counting from 0, then the measurement key, and then the time step of the measurement, and then a list of the value per cell as can be observed in figure 13. The hriv is the river water depth at the last node of the reach, the initial water depth is not stored as it can be calculated using the qin values, which helps to save memory. Moreover, there are two methods to export the results, one as CSV files for post-processing and the other format as VTU files for the visualization and post-processing calculations using ParaView pipelines.

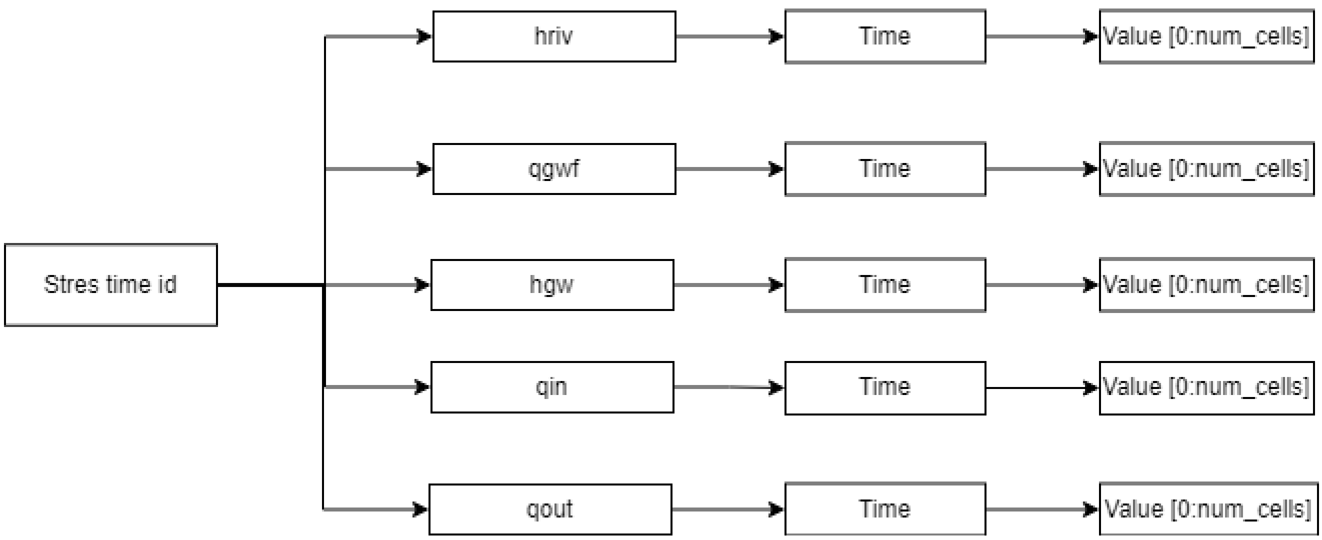

Figure 13. Result Storage key structure

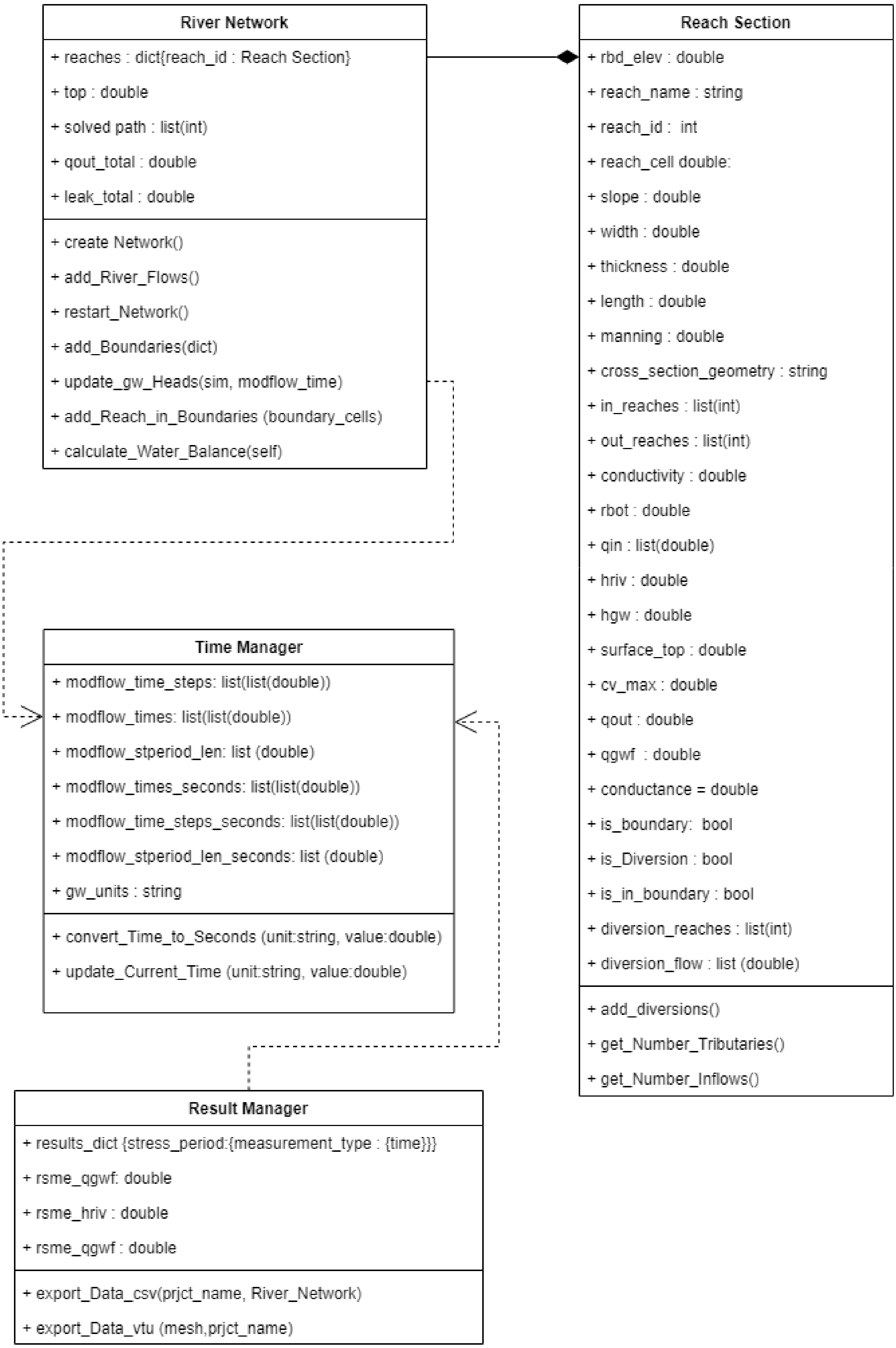

Figure 14. Data structure diagram

4.6 Processing

The first step to start the processing of the information is loading the MODFLOW model using FloPy. To open the model, it is necessary to input the directory path where all the MODFLOW files are stored, and the executable file from MODLFOW 6. The result of this process is a simulation class where it is possible to access the model and the different packages objects. The most relevant information of the packages is the time discretization and the head boundaries within the model.

It is important to know the head boundary cells of MODFLOW because there is no effect from the well injection or extraction when there is a head boundary in MODFLOW. Consequently, to preserve the water balance and avoid losing water in cells where there is no leakage it is important to determine if the reach is in a head boundary. In addition, the calculated leakage in these cells where there is no exchange, might generate perturbations in the RMSE as this value will not change when compared with the previous iteration, leading to an early convergence of the model if there are several cells within a head boundary cell.

Besides the hydraulic information from MODFLOW, it is also important to define the required temporal information from the model. As mentioned before, there are two levels of time in MODFLOW in which it is necessary to iterate in the time domain. The first level is the stress period, and the second level are the time intervals of the stress period which might not be uniform. This information will be used to fill the properties of the time manager object described above.

The time discretization from MODFLOW can be extracted when the model is loaded using FloPy. This information is called by accessing the time discretization package (TDIS). As this information comes from binary files it is necessary to know the headers of the different blocks. The information on the time units can be found in the “options” block of the TDIS file, and the information about the stress and time step is in the block "perioddata". In addition, of the block is necessary to know the data set name of the parameter within the block, in the case of the units the data set is called "time_units" and for the period data, the data set name is "perioddata" once again.

```
sim.get_package("tdis").blocks[HEADER].datasets[Data_Set_Name]
```

Once the information from MODFLOW model is extracted, it is time to load the information from the preprocessed files created before. For this it is necessary to input the paths of the files into the processing python as it was done before with the MODFLOW model. By using the river_connection.py file and the created CSV from the preprocessing stage it is possible to create the SW network to be solved. In addition to this a VTK or VTU mesh from the GW and SW is necessary for the storage and creation of the graphs for the outputs of the process.

After creating the GW and SW models, both models are calculated without any leakage. Then the water heads are updated in the Network Class and then the network is calculated to find the leakage. The leakage and the calculated head are stored within the result manager class. Then the calculated leakage units are verified and in case the GW model units are different than seconds a conversion is performed. After this check, the leakage is introduced into GW model through the MODFLOW WEL package, the simulation is written, and the model is run. The results from the head are stored in the result manager and updated into the river network class depending on the current time. As result a full iteration of the hydraulic model consists of one run of the SW model and one run of the GW, and there is a lag in the information of the GW model.

4.7 Data visualization and postprocessing

For the visualization of the results the selected software was ParaView. This is an open-source software that was developed to analyze information coming from vector data, or mesh data. This is the final stage of the module and the reason why it is necessary to have a version of the mesh in VTU format. In addition to being open source, ParaView file creation and manipulation can be performed with PyVista, a Python library used to create 2D and 3D visualization files. This library allows the automation of the file output creation for the required output manipulation for the post-processing stage.

The process to assign the information to the mesh starts with the creation of a NumPy array or a list with a size equal to the number of cells and a copy of the VTU mesh file that can be read using PyVista. Each cell has an ID number that corresponds to a value with the same index in the NumPy array or list. This means that the mesh must start counting the index from 0 instead of 1 to make a successful assignation of the mesh.

In addition to the 3D dimensional representation of the results, it is also possible to add a time dimension to the VTU files. This can be done by creating one PVD file which is an XML file with the time step and the VTU file name associated to that time. Then within ParaView, it is possible to create an animated video or a plot with several views to observe the evolution in time of the plotted parameter. The downside of this approach is that a VTU file per time must be created increasing the required storage size of the results plots.

In addition to the VTK files, the module also exports the data into CSV files, one file per data type key in the Result Storage dictionary. The file has a structure where there is a column with the cell ID, the reach ID in case this value exists, and the value of the measure per time. Examples of the CSV can be found in chapter 6.1.6, where there are tables with the CSV format of the Python module results.

4.8 File Structures

As mentioned, the module workflow is divided into two processes that can be performed independently, the preprocessing of the files and the calculation or data processing of the module. This was designed this way as some manual inputs must be introduced into the module, for example when there are bifurcations. The mentioned file can be found in the preprocessing directory. In figure 15 can be observed the full scheme of the file structure of how the information is organized.

The first main file is the file where the Python module can be found. This module is not contained within the workspace environment, as it might be called for different models or other processes that require the utilization of the module. Within this directory, the user will find the preprocessing file where the functions to assign the grid to the GW mesh, and to generate the data frames inputs for the routing calculation that will be stored in the preprocessing directory. The preprocessing directory will be created automatically when the preprocessing Python code is executed.

In the routing classes, the user can find the previously described data structures of the river network, reaches, and time manager. Moreover, in the routing calculation file, the functions to calculate the stage and the leakage can be found. In case the user wants to modify the river stage calculation method it is only necessary to modify that file, and any modification will not affect the information exchange between GW and the SW models. Moreover, through the elaboration of the code, several functions to modify the mesh format have been implemented in case a format change of the vtk file to shape file might be needed the user can refer to Convert Mesh Function python file.

One of the most important functions for the utilization of the module is the River Connection file. Through the execution of this code the connection between reaches, the identification of reaches bifurcations, and tributary connections are identified. As a result, this section is contained in a different file in case further modifications might be needed as there might be a special case where there is a special river connection that was not considered.

Then for each model the user must define a Workspace directory where all the information of the model will be created. The workspace will be the same for the preprocessing and processing of the model as well for the result storage. As several cases might be run in the module, the user must specify in which file they want to store the results. Is highly advice to manage one directory per case when there are several time steps and stress periods, due to the number of files that will be created for the ParaView visualization.

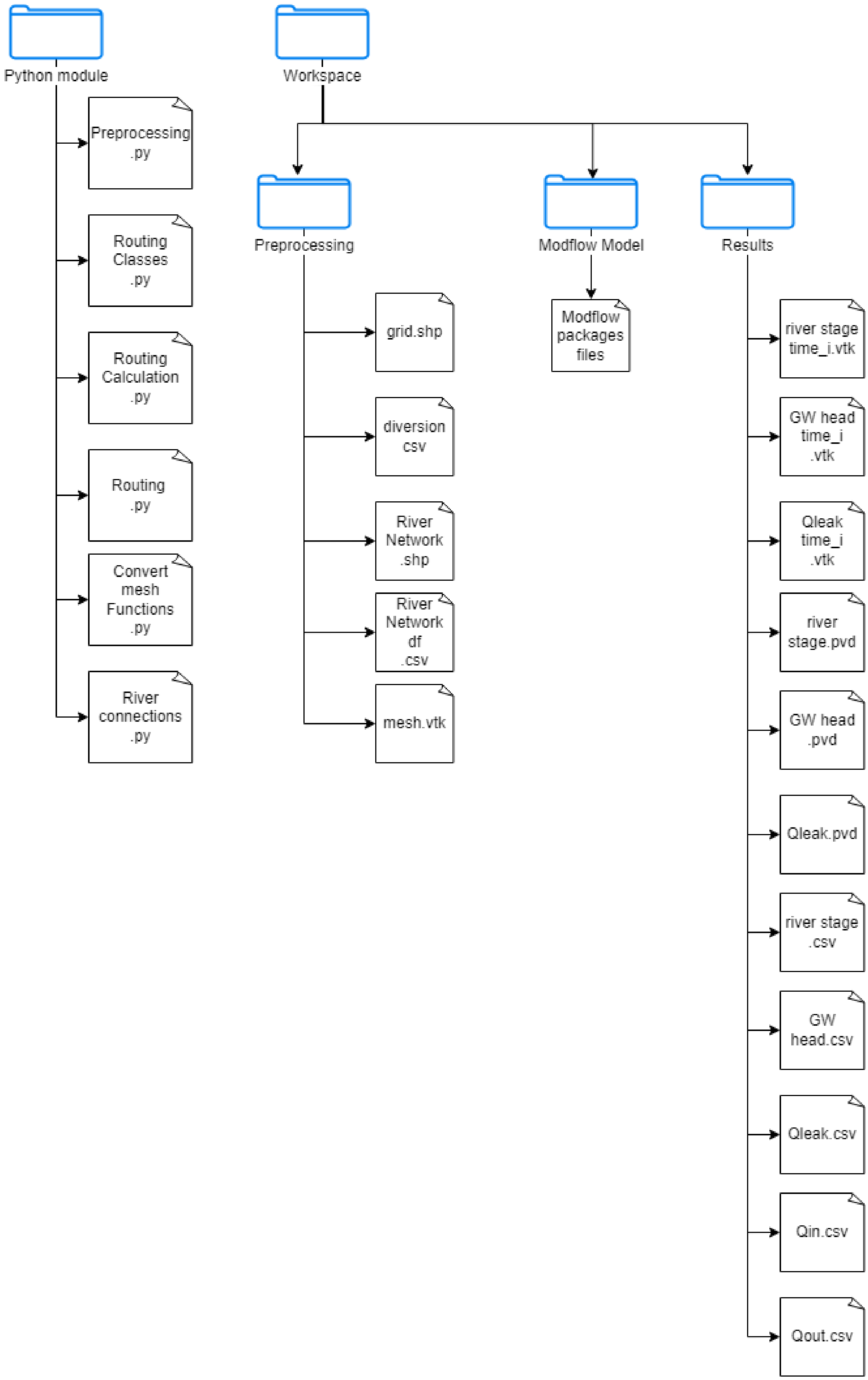

Figure 15. File Structure of the Python Module

4.9 Non resolved Issues and future improvements

As will be observed in this section, there are some issues and improvements that were not possible to solve due to time constraints. The main issue from the presented methodology is related to the boundary conditions of the wells. The first issue is related to the lack of a parameter to limit the pumping of the well to the river stage level when the ground water level is greater than the river stage. The main trigger of this issue is the conductance order of magnitude if the value is too high.

To solve the problem with the well several approaches were considered. The first solution was using the multi aquifer well package. The main advantage of this parameter is the addition of a head limit in the well. The reason why this approach was not selected to solve the over extraction of water from the soil, and the instabilities when the GW level is close to the river stage is due to the complexity and the required parameters for the multi aquifer wells. In comparison to the well package, it is required to specify the top elevation and bottom elevation of the well, the skin of the well and more features of the wells.

Even though the skin parameter can be fixed at 0, the well top and bottom elevation information can be obtained from the disv discretization. The two main reasons for not considering the multi-aquifer well were the complexity of extracting the information from the disv classes and the difference between the MODFLOW and FloPy documentation. In the MODFLOW documentation, it is mentioned that the head limit is only valid for the discharging wells (rate < 0), while the FloPy documentation mentions that the limit is valid for the discharging (rate<0) and injection wells (rate>0). It is important to remember that the FloPy library performs the calculations by running MODFLOW 6, making the MODFLOW 6 documentation more reliable than the FloPy documentation. As a result, and due to time constraints, this approach was discarded to solve the well instabilities as it will solve the issue only for discharging wells.

The second approach to solve the instability was using a shorter time step between iterations. In this case, the user will be required to increase the number of time steps of the stress periods to make the transition between boundaries smoother as can be observed in the result section. However, as it is only possible to input one pumping rate per time stress the Python module will not converge if the leakage rate is not constant in time.

Two approaches on how to combine the pumping rates from the different time steps were found in the literature. The first approach comes from a USGS in which they performed a SW and GW by coupling the 1D numerical model Branch with MODFLOW. To perform the coupling, the leakage from all the time intervals of BRANCH is averaged and this value is introduced as an input into MODFLOW with the WEL package (Swain & Wexler, 1996). This was the selected approach for the coupling as it required fewer changes in the developed code. Currently, this approach is under implementation and requires more work.

The second approach was also developed by using a loosely coupling. The name of the SW numerical modeling was 2dMb and it is a 2D SW numerical model. The performed schematic approach can be observed in figure 16 and it consists of calculating the accumulated leakage of all the time intervals of the 2dMb that are within a MODFLOW stress period (Ruf et al., 2006). To achieve this approach with the current code development it was required to calculate the network with the input flow one time, and then calculate the network with the river stage information of the previous time interval. This approach requires several changes in the network calculation, which is the reason why it was not possible to implement this feature.

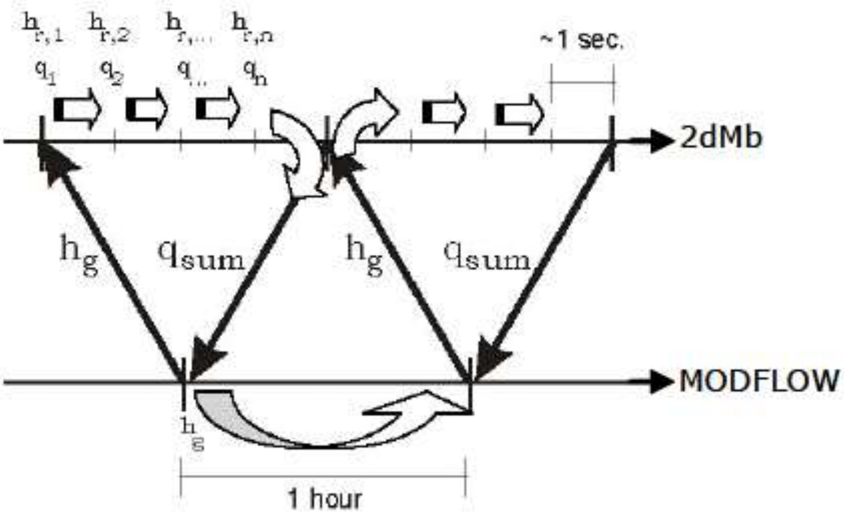

Figure 16. 2dMb coupling approach (Ruf et al., 2006).

For further improvements, it would be necessary to contrast both approaches to estimate the GW leakage and observe which solution offers better performance and stability. Furthermore, the current time discretization of the module is heavily dependent on the MODFLOW time discretization. As a result, giving more flexibility to introduce more inlets into the SW network, and not only one per time stress period like in MODFLOW to refine the calculation of the leakage rate is also necessary to improve the module.

4.10 Limitations

One key step in the development of a module is to analyze the limitations of the same. One of the main limitations is the inability of the module to perform calculations of rivers coupled to a confined aquifer for high conductivities. Through the testing of the module, it was possible to observe that when their aquifer is confined there is no convergence in most of the cases.

Moreover, the result that will be obtained using this module will always have an error. As was explained before, the loosely coupled models work by iteratively running both models and exchanging information until the conversion of error tolerance. Consequently, this module should not be used in cases where an exact result is required.

Another limitation identified through the literature review is the scale on which the model can be run. As was mentioned before the selected coupling method for the module is a loosely coupling reason why it is not advised to use this module for small scale analysis. One example of this limitation is the flood analysis in the riverbanks that are not being considered for the elaboration of this module. Furthermore, this module is running in a steady state, which is the reason why flooding analysis is not going to be properly modeled.

Furthermore, the developed module is the constant re-writing of the MODFLOW files, making the code slower as writing files is computationally expensive. The reason why it was not possible to improve this limitation is because FloPy is not a library to run MODFLOW internally in Python. Instead, this library was created to write the input MODFLOW files, reason why when the user wants to run a simulation it is necessary to input the executable file path of MODFLOW 6.

One alternative to improve this limitation might be searching for other groundwater software that runs fully in Python. Nevertheless, this option was not explored as MODFLOW is robust software with several tools to model other phenomena. This is the reason why computational time was sacrificed to have more functionalities and a more reliable tool.

5 Test cases

5.1 Synthetic networks

5.1.1 Simple case

For the first test case, an artificial network with one reach was tested to observe the behavior of the model and contrast the calculated values with manual calculations, to verify the physical behavior of the model. In this model, the goal was testing the proper physical behavior of dry conditions for the ground water system, dry conditions for the river, and a case where both the river network and the ground water are containing water. Furthermore, the GW model grid has 6 cells, and the river is on the surface of the first layer. In addition, there is a rotation in the GIS representation due to the coordinate reference system, that for this example is epsg:25833. Each dimension of the cells is 1m x1m x 1.41m and has an area of 0.5 m^2.

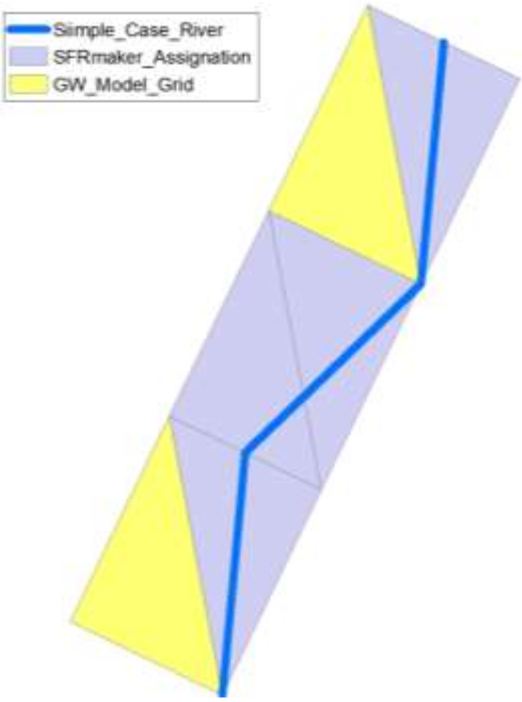

Figure 17. Simple Network layout

5.1.2 Bifurcation and tributary case

For the second test case the simple network was extended, and now it’s composed of 100 cells and the cell dimensions are the same as the simple case. Also, the river network now has a main branch, a tributary and a bifurcation downstream as can be seen in figure 18.

For this network, the river polylines were created from the south to the north reason why the tributary is upstream of the bifurcation. It is important to mention that the elevation of the grid has no impact on the surface water model because the elevation losses are considered in the slope of the reach on the Manning equation.

For this model due to the increase in the number of cells an SFR MODFLOW package model with the same parameters was created and non-manual calculation was performed. This will allow a proper validation of the module and find the limitations of the loosely coupling. Furthermore, this comparison is made as the equations for the calculation for the streamflow routing are the same as SFR, but with a different leakage exchange approach.

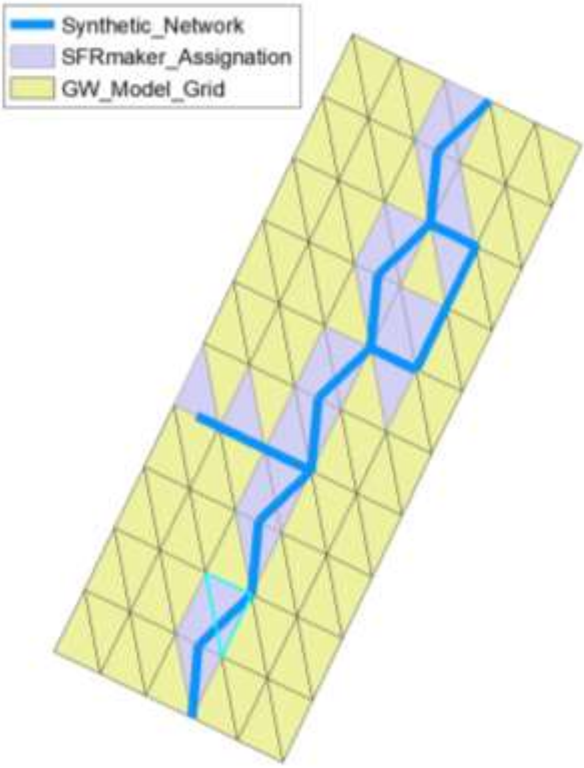

Figure 18. Synthetic Network layout

In addition, a sensibility analysis was made to observe the behavior of the model. The variables that were changed were related with conductance of the riverbed, the conductivity of the riverbed width. These conditions were tested under confined conditions to observe if it was possible to observe a similar behavior between the Python module and the SFR package.

5.2 Module tutorial

Another key aspect of the implementation of a module is explaining properly the usage of the tool. For this purpose, this section has a tutorial on how to use the module. The first step for the preprocessing of the files is opening the preprocessing.py. Within the file, the user must specify the working space (ws), the path of the vtk or vtu mesh file in case there is no shapefile of the grid, the MODFLOW executable path, and the river network shapefile. The reason why the MODFLOW executable is needed is that SFRmaker can only assign the rivers to an unstructured grid by creating an empty MODFLOW model simulation. Furthermore, in case the user does not have the shapefile of the grid, it is possible to convert the vtk file into a shapefile with the function vtu_to_shapefile. Then by using the function river_assignation the files required for the processing step are created.

```
ws ="D:/Master_Erasmus/Thesis/Simple_case"
crs = 25833
shp_name = "model_grid.shp"
vtu_mesh_path = r'D:/Master_Erasmus/Thesis/Simple_case/Simple_Grid.vtk'
vtu_to_shapefile(ws,vtu_mesh_path,shp_name,crs)

#Modflow EXE path
modfl_exe_path = "D:/mf6.4.1/bin/mf6.exe"
#Grid shape path
grid_shap_path = r'D:/Master_Erasmus/Thesis/Simple_case/Simple_Grid.shp'
#River shape file path
river_shape_file = r'D:/Master_Erasmus/Thesis/Simple_case/Simple_net.shp'
#sfrmaker requieres to include the espg in the name of the coordinate system
crs = 'epsg:25833'
project_name = "simple_case"

connect = river_assignation(modfl_exe_path,ws, grid_shap_path,river_shape_file, project_name,crs)
```

Figure 19. Input requirements for the preprocessing

The river network input file must have a specific format in the attribute table to be used in the river assignation. The attribute shapefile must specify an FID, the shape type of the geometry, as this is a 1D approach a polyline must be provided. The ID of the river (COMID), the river that receives the flow downstream (tocomid), in the case of the outlet of the river any id can be provided. A with1 and width2 values of the river, a GNIS_NAME, a name column, the elevation of the initial node of the polyline (elevupsmo), and the elevation downstream (elevdnsmo). It is important to mention that the names of the columns must be the same as the ones presented in table 3.

Shape *	id	name	COMID	tocomid	width1	width2	GNIS_NAME	elevupsmo	elevdnsmo
Polyline	1	main	50	54	5	5	a	10	9.4
Polyline	2	tributary	51	50	5	5	b	10	9.8
Polyline	2	bifurcation	52	50	5	5	c	9.6	9.5

Table 3. Example of the attribute table column names for the SFRmaker library.

After the river assignation, the attribute table of the shape file with the reaches will be different from the original one. In this process, several geometrical and topological variables are calculated, like the slope, the width, the reach id (rno), the cell id in which the reach is contained (node), the reach length, and a column with the name. The variables mentioned above can be inputs for the flow routing module. It is important to mention that the most important information from SFRmaker is the reach id and the cell in which the reach is contained and the name of the river that contains the reach.

Shape	rno	node	k	i	j	iseg	ireach	rchlen	width	slope	strtop	strthick	strhc1	thts	thti	eps	uhc	outreach	outseg	asum	line_id	name
Polyline	1	41	0	0	0	1	1	1.1	5	0.0	10.0	1	1	0	0	0	0	2	3	0.6	53	main
Polyline	2	42	0	0	0	1	2	0.4	5	0.1	9.9	1	1	0	0	0	0	3	3	1.3	53	main
Polyline	3	43	0	0	0	1	3	0.7	5	0.1	9.9	1	1	0	0	0	0	4	3	1.9	53	main
Polyline	4	45	0	0	0	1	4	1.1	5	0.0	9.9	1	1	0	0	0	0	5	3	2.8	53	main
Polyline	5	46	0	0	0	1	5	0.4	5	0.1	9.8	1	1	0	0	0	0	6	3	3.5	53	main
Polyline	6	47	0	0	0	1	6	0.7	5	0.1	9.8	1	1	0	0	0	0	10	3	4.1	53	main
Polyline	7	88	0	0	0	2	1	0.5	5	0.1	10.0	1	1	0	0	0	0	8	3	0.3	51	tributary
Polyline	8	68	0	0	0	2	2	1.0	5	0.1	9.9	1	1	0	0	0	0	9	3	1.0	51	tributary
Polyline	9	48	0	0	0	2	3	1.0	5	0.1	9.8	1	1	0	0	0	0	10	3	2.0	51	tributary
Polyline	10	49	0	0	0	3	1	1.1	5	0.0	9.8	1	1	0	0	0	0	11	4	0.6	54	main
Polyline	11	50	0	0	0	3	2	0.4	5	0.1	9.7	1	1	0	0	0	0	12	4	1.3	54	main
Polyline	12	51	0	0	0	3	3	0.7	5	0.1	9.7	1	1	0	0	0	0	13	4	1.9	54	main
Polyline	13	53	0	0	0	4	1	1.1	5	0.0	9.7	1	1	0	0	0	0	14	6	0.6	55	main
Polyline	14	54	0	0	0	4	2	0.4	5	0.1	9.6	1	1	0	0	0	0	15	6	1.3	55	main
Polyline	15	55	0	0	0	4	3	0.7	5	0.1	9.6	1	1	0	0	0	0	21	6	1.9	55	main
Polyline	16	32	0	0	0	5	1	0.6	5	0.0	9.7	1	1	0	0	0	0	17	6	0.3	52	bifurcation
Polyline	17	31	0	0	0	5	2	0.4	5	0.0	9.7	1	1	0	0	0	0	18	6	0.8	52	bifurcation
Polyline	18	33	0	0	0	5	3	1.0	5	0.0	9.7	1	1	0	0	0	0	19	6	1.5	52	bifurcation
Polyline	19	35	0	0	0	5	4	1.3	5	0.0	9.6	1	1	0	0	0	0	20	6	2.7	52	bifurcation
Polyline	20	36	0	0	0	5	5	0.7	5	0.1	9.6	1	1	0	0	0	0	21	6	3.7	52	bifurcation
Polyline	21	57	0	0	0	6	1	1.1	5	0.0	9.6	1	1	0	0	0	0	22	0	0.6	56	main
Polyline	22	58	0	0	0	6	2	0.4	5	0.1	9.5	1	1	0	0	0	0	23	0	1.3	56	main
Polyline	23	59	0	0	0	6	3	0.7	5	0.0	9.5	1	1	0	0	0	0	0	0	1.9	56	main

Table 4. SFRmaker output attribute table example

In figure 20 we can observe an example of the created files for the simple case. First, there is a shapefile of the grid called model_grid.shp, a CSV with the attribute table of the simple_case_river.shp, a shapefile with the cells that have a reach within them simple_case_grid.shp, and the river network where the rivers were divided into reaches simple_case_river.shp. An example of the visualization of the shape files can be found in figure 17 and 18.

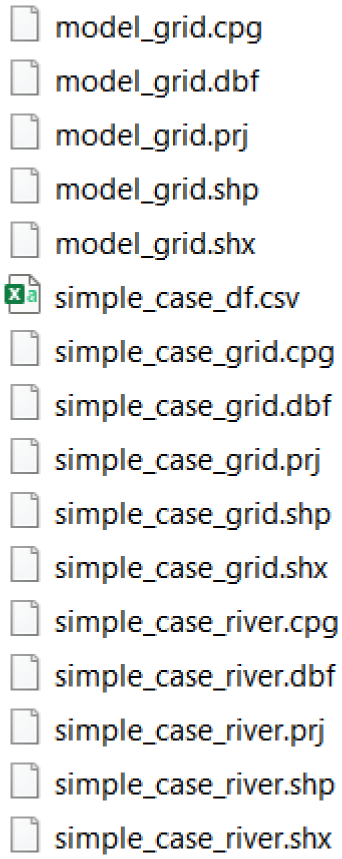

Figure 20. Files created during the preprocessing.

After the results of preprocessing files are created, the paths of the files in figure 21 must be given plus the crs of the shapefile. In case that there are no diversions no path should be provided.

```
#Synthetic Network
crs = 'epsg:25833'
prjct_name = "Synthetic_Network_Conductivity_width_0.1m_dummy"
river_Conections = "D:/Master_Erasmus/Thesis/preprocess_files/mf_river.shp"
sfr_df_path = "D:/Master_Erasmus/Thesis/preprocess_files/mf_df.csv"
diversion_df_path = "D:/Master_Erasmus/Thesis/preprocess_files/mf_div_df.csv"
mesh_vtk_path = r"D:/Master_Erasmus/Thesis/example_basin/mesh.vtu"
modflow_model_path = r'D:/Master_Erasmus/Thesis/workSpaceRouting'
exe_path  = "D:/mf6.4.1/bin/mf6.exe"
result_path = "D:/Master_Erasmus/Thesis/Results"
diversion_df = pd.read_csv(diversion_df_path)
```

Figure 21. Input files for the stream routing module.

Then the time manager class is created by inputting the MODFLOW simulation into the time manager constructor method. After creating the time manager, the river network is created. As can be seen in figure 22, extra information is needed in the river network data frame. For instance, the manning number, riverbed geometry, in the current version the only rectangle is a valid shape, riverbed elevation, slopes, riverbed thickness, aquifer conductivity, top cell elevation, reach ID, cell ID of the reach, river length, and the connection data that list that is created using the river_Connections function.

One of the key difficulties of the river network creation is the number of formats that can be used to store hydraulic and topological information. Therefore, a general input solution was not implemented. The advised workflow for this problem is to complement the created CSV in the preprocessing stage with the information stored at a river discretization level. This can be done by assigning the information to the reaches based on the river name column, which will be shared by the rivers data set and the reaches data set.

```
'''
Time Manager
'''
time_manager= rt_Classes.time_Manager(sim = sim)
stress = time_manager.modflow_stperiods
'''
Routing Creation
'''

sfr_df = pd.read_csv(sfr_df_path)
conect_data = rvrCon.river_Conections(river_Conections,crs = crs)

#Model characteristics
manning = [0.04 for i in range(0,len(sfr_df["name"]))]
geometry = ["rectangle" for i in range(0,len(sfr_df["name"]))]
conductivity_list = [0.0001 for i in range(0,len(sfr_df["name"]))]
rb_elev_list = [2 for i in range(0,len(sfr_df["name"]))]
thickness = [0.5 for i in range(0,len(sfr_df["name"]))]
slopes = [0.01 for i in range(0,len(sfr_df["name"]))]
width = [0.1 for i in range(0,len(sfr_df["name"]))]
conducivity_aquifer_list = [0.0001 for i in range(0,len(sfr_df["name"]))]
top_list = [10 for i in range(0,len(sfr_df["name"]))]

river_Network = rt_Classes.river_Network()

river_Network.create_Network(name_list= sfr_df["name"], slope_list = slopes,
                             reach_id_list = sfr_df["rno"]-1,
                             reach_cell_list = sfr_df["node"], manning_list = manning,
                             cross_section_list= geometry, up_elev_list=sfr_df["name"] ,
                             down_elev_list=sfr_df["name"],
                             conductivity_list=conductivity_list, length_list=sfr_df["rchlen"],
                             thick_list=thickness,
                             width_list = width,rbd_elev_list=rb_elev_list,
                             conect_data =conect_data,top_elev_list=top_list,
                             conductivity_aquifer_list=conducivity_aquifer_list,
                             diversion_df = diversion_df)
```

Figure 22. River network and time class creation

Then the result storage is created by loading the vtk mesh, and the result storage object is created by using the number of cells and the time manager information. Finally, the inflows are added to the network by specifying the reach id, in this case, reach 0 and reach 6 have an inflow, the dictionary is added as a boundary condition to the river network with the function add_boundaries, and the network is calculated without leakage. As was mentioned before, only one stress period inflow has been implemented reason why there is no temporal reference in the inflow dictionary. An example of the code input can be observed in figure 23. After this the loosely coupling calculation takes place and the results are exported as can be seen in figure 24.

```python
'''
Result Storage

'''
mesh = pv.read(mesh_vtk_path)
num_cells = mesh.GetNumberOfCells()
result_stor = rt_Classes.results_Manager(time_manager.modflow_stperiods,
                                         time_manager.modflow_times,num_cells,result_path)

'''
River model flow inputs
'''
#River boundaries Inflows
inflow_dict = {0:50, 6:50}

river_Network.add_Boundaries(inflow_dict)
river_Network.add_Reach_in_Boundaries(bounds_cells)
river_Network = rtCalc.calculate_Network(river_Network, conect_data)
```

Figure 23. Result storage and boundary conditions code section

Synthetic_Network_Conductivity_width_0.1m_dummy_hgw_1.0.vtu	8/11/2023 11:21 PM	Archivo VTU	4 KB
Synthetic_Network_Conductivity_width_0.1m_dummy_hgw_stress_p_0.pvd	8/11/2023 11:21 PM	Archivo PVD	1 KB
Synthetic_Network_Conductivity_width_0.1m_dummy_hriv_1.0.vtu	8/11/2023 11:21 PM	Archivo VTU	3 KB
Synthetic_Network_Conductivity_width_0.1m_dummy_hriv_stress_p_0.pvd	8/11/2023 11:21 PM	Archivo PVD	1 KB
Synthetic_Network_Conductivity_width_0.1m_dummy_qgwf_1.0.vtu	8/11/2023 11:21 PM	Archivo VTU	3 KB
Synthetic_Network_Conductivity_width_0.1m_dummy_qgwf_stress_p_0.pvd	8/11/2023 11:21 PM	Archivo PVD	1 KB
Synthetic_Network_Conductivity_width_0.1m_dummy_qin_stress_p_0.pvd	8/11/2023 11:21 PM	Archivo PVD	1 KB
Synthetic_Network_Conductivity_width_0.1m_dummy_qout_stress_p_0.pvd	8/11/2023 11:21 PM	Archivo PVD	1 KB
Synthetic_Network_Conductivity_width_0.1m_dummy_results_hgw_stress_p0.csv	8/11/2023 11:21 PM	Archivo de valores...	4 KB
Synthetic_Network_Conductivity_width_0.1m_dummy_results_hriv_stress_p0.csv	8/11/2023 11:21 PM	Archivo de valores...	3 KB
Synthetic_Network_Conductivity_width_0.1m_dummy_results_qgwf_stress_p0.csv	8/11/2023 11:21 PM	Archivo de valores...	3 KB
Synthetic_Network_Conductivity_width_0.1m_dummy_results_qin_stress_p0.csv	8/11/2023 11:21 PM	Archivo de valores...	3 KB
Synthetic_Network_Conductivity_width_0.1m_dummy_results_qout_stress_p0.csv	8/11/2023 11:21 PM	Archivo de valores...	3 KB

Figure 24. Exported files from the result storage class

6 Results and analysis

6.1 Module physical verification

6.1.1 Dry soil and river

For this case a state where there is no flux in the river and there is no water in the aquifer. This test was intended to check whether the module can manage a scenario without water and has no relevant result to be analyzed. In figures 25 and 26 we can observe that there is no leak and there is no water head, which is physically correct.

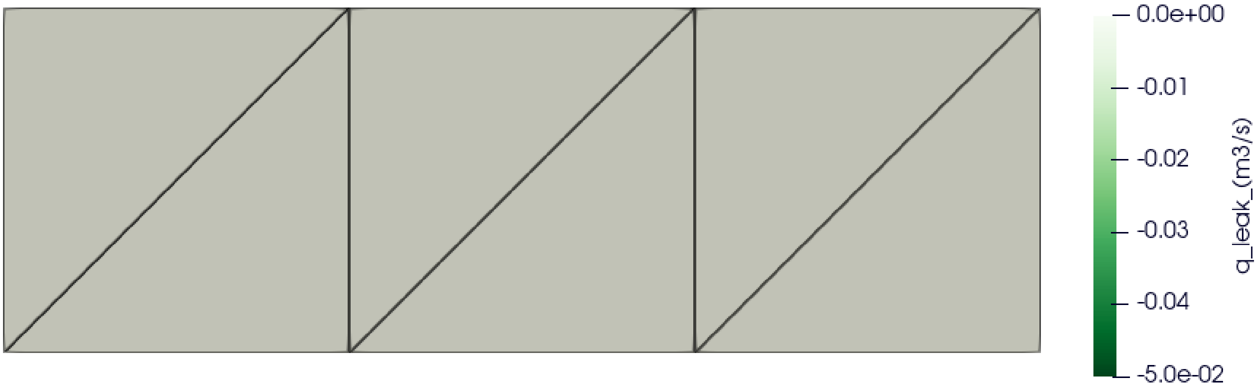

Figure 25. Leakage in the ground water under dry conditions

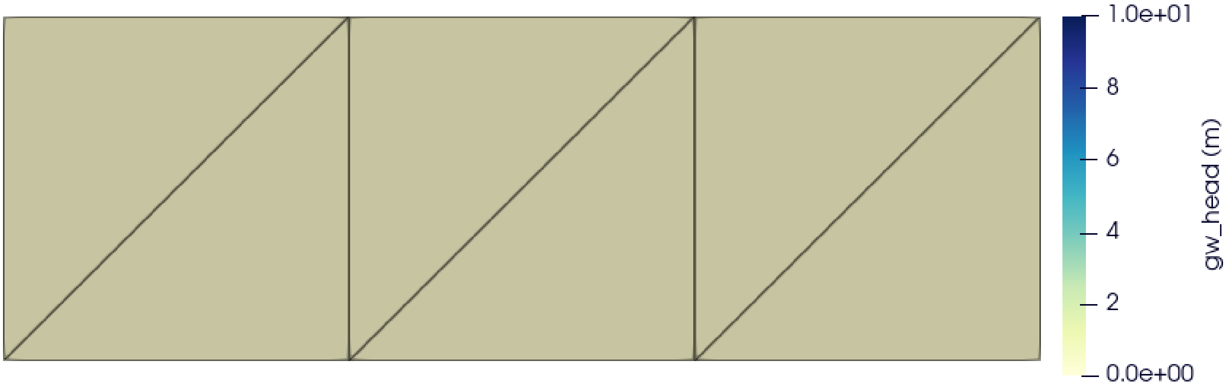

Figure 26. Ground water head under dry conditions

6.1.2 Dry soil and flow in the river manual calculation

For this model, a constant slope, top elevation, width manning, riverbed conductivity, and riverbed thickness were fixed for all the reaches and an inlet flow (qin) of 10 m3/s. The constant input values can be found in table 5. However, the value of the length of the reach is calculated from the geometry of the GIS river shapefile was not modified. In addition, for the GW model, the initial conditions of the head were set at 0 m and the head boundaries were set at 0 m too. The width of the river is bigger than the cell edge, this value was selected to observe the behavior of the module when there is a great inlet due to a big conductance.

qin (m^3/s)	10
Initial head (m)	0
Top elevation (m)	10
Manning ($s/m^{1/3}$)	0.04
width (m)	5
slope (-)	0.01
rb_elev (m)	2
rbot (m)	1.5
Conductivity river (m/s)	0.001
Conductivity aquifer (m/s)	0.0001
Thickness (m)	0.5

Table 5. General model parameters

In table 6 the results obtained from the Python module and the results calculated using Excel with the equations presented in the methodology. The expected behavior when the soil is dry is an exchange from the river to the ground, and due to the river flow and the conductance not being high enough, the pressure will not exceed the riverbed bottom level. In the results, it is possible to observe that the model behaves in a logical physical behavior. Nevertheless, there is a strange behavior in reach 4 where there is a GW level of 0 m even though there is a reach. This behavior is due to the location of the reach in a constant head boundary, and for those cases, no leak is calculated in the reach in the Python module as it is assumed there is no leakage.

It is important to mention that the results are presented from the perspective of the river, therefore the leakage is negative as the river is losing water. Furthermore, the SRME of the manual leakage vs the leakage calculated in the reaches using the Python module is $5.73*10^{-6}$ for the Q leak, resembling a proper behavior for this case. Moreover, reach 4 is not included in the SRME as the value is not calculated by the module due to the location over a boundary. Also, the selected execution time was 1 second with no time steps. This decision was made as the selected conductance was high and the levels of the GW head and river stage will reach the same depth as it will be shown later in this section when the conductance is high.

Reach ID	River length (m)	conductance (m^2/s)	GW head Python (m)	Stage final node Python (m)	Q leak Python (m^3/s)	H riv manual initial node (m)	Stage manual initial node (m)	Q leak manual (m^3/s)	Q final node (m^3/s)	Stage final node (m)
1	1.118	0.011	0.205	2.874	-0.015	0.875	2.875	-0.015	9.985	2.874
2	0.373	0.004	0.068	2.874	-0.005	0.874	2.874	-0.005	9.980	2.874
3	0.745	0.007	0.136	2.873	-0.010	0.874	2.874	-0.010	9.969	2.873
4	1.118	0.011	0.000	2.872	0	0.873	2.873	0	9.954	2.873

Table 6. Manual calculations and Python module results using

6.1.3 Dry river manual calculation

For this case it was assumed that the river was dry and that the initial GW head was the same as the cell top being this pressure 10 m and the head boundaries were set at 0 m once again, the change in the parameters can be observed in table 7. The reason why the boundaries are set at 0 m is to avoid the water to leave the model through the ground. In addition, the same physical properties of the reaches from the previous case were kept the same. The expected physical behavior is that the water level of the river will increase with a flow coming from the ground and that the river stage will not exceed the GW pressure, the reason why the flow will be only dependent on the ground. The execution time was again 1 second.

qin (m^3/s)	0
Initial head (m)	10

Table 7. Changed parameters for dry river conditions.

In table 8 can be found the results from the case where the river is dry. One important behavior that is important to mention is that the river is calculated to achieve a steady state in the river. This means that the developed module will not be able to capture the time that it takes for the river to fill and the propagation of the water, as it is a steady state for a set of given conditions. Moreover, the manually calculated values correspond to a water depth at second 1 with a high conductance, in real life the water depth will not increase that fast in that small fraction of time. In addition, under those conditions, there will be uncertainty about the saturation of the soil, which is a limitation of the model.

Reach ID	River length (m)	conductance (m^2/s)	GW head Python(m)	H stage Python(m)	Q leak Python (m^3/s)	Initial Stage manual (m)	Q Leak manual (m^3/s)	River Stage (m)
1	1.118	0.011	8.965	2.048	0.078	2.000	0.078	2.047
2	0.373	0.004	9.624	2.057	0.028	2.047	0.028	2.057
3	0.745	0.007	9.283	2.073	0.054	2.057	0.054	2.073
4	1.118	0.011	0.000	2.071	0	2.073	0	2.071

Table 8. Manual calculation for dry river case

6.1.4 River stage higher water depth than ground water manual calculation

From the previous case where the ground was totally dry, we were able to observe that the 10 m^3/s results in not a higher river stage. For this case the flow was increased in one order of magnitude and the ground water head was set at a level of 3 meters that is above the river bottom to test the interaction between ground and soil. In addition, 0m constant head boundaries were added once again to observe only the interaction between the GW and SW.

qin (m^3/s)	100
Initial head (m)	3
Soil upstream pressure (m)	0
Soil downstream pressure (m)	0

Table 9. Changed parameters for dry river conditions.

The results of this test are presented in table 10, once again we can observe that the model is behaving properly as the water is flowing from the river to the ground. In addition, the root square mean error for this case is $6.39*10^{-6}$ meaning that the manual calculation and the Python calculation has reached a similar result.

Reach ID	River length (m)	conductance (m^2/s)	GW head Python (m)	Stage final node Python (m)	Q leak Python (m^3/s)	H riv manual initial node (m)	Stage manual initial node (m)	Q leak manual (m^3/s)	Q final node (m^3/s)	Stage final node (m)
1	1.118	0.011	3.315	5.482	-0.024	3.482	5.482	-0.024	99.976	3.482
2	0.373	0.004	3.117	5.482	-0.009	3.482	5.482	-0.009	99.967	3.482
3	0.745	0.007	3.222	5.481	-0.017	3.482	5.482	-0.017	99.950	3.481
4	1.118	0.011	0.000	5.480	0.000	3.481	5.481	0	99.950	3.480

Table 10. Calculation for non-dry conditions where the water flows from the river to the ground.

6.1.5 GW level close to riverbed bottom with a non-dry river SFR comparison

This case was created to observe the behavior of the module through time and when the GW head is close to the limit that defines if the leakage is dependent of the GW head or not. For the first approach only one stress period was implemented, and the results presented instabilities when the GW head was close to the riverbed bottom. To assess the effectiveness of calculating this value an SFR model of the simple case was created to observe the behavior of both cases. In table 11 it is possible to observe the input values for both SFR and the MODFLOW model.

qin (m^3/s)	100
Initial head (m)	7
Soil upstream pressure (m)	0
Soil downstream pressure (m)	0

Table 11. Changed parameters for dry river conditions.

In table 12 are registered the results of the calculation using the developed module for a stress period length of 10 seconds without intermediate time steps. In this case, there was no convergence of the results due to the exceedance of the riverbed bottom (rbot) in reach id 1 by 0.48 m. Because of this, the leakage is first calculated using equation 6 for iteration 39 and equation 5 for iteration 40 and will go back to equation 6 for iteration 41 with the same results as iteration 39, meaning the calculation is stuck in an oscillation. This behavior can be explained as the system of equations that is being solved changes abruptly. Furthermore, this shows that the module approach is sensible when the GW depth or the SW stage is close to the limits that define the SW and GW exchange.

39 iterations					40 iterations				
cell id	reach id	gw head time 10.0 (m)	h stage time 10.0 (m)	q leak time 10.0 (m3/s)	cell id	reach id	gw head time 10.0 (m)	h stage time 10.0 (m)	q leak time 10.0 (m3/s)
0	No_reach	0	0.000	0.000	0	No_reach	0	0.000	0.000
1	1	1.232	2.874	-0.015	1	1	2.048	2.874	-0.009
2	2	0.683	2.874	-0.005	2	2	0.683	2.874	-0.005
3	3	1.365	2.873	-0.010	3	3	1.365	2.873	-0.010
4	No_reach	0.000	0.000	0.000	4	No_reach	0.000	0.000	0.000
5	4	0.000	2.872	0.000	5	4	0.000	2.873	0.000

Table 12. Results of the test case with 39 and 40 iterations using the PYTHON module using a time step of 1 and a duration of 10 seconds

To comprehend more about the issue behavior and observe if the SFR package is sensible too, a version of the simple model using the SFR package with the same hydraulic inputs and the same stress period length was created. To assess the sensibility of this case the same model was run by using different time intervals for the calculation. In table 13 the results with the same conditions as the previous model are presented.

cell_id	reach_id	gw head time 10 (m)	h stage time 10 (m)	q leak time 10 (m3/s)
0	No_reach	0	0	0
1	1	1.720	2.874	-0.0129
2	2	0.683	2.874	-0.0051
3	3	1.365	2.873	-0.0102
4	No_reach	0	0	0
5	4	0	2.873205704	0

Table 13. Results of SFR using a time step of 10 and a duration of 10 seconds

Then the number of time steps for the stress period was increased to 10. In table 14 the data for the last two time periods are presented. The results with all the time steps can be found in appendix 2. The first thing to notice by comparing both cases is that SFR is also sensible for cases close to the limits of calculation. In the reach number 1, where instability is presented the gw head changed from 1.72 m to 1.925 m, even though both models have the same inputs except for the time step.

cell_id	reach_id	gw head time 9 (m)	h stage time 9 (m)	q leak time 9 (m3/s)	h stage time 10 (m)	gw head time 10 (m)	q leak time 10 (m3/s)
0	No_reach	0	0	0	0	0	0
1	1	1.784	2.874	-0.0122	2.874	1.925	-0.0106
2	2	0.614	2.874	-0.0051	2.874	0.683	-0.0051
3	3	1.228	2.874	-0.0102	2.874	1.365	-0.0102
4	No_reach	0	0	0	0	0	0
5	4	0	2.873	0	2.873	0	0

Table 14. Results of SFR using a time step of 1 and a duration of 10 seconds

Based on the improvements of the head for the SFR package, the model with the 1-second time step was also run using the developed module. In this case, no improvement was observed in the calculation of the module. Furthermore, both results were graphed based on the theory presented in figure 8, where the slope of the ground leakage and the GW head is the conductance. Figure 27 and 28 are made at time 8 as in both numerical models the transition of constant leak to a decrease in the flow from the river to the soil takes place at second 8. Both results were plotted to observe if there was a change in the slope due to the instability, but in both cases, the calculated slope fulfilled the theory as the conductance for this reach was 0.0112 m2/s. From these results, it is possible to conclude that the loosely coupled is sensible when the calculated value approaches the riverbed bottom due to the transition of equations.

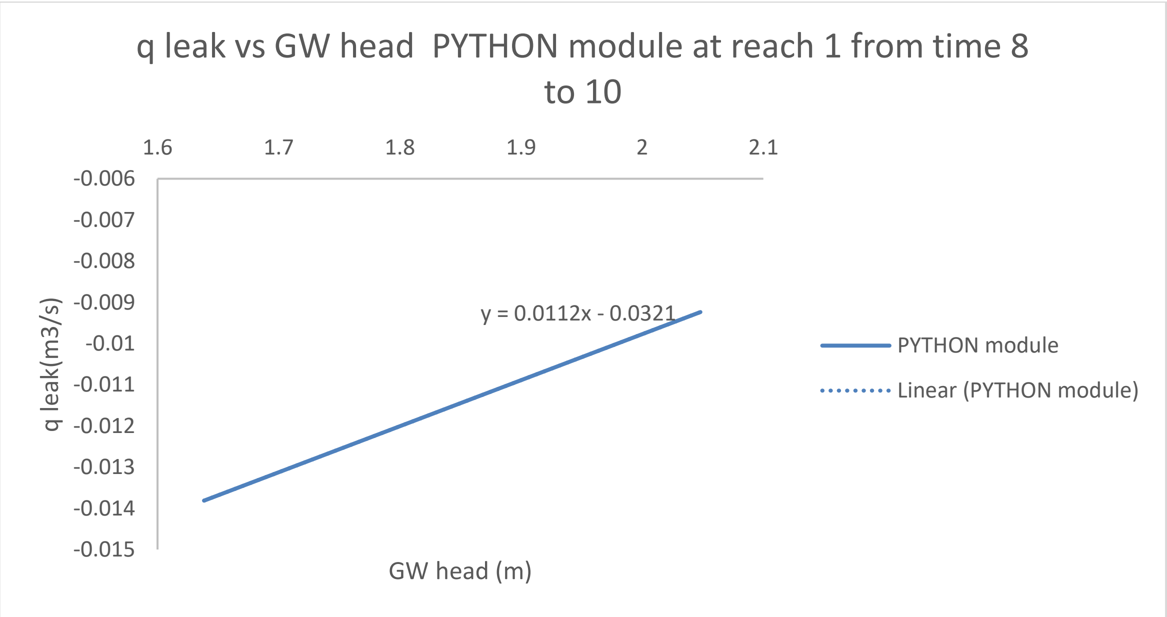

Figure 27. q leak vs ground water head using the developed module at reach 1

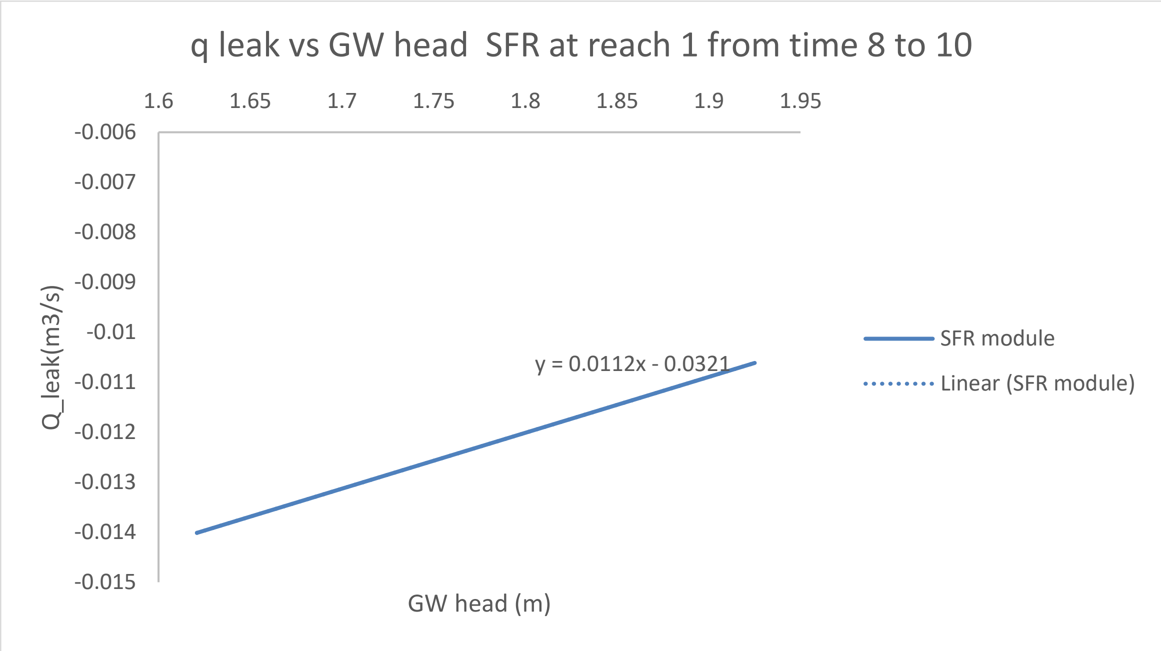

Figure 28. q leak vs ground water head using SFR at reach 1

As can be seen in figures 29 and 30 when the Python module is able to model the time step in which the leakage coming from the river to the soil starts to reduce. However, when the water depth approaches the riverbed bottom the slope of both functions starts to differ. This behavior can be seen better in figure 30 where the growth of the GW head stops being linear and starts to reduce smoothly. It is possible that SFR implemented a smoothing reduction function for the GW head when it approaches the riverbed bottom.

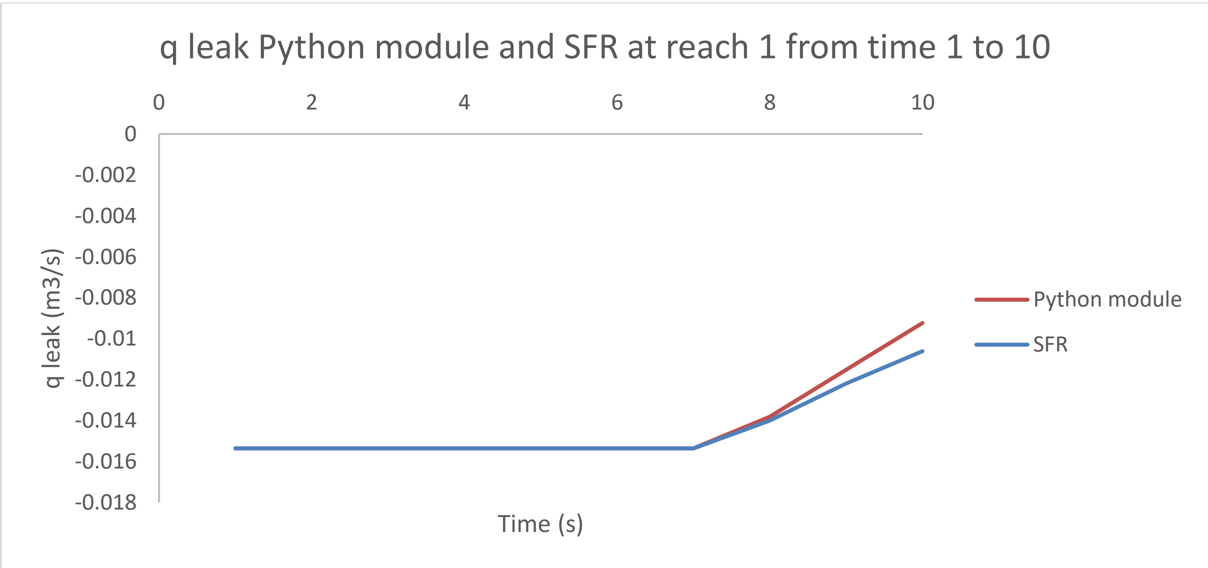

Figure 29. Leakage evolution in time at reach 1 using Python module and SFR.

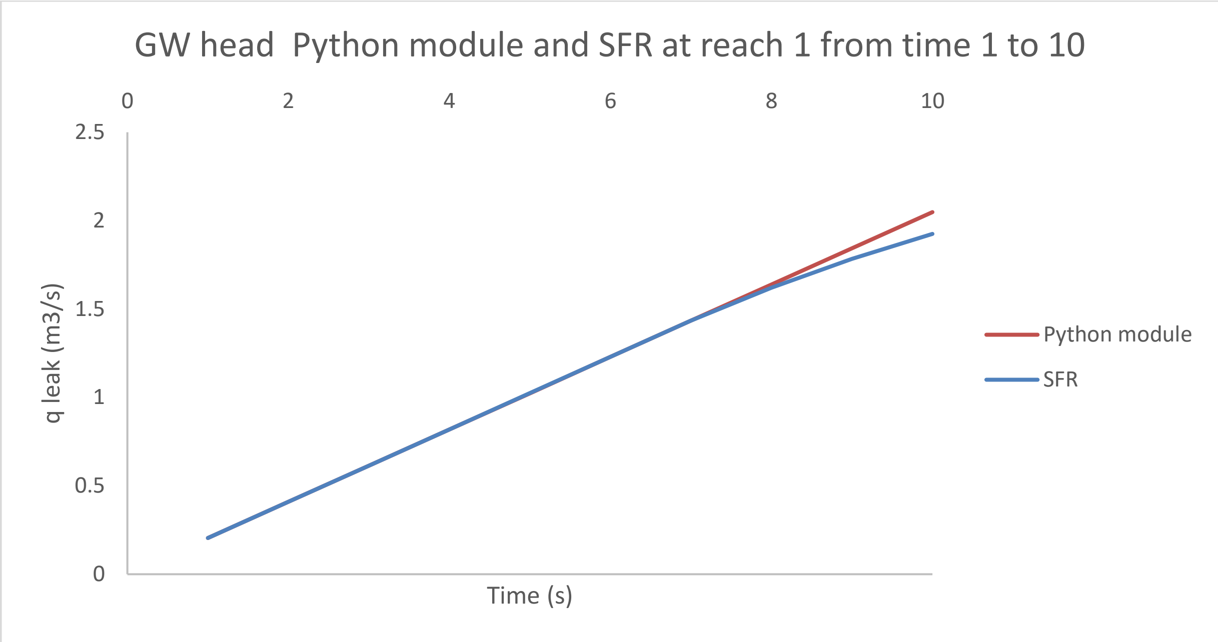

Figure 30. GW head evolution in time at reach 1 using Python module and SFR.

Due to the riverbed width being greater than the cell edges there is going to be a faster movement of water as the conductance increases with the width. Thanks to this it was possible to observe the evolution of the interaction between SW and GW and how they will reach an equilibrium state in a small period of 14 seconds when no water is entering or leaving the system. However, in real life, this process might take several days, and the leakage rate will not change unless there is a significant change in the river inlets.

Furthermore, when the conductivity in the riverbed is high the model will not reach a stationary state and the created module will not function properly as was observed in this case. This means that the loosely coupling will not work with transient models in most cases and it will be more sensible to the time step when there is a fast infiltration. For instance, when the stress period duration was higher than the 14 seconds that it takes to reach equilibrium between the river stage and GW head, with only one time step, the model crashed and an error from MODFLOW was raised. This is because MODFLOW is calculating one discrete GW head at a discrete time, and it is not possible for a loosely coupling approach to know information about the previous times in comparison to a fully coupled approach.

6.2 Python module comparison with MODFLOW SFR under confined conditions

For the sensibility analysis, a base case and then different variations were performed by changing only inputs from the base case. The input values of the base case are presented in table 15, and the conductivity was assumed as constant for the riverbed and the aquifer. The initial head was set at 10 meters to observe if there was an effect of confinement due to the initial conditions. Based on the previous results the river width was set once again at 5m to observe once again the sensibility of the module at extreme conditions. In addition, most of the selected parameters for the sensibility analysis are not realistic and were selected to observe the behavior of the module and comprehend under which conditions the module will not be able to achieve a satisfactory result.

Qin reach 7 (m^3/s)	50
Qin reach 7 (m^3/s)	50
Initial head (m)	10
Cell top elevation (m)	5
Manning ($s/m^{1/3}$)	0.04
width (m)	5
slope (-)	0.01
rb_elev (m)	2
rbot (m)	1.5
Conductivity river (m/s)	0.0001
Conductivity aquifer (m/s)	0.0001
Thickness (m)	0.5
Soil upstream pressure (m)	0
Soil downstream pressure (m)	0

Table 15. Base case input parameters for the sensibility analysis

In this case, there is also a bifurcation reason why it is necessary to specify the diverted flow. The diversion of the flow is presented in table 16 based on the diversion CSV file format obtained in the preprocessing stage. Moreover, the bifurcated flow was divided in equal quantities for the two downstream reaches for all the cases that were performed during the sensibility analysis.

Diverted reach	Out reach	Reach type	q (m3/s)
12	13	main	0
12	16	diversion	50

Table 16. Diversion input parameters

After setting up both modules the results were compared for the GW head h stage and qleak by calculating the RSME of between SFR and the developed module. A table containing the results from both approaches can be found in appendix 3. From the results we can observe that the calculated parameter with the highest error is the GW head. This can be explained as the size of the cell is small, making it more sensible to increase the water depth with a smaller amount of leakage.

gw head RSME	h stage RSME	q leak RSME
0.001	0.000	0.000

Table 17. RSME comparison between SFR and the PYTHON module results.

In addition to the calculation of the RSME, graphics of the GW head and the leakage flow introduced into the cell of the GW model can be observed in figures 31 and 32. From the initial conditions, we can observe that even though there was an initial head condition of 10m, 5m above the cell top elevation, the excess water was taken out of the model, and a successful run was performed. The step in which the excess of water is taken out of the system is when the GW model is run without leakage. In addition, when the graph of the leakage is observed, it is possible to notice that there is an inflow from the SW to the GW model. This can be explained as these two middle cells come after the tributary connection with the main river. Consequently, there is an increase in the SW stage due to a higher flow which increases the stage level above the GW head.

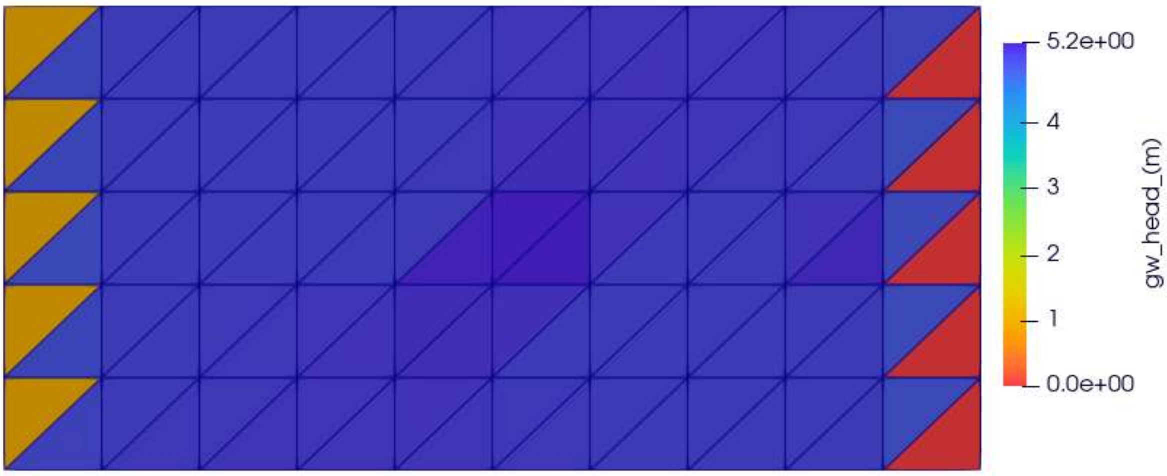

Figure 31. Ground water head using Python module.

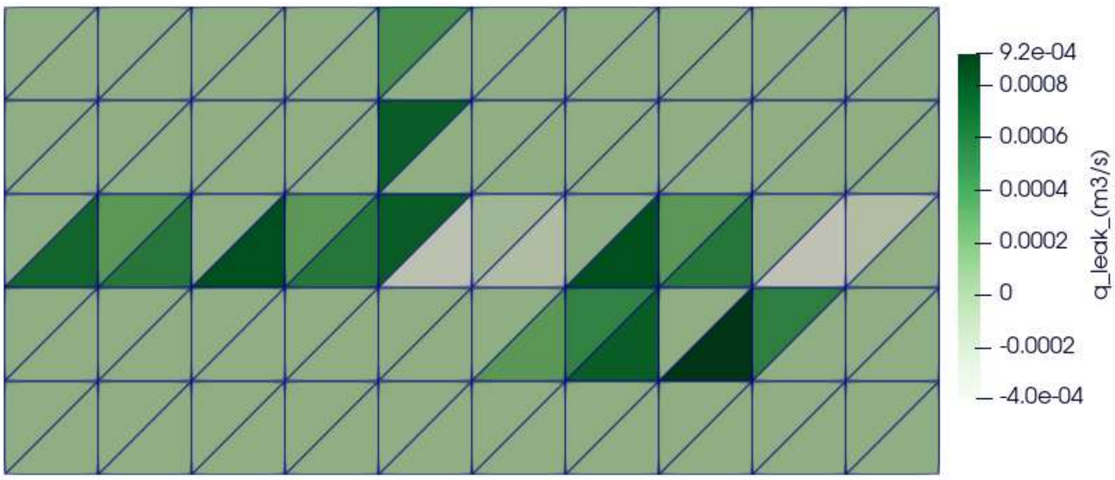

Figure 32. Leakage from the river base case using Python module.

6.2.1 Riverbed conductivity and initial head sensibility

6.2.1.1 Riverbed conductivity 0.001 m/s with 5m initial head

For this approach, the conductivity of the riverbed was increased by one order of magnitude until an error was found. The conductivities until reaching the error. The simulated conductivities are presented in table 18. As can be seen, the module was not able to converge after the first conductivity, because of this it was decided to reduce the initial head to observe the triggering of the error. No comparison with the SFR was made as there was non-convergence from the module.

Conductivity river (m/s)	0.001

Table 18. Number of conductivities analyzed before non-convergence.

The graph of the GW head is presented in figure 33, and we can observe heads 200 m in the GW domain. This means that the aquifer has a confined reason why the head is above the top level. This example shows that in comparison to the base case having heads close to the top elevation of the model with a high conductivity on the riverbed will generate instability in the calculation using the developed module. From this behavior, it is possible to conclude that further research on how to implement confined conditions must be done to achieve a more general module.

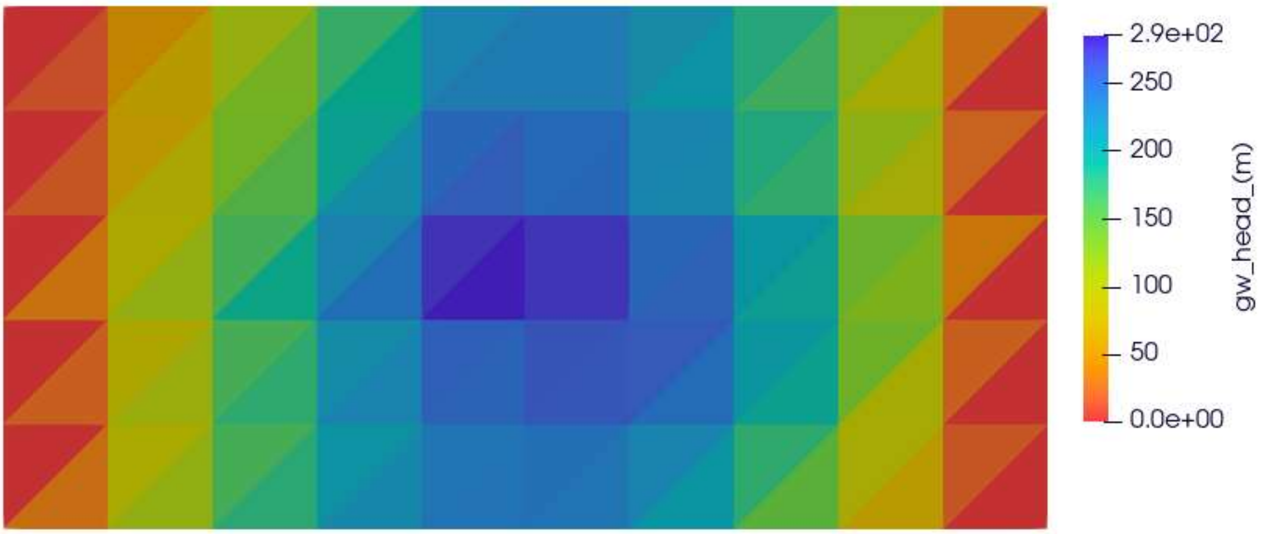

Figure 33. Calculated heads using PYTHON module non-convergence case.

6.2.1.2 Riverbed conductivity 0.001 m/s with 2m initial head

As it was mentioned before the initial head was set close to the top elevation to observe the effect of a confined aquifer. Once again, the RSME and the graphs of the leakage and the GW head are presented below. In comparison to the previous case, there is no leak from the soil to the river as the initial GW head has decreased considerably, showing the impact of initial conditions in the model. Once again, the calculated RSME is low showing that both numerical models are reaching similar results.

gw head RSME	h stage RSME	q leak RSME
0.00003	0.00027	0.00000

Table 19. RSME comparison between SFR and the PYTHON module results.

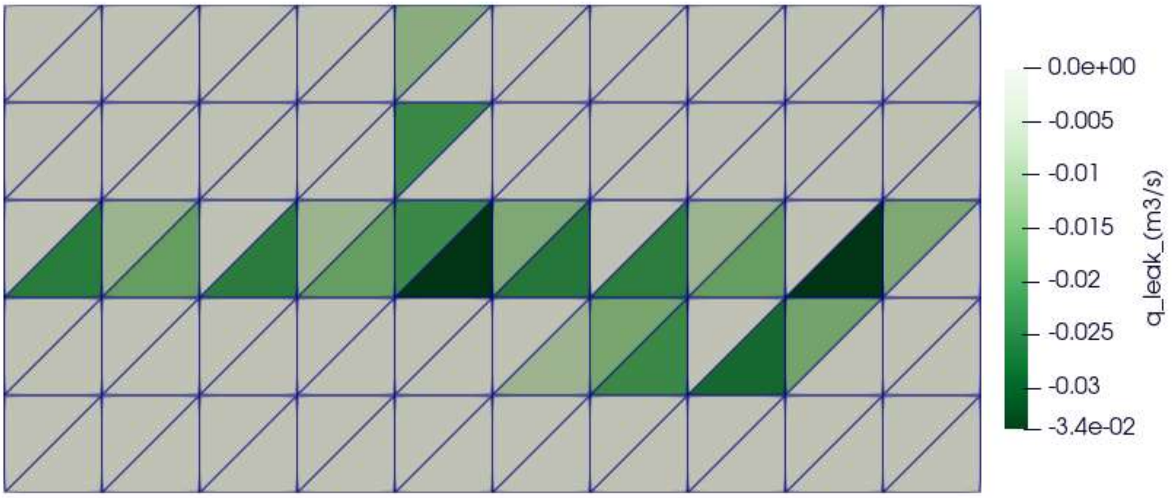

Figure 34. Leakage from the river base case using Python module.

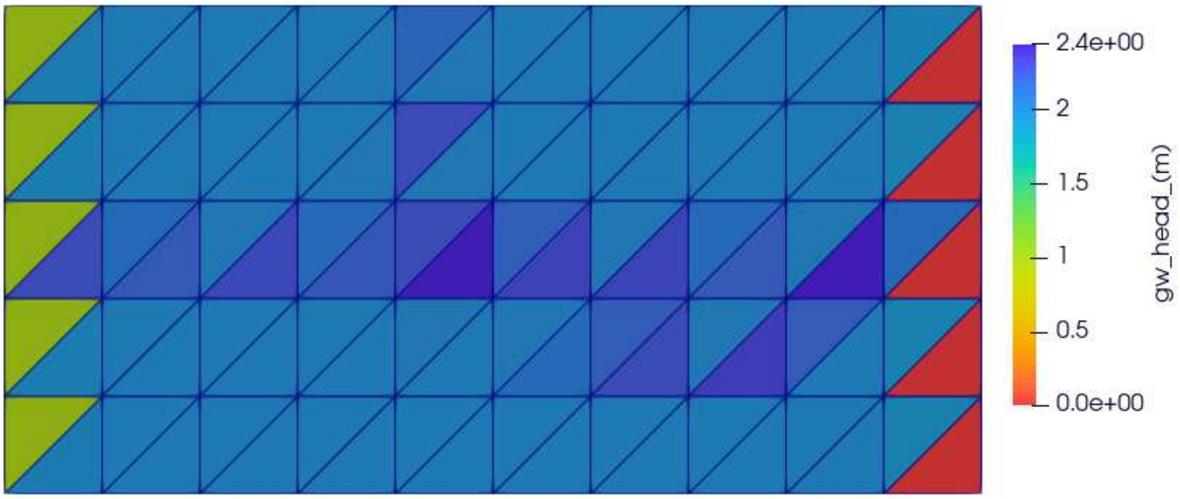

Figure 35. Ground water head using Python module.

From this analysis, it was possible to observe once again that loosely coupling is more sensible to conductivity. In addition, when the conductivity is increased the sensibility of the module to the initial parameters will be increased too. Under high conductivity conditions, more water will flow from the riverbed to the ground causing confinement at the initial iterations, and not allowing the module to converge.

6.2.2 Riverbed sensibility analysis for the river width

For the riverbed width sensibility analysis, the value was reduced to observe how it affects the results. In table 20 the tested widths are presented. As the river width has been reduced and the flow remains the same, it is expected that the SW depth will increase due to the reduction of the width. Through this reduction it will be possible to observe the behavior of the module when the SW exceeds the top elevation of the cell. As it was mentioned before this module was not developed to perform in flooding cases, nevertheless it is important to know the behavior as an edge case.

Width (m)
1
0.1

Table 20. Input widths for the sensibility analysis.

6.2.2.1 Riverbed width 1meter

For the river, the calculation of the RSME is presented in table 21. The first thing that it is possible to observe is that the river stage and the leakage have a small error while the GW head higher error. Due to the disparity of the RSME of the 3 variables and the achievement of convergence of the model, both model results are graphed and compared as the RSME is too high in the GW head and small for the stage, and the leakage.

gw head RSME	h stage RSME	q leak RSME
4.36	0.00022	0.00042

Table 21. RSME of the case with a width of 1m

In figure 36 the GW head is presented, and it is possible to observe that the Python module and the SFR module are estimating a GW head above the cell top elevation. Nevertheless, in comparison to the conductivity case in chapter 6.2.1.1 where the head exceeded by two orders of magnitude the calculation of SFR, having a small conductivity allowed the Python module to achieve a closer estimation of the GW, as in chapter 6.2.1.1the calculated heads were above the 200 meters.

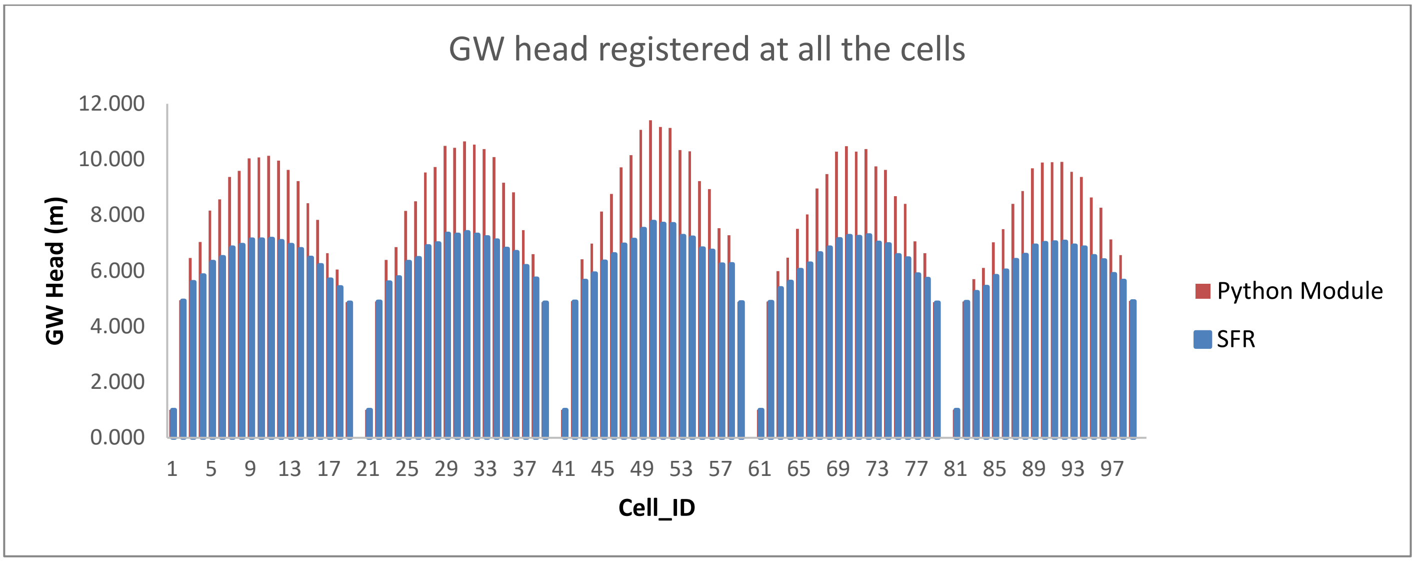

Figure 36. GW head at each cell graph using the Python module and SFR.

The biggest difference in the behavior of both models can be observed in figure 37. While in the SFR model, it is assumed that the water is flowing from the river to the soil, the Python module is indicating the opposite behavior. In addition, the small calculation of the RSME can be explained as both values have a similar value with the same magnitude and a different sign. From this result, it is observed the necessity of using a different methodology for the calculation of the RSME for the q leakage. In this case, a normalization of the results will be necessary as the leakage is low in magnitude.

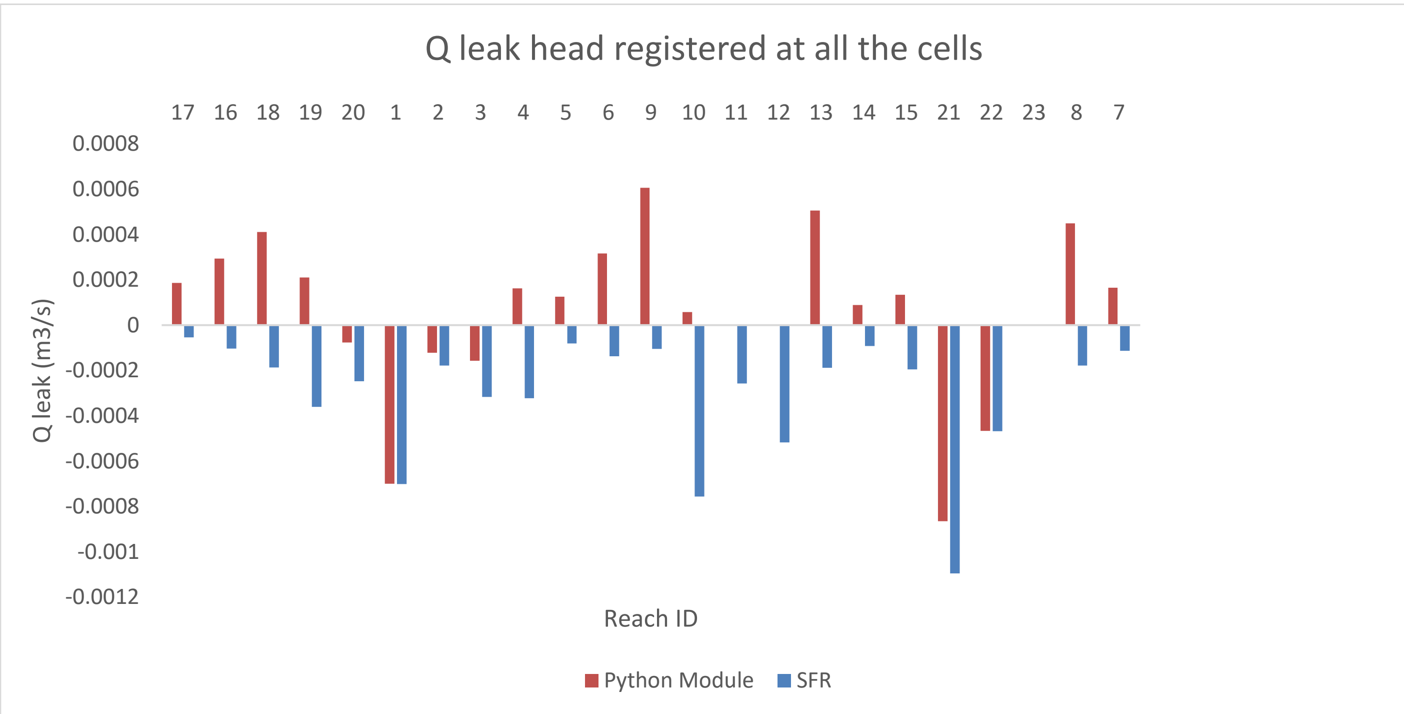

Figure 37. Q leakage at each cell graph using the Python module and SFR.

In figure 38 the river stage is the least sensible value as the calculation of SFR, and the Python module is the same. The cells where the river stage is equal to 0 m^3/s are cells where there are no reaches. The reason that might explain the stability of the river stage is the high-water flow that is flowing across the river in comparison to the leakage. This lack of sensibility might change if the river flow and the leakage have a similar order of magnitude.

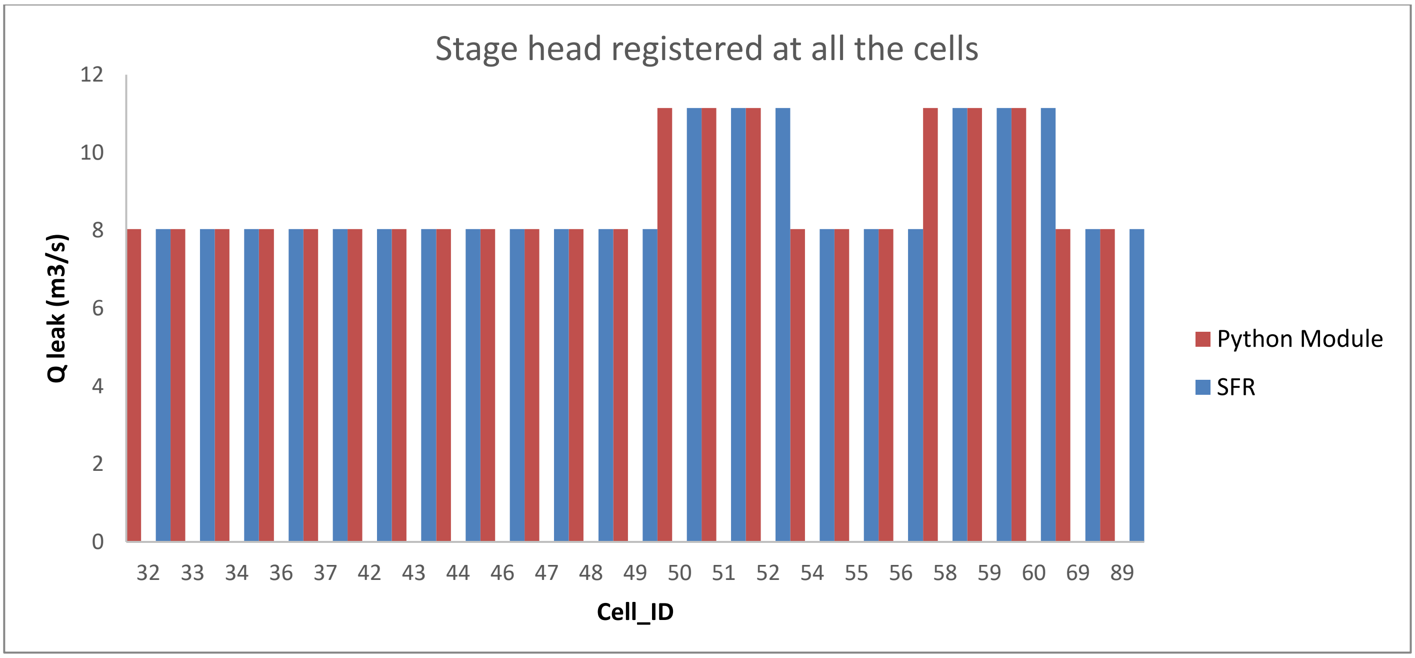

Figure 38. River stage at each cell graph using the Python module and SFR.

In addition to observing previous behaviors from the ones that have being explained before, it was possible to observe that the river stage is less sensible to the leakage than the GW head. Moreover, this means that using the river stage as a convergence criterion when there are high flows might lead to early convergence of the model with a poor calculation of the GW head.

6.2.2.2 Riverbed width 0.1meters

When the riverbed width was reduced to 0.1m both SFR and the Python module were producing similar results. In this case, even though the aquifer reached a confined condition, the GW head was similar for both numerical models. Two explanations for the similarity in the results were found. The first explanation is related to the conductance that will be on the order of magnitude lower than in the previous case making the model more stable. The second explanation is like the explanation given in chapter 6.1.5, where the GW head was close to the river bottom. In the 1 m riverbed width case, the GW head was close to the soil top elevation, which might have triggered instability. As both cases converged to similar solutions more comments on the results are not made.

GW head RSME	h stage RSME	q leak RSME
0.00001	0.00005	0.00000

Table 22. RSME of the case with a width of 0.1m

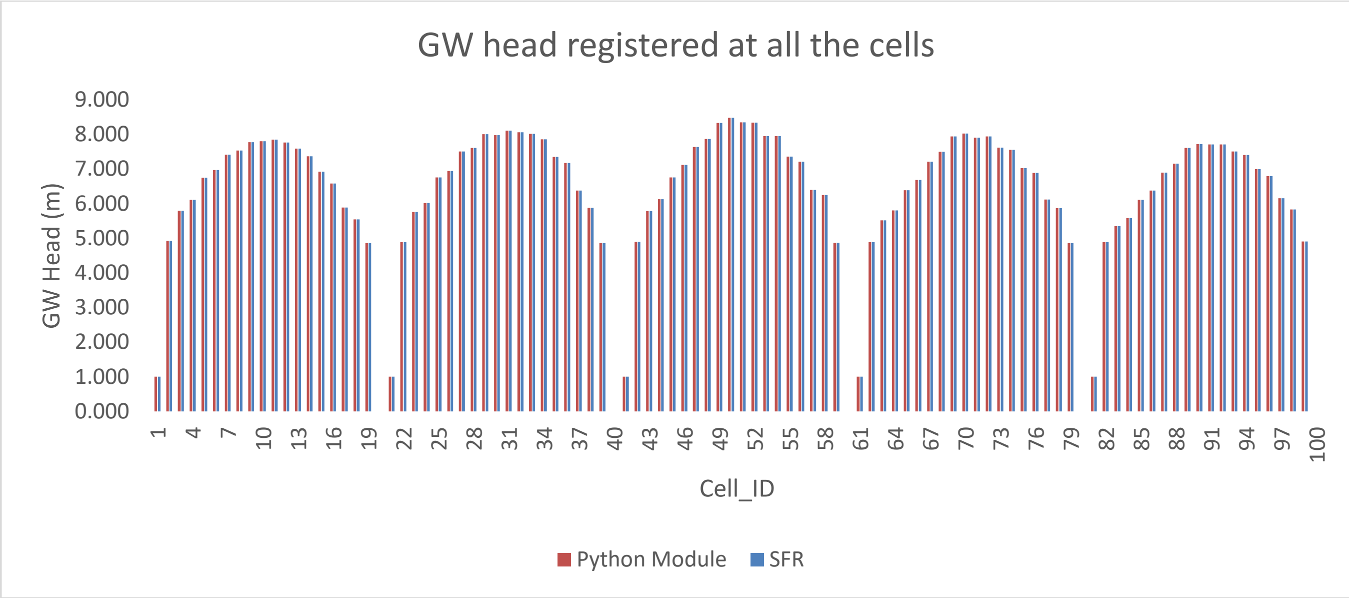

Figure 39. GW head at each cell graph using the Python module and SFR

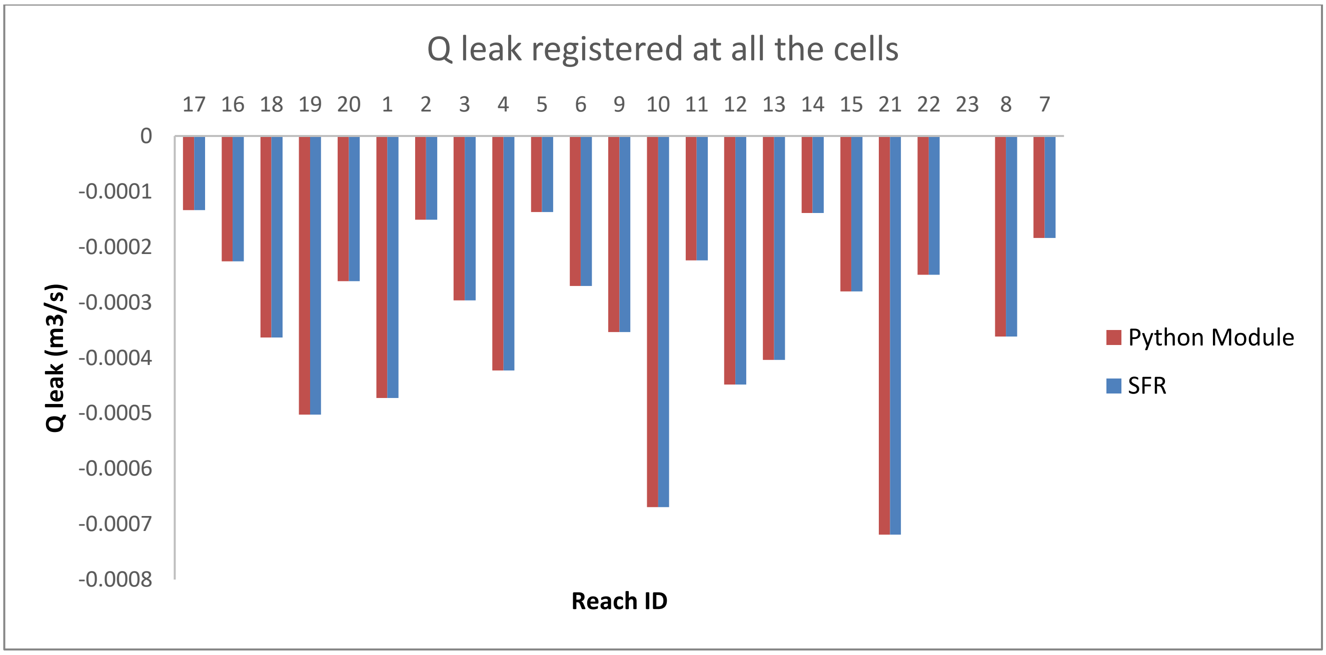

Figure 40. Q leakage at each cell graph using the Python module and SFR.

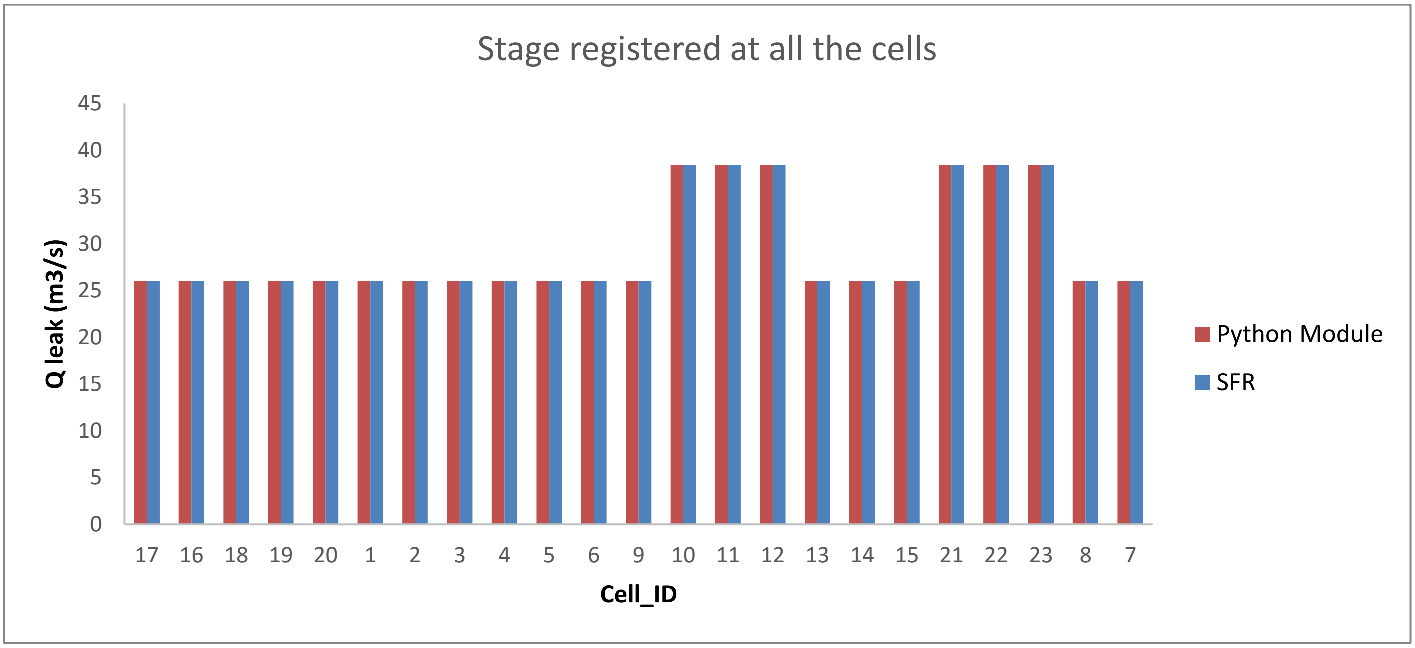

Figure 41. River stage at each cell graph using the Python module and SFR.

7 Conclusions

This document describes the development of a streamflow routing tool, that is loosely coupled to the GW software MODFLOW through the exchange of leakage and GW head information. The developed module requires further improvements, however under the current state of development it can achieve similar results to SFR module when the conductance of the reaches is not high, and the aquifer is not confined. Furthermore, a comparison between both numerical models was possible, as the equations for the calculation of the streamflow routing and the leakage terms were the same equations of the SFR package, but with a different coupling approach.

From the development perspective of the module, it was possible to conclude that more improvements in the code are required. Specifically, a function to smooth the calculation when the GW head or the river stage is close to the different reach cross-section limits, like the riverbed bottom, riverbed elevation, and the cell top elevation. Furthermore, by reviewing the SFR Fortran code it was possible to find a smoothing function when the river stage is approaching 0 m. This function was implemented in the Python module but not tested. Furthermore, a similar function might be useful for smoothing the calculation when the stage or the GW head approaches a reach cross-section limit.

After implementing a smoothing function, the next improvement must be increasing the number of stress periods that can be calculated, and expanding the number of river inflows that can be input into the River Network. This is necessary to perform a better estimation of the reaches stage through time. However, to confirm if these changes will improve the calculation, it is necessary to contrast the simulated results from the Python module with a real case scenario. Also, two proposals on how to increase the number of river inflows were made and one of them is being implemented.

Through the testing of the module, it was possible to conclude that the most sensible parameter for the loosely coupling is the conductance. Furthermore, under high conductance conditions, the assumption of a steady state will not be valid, and the loosely coupling approach will not be able to converge to a solution if there are no intermediate time steps to observe the evolution of the model. In addition, the developed module was not reliable under confined aquifer conditions, as it was not converging for all the hypothetical cases.

Another conclusion from this project was that more testing cases must be done to identify more edge cases, that might not work in the current implementation. For instance, no sensibility analysis for the variation of the cell size was made, and there was no test case when two reaches are in the same cell. In the current test cases only a river network of one reach, and a river network with one tributary and one bifurcation were tested. However, exhaustive testing of a digital tool is a time-consuming task as it requires developing several test cases with different features and different combinations of inputs, the reason why more test cases and sensibility analyses are needed.

Future work to improve the module will be exploring visualization of the river stage by assigning values to the river nodes in the VTU conversion function. Implementing other numerical schemes for the calculation of the river stage and comparing the results with the current calculation approach. Finally, developing a graphical user interface instead of inputting the paths of the files directly in the scripts for users that are not used to coding.

8 References

Bakker, Mark, Post, Vincent, Hughes, J. D., Langevin, C. D., White, J. T., Leaf, A. T., Paulinski, S. R., Bellino, J. C., Morway, E. D., Toews, M. W., Larsen, J. D., Fienen, M. N., Starn, J. J., Brakenhoff, D. A., and Bonelli, W. P., 2023, FloPy v3.5.0.dev0 (preliminary): U.S. Geological Survey Software Release, 13 July 2023, https://doi.org/10.5066/F7BK19FH

Bakker, Mark, Post, Vincent, Langevin, C. D., Hughes, J. D., White, J. T., Starn, J. J. and Fienen, M. N., 2016, Scripting MODFLOW Model Development Using Python and FloPy: Groundwater, v. 54, p. 733–739, doi:10.1111/gwat.12413.

BGR. (2022). BGR Report. Available: https://www.bgr.bund.de/EN/Gemeinsames/Produkte/Downloads/Report2021_en.pdf?__blob=publicationFile&v=2

Fenske, J., Banta, E., Piper, S., Donchyts, G., & Wexler, E. J. (2011). Coupling HEC-RAS and MODFLOW using OpenMI.

Gammons, C. H., Duaime, T. E., Parker, S. R., Poulson, S. R., & Kennelly, P. (2010). Geochemistry and stable isotope investigation of acid mine drainage associated with abandoned coal mines in central Montana, USA. Rates of Geochemical Processes and their Application to Natural Systems, 269(1), 100-112. https://doi.org/10.1016/j.chemgeo.2009.05.026

Gleeson, T., & Paszkowski, D. (2014). Perceptions of scale in hydrology: What do you mean by regional scale? Hydrological Sciences Journal, 59(1), 99-107. https://doi.org/10.1080/02626667.2013.797581

Haque, Salama, A., Lo, K., & Wu, P. (2021). Surface and Groundwater Interactions: A Review of (Langevin et al., 2017) Coupling Strategies in Detailed Domain Models. Hydrology, 8(1), 35–. https://doi.org/10.3390/hydrology8010035

Houben, G., Broda, S. (n.d) Groundwater resources in Germany. BGR. Available: https://www.deutsche-rohstoffagentur.de/EN/Themen/Wasser/grundwasser_deutschland_en.html

Hughes, J.D., Langevin, C.D., Paulinski, S.R., Larsen, J.D. and Brakenhoff, D. (2023), FloPy Workflows for Creating Structured and Unstructured MODFLOW Models. Groundwater. https://doi.org/10.1111/gwat.13327

Khan, H. H., & Khan, A. (2019). Chapter 14—Groundwater and Surface Water Interaction. En S. Venkatramanan, M. V. Prasanna, & S. Y. Chung (Eds.), GIS and Geostatistical Techniques for Groundwater Science (pp. 197-207). Elsevier. https://doi.org/10.1016/B978-0-12-815413-7.00014-6

Langevin, C. D., Hughes, J. D., Banta, E. R., Niswonger, R. G., Panday, S., & Provost, A. M. (2017). Documentation for the MODFLOW 6 Groundwater Flow Model (Report N.o 6-A55; Techniques and Methods). USGS Publications Warehouse. https://doi.org/10.3133/tm6A55

Leaf, A.T., Fienen, M.N. and Reeves, H.W. (2021), SFRmaker and Linesink-Maker: Rapid Construction of Streamflow Routing Networks from Hydrography Data. Groundwater, 59: 761-771. https://doi.org/10.1111/gwat.13095

Leaf, A.T., Fienen, M.N. and Reeves, H.W., 2021, SFRmaker version 0.7.1: U.S. Geological Survey Software Release, 29 Jan. 2021, https://doi.org/10.5066/P9U2T031

Marlow, F., Sahlbach, T., Brückner, F., Mansel, H., & Riedel, J. (2016). Gekoppeltes Grundwasser-Oberflächenwasser- Modell im Bundesprogramm Biologische Vielfalt „Lebendige Luppe" (I. für W. und technische H. Technische Universität Dresden, Ed.; Número 57, pp. 227-235). Technische Universität Dresden, Institut für Wasserbau und technische Hydromechanik. https://hdl.handle.net/20.500.11970/103328

Rodriguez, L. B., Cello, P. A., Vionnet, C. A., & Goodrich, D. (2008). Fully conservative coupling of HEC-RAS with MODFLOW to simulate stream–aquifer interactions in a drainage basin. Journal of Hydrology, 353(1), 129-142. https://doi.org/10.1016/j.jhydrol.2008.02.002

Ruf, W., Perona, P., Molnar, P., & Burlando, P. (2006). Kopplung eines hydrodynamischen Strömungsmodells und eines Grundwassermodells (I. für W. und technische H. Technische Universität Dresden, Ed.; Número 32, pp. 261-276). Technische Universität Dresden, Institut für Wasserbau und technische Hydromechanik. https://hdl.handle.net/20.500.11970/103825

Scantlebury, L. (2021). Pragmatic Groundwater-Surface Water Model Coupling with Unstructured Grids. https://doi.org/10.13140/RG.2.2.33716.17288

Sullivan and Kaszynski, (2019). PyVista: 3D plotting and mesh analysis through a streamlined interface for the Visualization Toolkit (VTK). Journal of Open Source Software, 4(37), 1450, https://doi.org/10.21105/joss.01450

Swain, E. D., & Wexler, E. J. (1996). A coupled surface-water and ground-water flow model (MODBRANCH) for simulation of stream-aquifer interaction (Report N.o 06-A6; Techniques of Water-Resources Investigations). USGS Publications Warehouse. https://doi.org/10.3133/twri06A6

Tu, Yeoh, G. H., & Liu, C. (2018). Computational Fluid Dynamics - A Practical Approach (3rd Edition).

USGS. (2017). MODFLOW 6 – Description of Input and Output. Available at: https://water.usgs.gov/ogw/modflow/mf6io.pdf

Winter, T. C., Harvey, J. W., Franke, O. L., & Alley, W. M. (1998). Ground water and surface water: A single resource (Report N.o 1139; Circular). USGS Publications Warehouse. https://doi.org/10.3133/cir1139

9 Appendix 1 Budget file

Flow Type (TEXT)	Method Code (IMETH)	Description
FLOW-JA-FACE	1	intercell flow; array of size(NJA)
STO-SS	1	confined storage; array of size (NCELLS)
STO-SY	1	unconfined storage; array of size (NCELLS)
CHD	6	constant head flow
WEL	6	well flow
WEL-TO-MVR	6	well flow that is routed to Mover Package
DRN	6	drain flow
DRN-TO-MVR	6	drain flow that is routed to Mover Package
RIV	6	river leakage
RIV-TO-MVR	6	river leakage that is routed to Mover Package
GHB	6	general-head boundary flow
GHB-TO-MVR	6	general-head boundary flow that is routed to Mover Package
RCH	6	recharge flow
EVT	6	evapotranspiration flow
MAW	6	multi-aquifer well flow; ID2 contains the well number
LAK	6	lake leakage; ID2 contains the lake number
SFR	6	stream leakage; ID2 contains the stream reach number
UZF-GWRCH	6	water table recharge from UZF Package
UZF-GWET	6	water table evapotranspiration from UZF Package
UZF-GWD	6	groundwater discharge to land surface from UZF Package
UZF-GWD-TO-MVR	6	groundwater discharge to land surface from UZF Package that is routed to Mover Package
FLOW-JA-FACE	6	flow to or from a cell in another GWF Model; TXT1ID1 is the name of the GWF Model described by this budget file, TXT2ID1 is the name of the GWF-GWF Exchange, TXT1ID2 is the name of the connected GWF Model, TXT2ID2 is the name of the GWF-GWF Exchange, and ID2 is the cell or node number of the cell in the connected model

Table 23. Record names that might be stored in the GW model budget file (USGS, 2017)

Flow term	IMETH	NDAT / NLIST	Description
FLOW-JA-FACE	6	2 / $\sum_{n=1}^{\mathrm{maxbound}} \mathrm{nconn}_n$	Connection flow from reach (ID1) to unmanaged and managed (tributaries) connections (ID2). The cross-sectional flow area (FLOW-AREA) is saved as an auxiliary data item for this flow term.
GWF	6	2 / maxbound	Calculated flow from reach (ID1) to GWF cell (ID2). The reach-aquifer flow area (FLOW-AREA) is saved as an auxiliary data item for this flow term.
EXT-INFLOW	6	1 / maxbound	Specified inflow to reach. The reach number is written to (ID1) and (ID2).
RUNOFF	6	1 / maxbound	Specified runoff to reach. The reach number is written to (ID1) and (ID2).
RAIN	6	1 / maxbound	Specified rainfall on reach. The reach number is written to (ID1) and (ID2).
EVAPORATION	6	1 / maxbound	Calculated evaporation from reach. The reach number is written to (ID1) and (ID2).
EXT-OUTFLOW	6	1 / maxbound	Calculated outflow to external boundaries (is nonzero for reaches with no downstream connections). The reach number is written to (ID1) and (ID2).
FROM-MVR	6	1 / maxbound	Calculated flow to reach from the MVR Package. Only saved if MVR Package is used in the SFR Package. The reach number is written to (ID1) and (ID2).
TO-MVR	6	1 / maxbound	Calculated flow from reach to the MVR Package. Only saved if MVR Package is used in the SFR Package. The reach number is written to (ID1) and (ID2).
AUXILIARY	6	naux+1 / maxbound	Auxiliary variables, if specified in the SFR Package, are saved to this flow term. The first entry of the DATA2D column has a value of zero. The reach number is written to (ID1) and (ID2).

Table 24. Record names that might be stored in the SFR model budget file (USGS, 2017)

10 Appendix 2 Results from Simple case

10.1 SFR results in simple case GW level close to riverbed bottom.

cell_id	reach_id	q leak time 1 (m3/s)	q leak time 2 (m3/s)	q leak time 3 (m3/s)	q leak time 4 (m3/s)	q leak time 5 (m3/s)	q leak time 6 (m3/s)	q leak time 7 (m3/s)	q leak time 8 (m3/s)	q leak time 9 (m3/s)	q leak time 10 (m3/s)
0	No_reach	0	0	0	0	0	0	0	0	0	0
1	1	-0.0154	-0.0154	-0.0154	-0.0154	-0.0154	-0.0154	-0.0154	-0.0140	-0.0122	-0.0106
2	2	-0.0051	-0.0051	-0.0051	-0.0051	-0.0051	-0.0051	-0.0051	-0.0051	-0.0051	-0.0051
3	3	-0.0102	-0.0102	-0.0102	-0.0102	-0.0102	-0.0102	-0.0102	-0.0102	-0.0102	-0.0102
4	No_reach	0	0	0	0	0	0	0	0	0	0
5	4	0	0	0	0	0	0	0	0	0	0

Table 25. Leakage results for all the cells from time 1 to 10 using SFR

cell_id	reach_id	h stage time 1 (m)	h stage time 2 (m)	h stage time 3 (m)	h stage time 4 (m)	h stage time 5 (m)	h stage time 6 (m)	h stage time 7 (m)	h stage time 8 (m)	h stage time 9 (m)	h stage time 10 (m)
0	No_reach	0	0	0	0	0	0	0	0	0	0
1	1	2.874	2.874	2.874	2.874	2.874	2.874	2.874	2.874	2.874	2.874
2	2	2.874	2.874	2.874	2.874	2.874	2.874	2.874	2.874	2.874	2.874
3	3	2.873	2.873	2.873	2.873	2.873	2.873	2.873	2.873	2.874	2.874
4	No_reach	0	0	0	0	0	0	0	0	0	0
5	4	2.873	2.873	2.873	2.873	2.873	2.873	2.873	2.873	2.873	2.873

Table 26. Stage results for all the cells from time 1 to 10 using SFR

cell_id	reach_id	gw head time 1 (m)	gw head time 2 (m)	gw head time 3 (m)	gw head time 4 (m)	gw head time 5 (m)	gw head time 6 (m)	gw head time 7 (m)	gw head time 8 (m)	gw head time 9 (m)	gw head time 10 (m)
0	No_reach	0	0	0	0	0	0	0	0	0	0
1	1	0.205	0.410	0.615	0.819	1.024	1.229	1.434	1.621	1.784	1.925
2	2	0.068	0.137	0.205	0.273	0.341	0.410	0.478	0.546	0.614	0.683
3	3	0.136	0.273	0.409	0.546	0.682	0.819	0.955	1.092	1.228	1.365
4	No_reach	0	0	0	0	0	0	0	0	0	0
5	4	0	0	0	0	0	0	0	0	0	0

Table 27. GW head results for all the cells from time 1 to 10 using SFR

10.2 Developed module results simple case GW level close to riverbed bottom.

cell_id	reach_id	h stage time 1.0 (m)	h stage time 2.0 (m)	h stage time 3.0 (m)	h stage time 4.0 (m)	h stage time 5.0 (m)	h stage time 6.0 (m)	h stage time 7.0 (m)	h stage time 8.0 (m)	h stage time 9.0 (m)	h stage time 10.0 (m)
0	No_reach	0	0	0	0	0	0	0	0	0	0
1	1	2.874	2.874	2.874	2.874	2.874	2.874	2.874	2.874	2.874	2.874
2	2	2.874	2.874	2.874	2.874	2.874	2.874	2.874	2.874	2.874	2.874
3	3	2.873	2.873	2.873	2.873	2.873	2.873	2.873	2.873	2.873	2.873
4	No_reach	0	0	0	0	0	0	0	0	0	0
5	4	2.872	2.872	2.872	2.872	2.872	2.872	2.872	2.872	2.872	2.873

Table 28. Stage results for all the cells from time 1 to 10 using Python module

cell_id	reach_id	q leak time 1.0 (m3/s)	q leak time 2.0 (m3/s)	q leak time 3.0 (m3/s)	q leak time 4.0 (m3/s)	q leak time 5.0 (m3/s)	q leak time 6.0 (m3/s)	q leak time 7.0 (m3/s)	q leak time 8.0 (m3/s)	q leak time 9.0 (m3/s)	q leak time 10.0 (m3/s)
0	No_reach	0	0	0	0	0	0	0	0	0	0
1	1	-0.0154	-0.0154	-0.0154	-0.0154	-0.0154	-0.0154	-0.0154	-0.0138	-0.0115	-0.0092
2	2	-0.0051	-0.0051	-0.0051	-0.0051	-0.0051	-0.0051	-0.0051	-0.0051	-0.0051	-0.0051
3	3	-0.0102	-0.0102	-0.0102	-0.0102	-0.0102	-0.0102	-0.0102	-0.0102	-0.0102	-0.0102
4	No_reach	0	0	0	0	0	0	0	0	0	0
5	4	0	0	0	0	0	0	0	0	0	0

Table 29. Leak results for all the cells from time 1 to 10 using Python module

cell_id	reach_id	gw head time 1.0 (m)	gw head time 2.0 (m)	gw head time 3.0 (m)	gw head time 4.0 (m)	gw head time 5.0 (m)	gw head time 6.0 (m)	gw head time 7.0 (m)	gw head time 8.0 (m)	gw head time 9.0 (m)	gw head time 10.0 (m)
0	No_reach	0	0	0	0	0	0	0	0	0	0
1	1	0.205	0.410	0.614	0.819	1.024	1.229	1.434	1.638	1.843	2.048
2	2	0.068	0.137	0.205	0.273	0.341	0.410	0.478	0.546	0.614	0.683
3	3	0.136	0.273	0.409	0.546	0.682	0.819	0.955	1.092	1.228	1.365
4	No_reach	0	0	0	0	0	0	0	0	0	0
5	4	0	0	0	0	0	0	0	0	0	0

Table 30. GW head results for all the cells from time 1 to 10 using Python module

11 Appendix 3 Results from the bifurcation and tributary case

11.1 Base case results

Python					SFR				
cell_id	reach_id	gw head time 1.0 (m)	h stage time 1.0 (m)	q leak time 1.0 (m3/s)	cell_id	reach_id	gw head time 1.0 (m)	h stage time 1.0 (m)	q leak time 1.0 (m3/s)
1	No_reach	1.000	0.000	0.000000	1	No_reach	1.000	0.000	0.000000
2	No_reach	4.922	0.000	0.000000	2	No_reach	4.922	0.000	0.000000
3	No_reach	4.999	0.000	0.000000	3	No_reach	4.999	0.000	0.000000
4	No_reach	5.005	0.000	0.000000	4	No_reach	5.005	0.000	0.000000
5	No_reach	5.015	0.000	0.000000	5	No_reach	5.015	0.000	0.000000
6	No_reach	5.022	0.000	0.000000	6	No_reach	5.023	0.000	0.000000
7	No_reach	5.037	0.000	0.000000	7	No_reach	5.037	0.000	0.000000
8	No_reach	5.040	0.000	0.000000	8	No_reach	5.041	0.000	0.000000
9	No_reach	5.047	0.000	0.000000	9	No_reach	5.047	0.000	0.000000
10	No_reach	5.035	0.000	0.000000	10	No_reach	5.036	0.000	0.000000
11	No_reach	5.013	0.000	0.000000	11	No_reach	5.013	0.000	0.000000
12	No_reach	5.008	0.000	0.000000	12	No_reach	5.008	0.000	0.000000
13	No_reach	5.000	0.000	0.000000	13	No_reach	5.000	0.000	0.000000
14	No_reach	5.000	0.000	0.000000	14	No_reach	5.000	0.000	0.000000
15	No_reach	5.000	0.000	0.000000	15	No_reach	5.000	0.000	0.000000
16	No_reach	5.000	0.000	0.000000	16	No_reach	5.000	0.000	0.000000
17	No_reach	5.000	0.000	0.000000	17	No_reach	5.000	0.000	0.000000
18	No_reach	4.999	0.000	0.000000	18	No_reach	4.999	0.000	0.000000
19	No_reach	4.856	0.000	0.000000	19	No_reach	4.856	0.000	0.000000
20	No_reach	0.000	0.000	0.000000	20	No_reach	0.000	0.000	0.000000
21	No_reach	1.000	0.000	0.000000	21	No_reach	1.000	0.000	0.000000
22	No_reach	4.885	0.000	0.000000	22	No_reach	4.885	0.000	0.000000
23	No_reach	4.999	0.000	0.000000	23	No_reach	4.999	0.000	0.000000
24	No_reach	5.000	0.000	0.000000	24	No_reach	5.000	0.000	0.000000
25	No_reach	5.003	0.000	0.000000	25	No_reach	5.003	0.000	0.000000
26	No_reach	5.011	0.000	0.000000	26	No_reach	5.011	0.000	0.000000
27	No_reach	5.024	0.000	0.000000	27	No_reach	5.024	0.000	0.000000
28	No_reach	5.045	0.000	0.000000	28	No_reach	5.045	0.000	0.000000
29	No_reach	5.095	0.000	0.000000	29	No_reach	5.096	0.000	0.000000
30	No_reach	5.077	0.000	0.000000	30	No_reach	5.077	0.000	0.000000
31	No_reach	5.069	0.000	0.000000	31	No_reach	5.070	0.000	0.000000
32	17	4.998	4.297	0.000261	32	17	4.998	4.297	0.000261
33	16	4.994	4.297	0.000437	33	16	4.994	4.297	0.000437
34	18	4.991	4.297	0.000694	34	18	4.991	4.297	0.000694
35	No_reach	5.000	0.000	0.000000	35	No_reach	5.000	0.000	0.000000
36	19	4.988	4.297	0.000921	36	19	4.988	4.297	0.000921
37	20	4.995	4.297	0.000465	37	20	4.995	4.297	0.000465
38	No_reach	4.999	0.000	0.000000	38	No_reach	4.999	0.000	0.000000
39	No_reach	4.856	0.000	0.000000	39	No_reach	4.856	0.000	0.000000
40	No_reach	0.000	0.000	0.000000	40	No_reach	0.000	0.000	0.000000
41	No_reach	1.000	0.000	0.000000	41	No_reach	1.000	0.000	0.000000

42	1	4.876	4.297	0.000647	42	1	4.876	4.297	0.000647
43	2	4.995	4.297	0.000260	43	2	4.995	4.297	0.000260
44	3	4.993	4.297	0.000519	44	3	4.993	4.297	0.000519
45	No_reach	5.000	0.000	0.000000	45	No_reach	5.000	0.000	0.000000
46	4	4.990	4.297	0.000774	46	4	4.990	4.297	0.000774
47	5	4.996	4.297	0.000260	47	5	4.996	4.297	0.000260
48	6	4.993	4.297	0.000519	48	6	4.993	4.297	0.000519
49	9	4.995	4.297	0.000697	49	9	4.995	4.297	0.000697
50	10	5.183	5.482	-0.000335	50	10	5.184	5.482	-0.000333
51	11	5.205	5.482	-0.000103	51	11	5.207	5.482	-0.000103
52	12	5.203	5.482	-0.000208	52	12	5.205	5.482	-0.000207
53	No_reach	5.059	0.000	0.000000	53	No_reach	5.060	0.000	0.000000
54	13	4.991	4.298	0.000775	54	13	4.991	4.298	0.000775
55	14	4.997	4.298	0.000261	55	14	4.997	4.298	0.000261
56	15	4.994	4.298	0.000519	56	15	4.994	4.298	0.000519
57	No_reach	5.062	0.000	0.000000	57	No_reach	5.062	0.000	0.000000
58	21	5.127	5.482	-0.000397	58	21	5.127	5.482	-0.000397
59	22	4.860	5.482	-0.000232	59	22	4.860	5.482	-0.000232
60	23	0.000	5.482	0.000000	60	23	0.000	5.482	0.000000
61	No_reach	1.000	0.000	0.000000	61	No_reach	1.000	0.000	0.000000
62	No_reach	4.885	0.000	0.000000	62	No_reach	4.885	0.000	0.000000
63	No_reach	4.999	0.000	0.000000	63	No_reach	4.999	0.000	0.000000
64	No_reach	5.000	0.000	0.000000	64	No_reach	5.000	0.000	0.000000
65	No_reach	5.000	0.000	0.000000	65	No_reach	5.000	0.000	0.000000
66	No_reach	5.000	0.000	0.000000	66	No_reach	5.000	0.000	0.000000
67	No_reach	5.000	0.000	0.000000	67	No_reach	5.000	0.000	0.000000
68	No_reach	5.000	0.000	0.000000	68	No_reach	5.000	0.000	0.000000
69	8	4.991	4.297	0.000694	69	8	4.991	4.297	0.000694
70	No_reach	5.009	0.000	0.000000	70	No_reach	5.010	0.000	0.000000
71	No_reach	5.060	0.000	0.000000	71	No_reach	5.061	0.000	0.000000
72	No_reach	5.097	0.000	0.000000	72	No_reach	5.098	0.000	0.000000
73	No_reach	5.061	0.000	0.000000	73	No_reach	5.061	0.000	0.000000
74	No_reach	5.051	0.000	0.000000	74	No_reach	5.052	0.000	0.000000
75	No_reach	5.025	0.000	0.000000	75	No_reach	5.025	0.000	0.000000
76	No_reach	5.012	0.000	0.000000	76	No_reach	5.012	0.000	0.000000
77	No_reach	5.002	0.000	0.000000	77	No_reach	5.002	0.000	0.000000
78	No_reach	4.999	0.000	0.000000	78	No_reach	4.999	0.000	0.000000
79	No_reach	4.856	0.000	0.000000	79	No_reach	4.856	0.000	0.000000
80	No_reach	0.000	0.000	0.000000	80	No_reach	0.000	0.000	0.000000
81	No_reach	1.000	0.000	0.000000	81	No_reach	1.000	0.000	0.000000
82	No_reach	4.885	0.000	0.000000	82	No_reach	4.885	0.000	0.000000
83	No_reach	4.999	0.000	0.000000	83	No_reach	4.999	0.000	0.000000
84	No_reach	5.000	0.000	0.000000	84	No_reach	5.000	0.000	0.000000
85	No_reach	5.000	0.000	0.000000	85	No_reach	5.000	0.000	0.000000
86	No_reach	5.000	0.000	0.000000	86	No_reach	5.000	0.000	0.000000
87	No_reach	5.000	0.000	0.000000	87	No_reach	5.000	0.000	0.000000
88	No_reach	5.000	0.000	0.000000	88	No_reach	5.000	0.000	0.000000
89	7	4.996	4.297	0.000349	89	7	4.996	4.297	0.000349
90	No_reach	5.002	0.000	0.000000	90	No_reach	5.002	0.000	0.000000
91	No_reach	5.027	0.000	0.000000	91	No_reach	5.027	0.000	0.000000
92	No_reach	5.039	0.000	0.000000	92	No_reach	5.039	0.000	0.000000

93	No_reach	5.042	0.000	0.000000	93	No_reach	5.043	0.000	0.000000
94	No_reach	5.044	0.000	0.000000	94	No_reach	5.044	0.000	0.000000
95	No_reach	5.030	0.000	0.000000	95	No_reach	5.030	0.000	0.000000
96	No_reach	5.023	0.000	0.000000	96	No_reach	5.023	0.000	0.000000
97	No_reach	5.007	0.000	0.000000	97	No_reach	5.007	0.000	0.000000
98	No_reach	4.999	0.000	0.000000	98	No_reach	4.999	0.000	0.000000
99	No_reach	4.903	0.000	0.000000	99	No_reach	4.903	0.000	0.000000
100	No_reach	0.000	0.000	0.000000	100	No_reach	0.000	0.000	0.000000

Table 31. Base case results from SFR and Python module with a 1 second duration

11.2 Riverbed conductivity 0.001 m/s with 5m head at initial conditions

Python					SFR				
cell_id	reach_id	gw head time 1.0 (m)	h stage time 1.0 (m)	q leak time 1.0 (m3/s)	cell_id	reach_id	gw head time 1 (m)	h stage time 1 (m)	q leak time 1 (m3/s)
1	No_reach	1.000	0.000	0.000	1	No_reach	1.000	0.000	0.00000
2	No_reach	31.626	0.000	0.000	2	No_reach	4.922	0.000	0.00000
3	No_reach	92.878	0.000	0.000	3	No_reach	4.999	0.000	0.00000
4	No_reach	115.868	0.000	0.000	4	No_reach	5.008	0.000	0.00000
5	No_reach	161.850	0.000	0.000	5	No_reach	5.024	0.000	0.00000
6	No_reach	177.892	0.000	0.000	6	No_reach	5.038	0.000	0.00000
7	No_reach	209.975	0.000	0.000	7	No_reach	5.066	0.000	0.00000
8	No_reach	218.917	0.000	0.000	8	No_reach	5.074	0.000	0.00000
9	No_reach	236.800	0.000	0.000	9	No_reach	5.088	0.000	0.00000
10	No_reach	238.717	0.000	0.000	10	No_reach	5.063	0.000	0.00000
11	No_reach	242.552	0.000	0.000	11	No_reach	5.012	0.000	0.00000
12	No_reach	236.475	0.000	0.000	12	No_reach	5.008	0.000	0.00000
13	No_reach	224.321	0.000	0.000	13	No_reach	4.999	0.000	0.00000
14	No_reach	207.802	0.000	0.000	14	No_reach	5.000	0.000	0.00000
15	No_reach	174.763	0.000	0.000	15	No_reach	4.999	0.000	0.00000
16	No_reach	148.823	0.000	0.000	16	No_reach	5.000	0.000	0.00000
17	No_reach	96.944	0.000	0.000	17	No_reach	5.000	0.000	0.00000
18	No_reach	70.505	0.000	0.000	18	No_reach	4.999	0.000	0.00000
19	No_reach	17.626	0.000	0.000	19	No_reach	4.856	0.000	0.00000
20	No_reach	0.000	0.000	0.000	20	No_reach	0.000	0.000	0.00000
21	No_reach	1.000	0.000	0.000	21	No_reach	1.000	0.000	0.00000
22	No_reach	22.952	0.000	0.000	22	No_reach	4.885	0.000	0.00000
23	No_reach	88.808	0.000	0.000	23	No_reach	4.998	0.000	0.00000
24	No_reach	108.148	0.000	0.000	24	No_reach	5.000	0.000	0.00000
25	No_reach	162.099	0.000	0.000	25	No_reach	4.999	0.000	0.00000
26	No_reach	175.747	0.000	0.000	26	No_reach	5.013	0.000	0.00000
27	No_reach	216.941	0.000	0.000	27	No_reach	5.028	0.000	0.00000
28	No_reach	224.176	0.000	0.000	28	No_reach	5.080	0.000	0.00000
29	No_reach	252.846	0.000	0.000	29	No_reach	5.198	0.000	0.00000
30	No_reach	250.848	0.000	0.000	30	No_reach	5.153	0.000	0.00000
31	No_reach	260.902	0.000	0.000	31	No_reach	5.129	0.000	0.00000
32	17	258.541	4.366	0.947	32	17	4.970	4.298	0.00251
33	16	255.915	4.341	1.579	33	16	4.945	4.297	0.00406
34	18	245.206	4.431	2.408	34	18	4.919	4.298	0.00622
35	No_reach	207.396	0.000	0.000	35	No_reach	4.997	0.000	0.00000

36	19	193.603	4.497	2.521	36	19	4.897	4.298	0.00799
37	20	135.173	4.520	0.871	37	20	4.947	4.298	0.00433
38	No_reach	97.944	0.000	0.000	38	No_reach	4.998	0.000	0.00000
39	No_reach	24.486	0.000	0.000	39	No_reach	4.856	0.000	0.00000
40	No_reach	0.000	0.000	0.000	40	No_reach	0.000	0.000	0.00000
41	No_reach	1.000	0.000	0.000	41	No_reach	1.000	0.000	0.00000
42	1	34.297	4.307	0.335	42	1	4.811	4.297	0.00574
43	2	92.501	4.316	0.329	43	2	4.965	4.298	0.00249
44	3	115.984	4.338	0.832	44	3	4.938	4.298	0.00477
45	No_reach	162.344	0.000	0.000	45	No_reach	4.998	0.000	0.00000
46	4	188.753	4.394	2.061	46	4	4.912	4.298	0.00686
47	5	226.564	4.416	0.828	47	5	4.966	4.298	0.00249
48	6	243.666	4.463	1.783	48	6	4.938	4.298	0.00477
49	9	276.828	4.466	2.724	49	9	4.927	4.298	0.00629
50	10	285.510	5.798	3.127	50	10	5.406	5.483	-0.00086
51	11	276.250	5.818	1.008	51	11	5.428	5.483	-0.00020
52	12	275.678	5.857	2.011	52	12	5.421	5.483	-0.00046
53	No_reach	249.795	0.000	0.000	53	No_reach	5.083	0.000	0.00000
54	13	251.307	4.844	2.756	54	13	4.913	4.299	0.00687
55	14	208.152	4.862	0.758	55	14	4.966	4.299	0.00249
56	15	197.170	4.896	1.433	56	15	4.940	4.299	0.00478
57	No_reach	137.273	0.000	0.000	57	No_reach	5.185	0.000	0.00000
58	21	126.362	6.136	1.344	58	21	5.392	5.484	-0.00102
59	22	36.524	6.138	0.113	59	22	4.888	5.484	-0.00222
60	23	0.000	6.137	0.000	60	23	0.000	5.484	0.00000
61	No_reach	1.000	0.000	0.000	61	No_reach	1.000	0.000	0.00000
62	No_reach	17.450	0.000	0.000	62	No_reach	4.885	0.000	0.00000
63	No_reach	66.799	0.000	0.000	63	No_reach	4.999	0.000	0.00000
64	No_reach	89.842	0.000	0.000	64	No_reach	5.000	0.000	0.00000
65	No_reach	133.269	0.000	0.000	65	No_reach	5.000	0.000	0.00000
66	No_reach	155.884	0.000	0.000	66	No_reach	5.000	0.000	0.00000
67	No_reach	194.655	0.000	0.000	67	No_reach	5.000	0.000	0.00000
68	No_reach	216.286	0.000	0.000	68	No_reach	4.999	0.000	0.00000
69	8	249.269	4.394	2.449	69	8	4.920	4.298	0.00623
70	No_reach	255.353	0.000	0.000	70	No_reach	4.999	0.000	0.00000
71	No_reach	246.047	0.000	0.000	71	No_reach	5.113	0.000	0.00000
72	No_reach	248.373	0.000	0.000	72	No_reach	5.191	0.000	0.00000
73	No_reach	225.146	0.000	0.000	73	No_reach	5.109	0.000	0.00000
74	No_reach	220.886	0.000	0.000	74	No_reach	5.085	0.000	0.00000
75	No_reach	183.458	0.000	0.000	75	No_reach	5.037	0.000	0.00000
76	No_reach	173.107	0.000	0.000	76	No_reach	5.013	0.000	0.00000
77	No_reach	117.359	0.000	0.000	77	No_reach	5.011	0.000	0.00000
78	No_reach	99.198	0.000	0.000	78	No_reach	5.016	0.000	0.00000
79	No_reach	24.799	0.000	0.000	79	No_reach	4.856	0.000	0.00000
80	No_reach	0.000	0.000	0.000	80	No_reach	0.000	0.000	0.00000
81	No_reach	1.000	0.000	0.000	81	No_reach	1.000	0.000	0.00000
82	No_reach	13.557	0.000	0.000	82	No_reach	4.885	0.000	0.00000
83	No_reach	51.228	0.000	0.000	83	No_reach	4.999	0.000	0.00000
84	No_reach	70.063	0.000	0.000	84	No_reach	5.000	0.000	0.00000
85	No_reach	110.998	0.000	0.000	85	No_reach	5.000	0.000	0.00000
86	No_reach	131.466	0.000	0.000	86	No_reach	5.000	0.000	0.00000

87	No_reach	170.598	0.000	0.000	87	No_reach	5.000	0.000	0.00000
88	No_reach	190.164	0.000	0.000	88	No_reach	5.000	0.000	0.00000
89	7	224.804	4.328	1.102	89	7	4.957	4.297	0.00330
90	No_reach	232.801	0.000	0.000	90	No_reach	4.999	0.000	0.00000
91	No_reach	232.326	0.000	0.000	91	No_reach	5.046	0.000	0.00000
92	No_reach	232.089	0.000	0.000	92	No_reach	5.070	0.000	0.00000
93	No_reach	217.656	0.000	0.000	93	No_reach	5.075	0.000	0.00000
94	No_reach	210.440	0.000	0.000	94	No_reach	5.078	0.000	0.00000
95	No_reach	181.301	0.000	0.000	95	No_reach	5.051	0.000	0.00000
96	No_reach	166.732	0.000	0.000	96	No_reach	5.038	0.000	0.00000
97	No_reach	120.867	0.000	0.000	97	No_reach	5.012	0.000	0.00000
98	No_reach	97.934	0.000	0.000	98	No_reach	4.999	0.000	0.00000
99	No_reach	32.645	0.000	0.000	99	No_reach	4.903	0.000	0.00000
100	No_reach	0.000	0.000	0.000	100	No_reach	0.000	0.000	0.00000

Table 32. Riverbed conductivity 0.001 m/s with 5m head at initial condition case results from SFR and Python module with a 1 second duration

11.3 Riverbed conductivity 0.001 m/s with 2m head at initial conditions

Python					SFR				
cell_id	reach_id	gw head time 1.0 (m)	h stage time 1.0 (m)	q leak time 1.0 (m3/s)	cell_id	reach_id	gw head time 1.0 (m)	h stage time 1.0 (m)	q leak time 1.0 (m3/s)
1	No_reach	1.000	0.000	0.0000	1	No_reach	1.000	0.000	0.0000
2	No_reach	1.981	0.000	0.0000	2	No_reach	1.981	0.000	0.0000
3	No_reach	2.000	0.000	0.0000	3	No_reach	2.000	0.000	0.0000
4	No_reach	2.000	0.000	0.0000	4	No_reach	2.000	0.000	0.0000
5	No_reach	2.000	0.000	0.0000	5	No_reach	2.000	0.000	0.0000
6	No_reach	2.000	0.000	0.0000	6	No_reach	2.000	0.000	0.0000
7	No_reach	2.000	0.000	0.0000	7	No_reach	2.000	0.000	0.0000
8	No_reach	2.000	0.000	0.0000	8	No_reach	2.000	0.000	0.0000
9	No_reach	2.000	0.000	0.0000	9	No_reach	2.000	0.000	0.0000
10	No_reach	2.000	0.000	0.0000	10	No_reach	2.000	0.000	0.0000
11	No_reach	2.001	0.000	0.0000	11	No_reach	2.001	0.000	0.0000
12	No_reach	2.000	0.000	0.0000	12	No_reach	2.000	0.000	0.0000
13	No_reach	2.003	0.000	0.0000	13	No_reach	2.003	0.000	0.0000
14	No_reach	2.000	0.000	0.0000	14	No_reach	2.000	0.000	0.0000
15	No_reach	2.003	0.000	0.0000	15	No_reach	2.003	0.000	0.0000
16	No_reach	2.000	0.000	0.0000	16	No_reach	2.000	0.000	0.0000
17	No_reach	2.000	0.000	0.0000	17	No_reach	2.000	0.000	0.0000
18	No_reach	1.999	0.000	0.0000	18	No_reach	1.999	0.000	0.0000
19	No_reach	1.942	0.000	0.0000	19	No_reach	1.942	0.000	0.0000
20	No_reach	0.000	0.000	0.0000	20	No_reach	0.000	0.000	0.0000
21	No_reach	1.000	0.000	0.0000	21	No_reach	1.000	0.000	0.0000
22	No_reach	1.971	0.000	0.0000	22	No_reach	1.971	0.000	0.0000
23	No_reach	2.002	0.000	0.0000	23	No_reach	2.002	0.000	0.0000
24	No_reach	2.000	0.000	0.0000	24	No_reach	2.000	0.000	0.0000
25	No_reach	2.003	0.000	0.0000	25	No_reach	2.003	0.000	0.0000
26	No_reach	2.000	0.000	0.0000	26	No_reach	2.000	0.000	0.0000
27	No_reach	2.002	0.000	0.0000	27	No_reach	2.002	0.000	0.0000
28	No_reach	2.000	0.000	0.0000	28	No_reach	2.000	0.000	0.0000

29	No_reach	2.004	0.000	0.0000	29	No_reach	2.004	0.000	0.0000
30	No_reach	2.000	0.000	0.0000	30	No_reach	2.000	0.000	0.0000
31	No_reach	2.005	0.000	0.0000	31	No_reach	2.005	0.000	0.0000
32	17	2.106	4.297	-0.0082	32	17	2.106	4.297	-0.0082
33	16	2.179	4.297	-0.0133	33	16	2.179	4.297	-0.0133
34	18	2.264	4.296	-0.0203	34	18	2.264	4.297	-0.0203
35	No_reach	2.011	0.000	0.0000	35	No_reach	2.011	0.000	0.0000
36	19	2.337	4.296	-0.0261	36	19	2.337	4.296	-0.0261
37	20	2.188	4.295	-0.0140	37	20	2.188	4.295	-0.0141
38	No_reach	2.003	0.000	0.0000	38	No_reach	2.003	0.000	0.0000
39	No_reach	1.942	0.000	0.0000	39	No_reach	1.942	0.000	0.0000
40	No_reach	0.000	0.000	0.0000	40	No_reach	0.000	0.000	0.0000
41	No_reach	1.000	0.000	0.0000	41	No_reach	1.000	0.000	0.0000
42	1	2.264	4.297	-0.0227	42	1	2.264	4.297	-0.0227
43	2	2.111	4.297	-0.0081	43	2	2.111	4.297	-0.0081
44	3	2.202	4.296	-0.0156	44	3	2.202	4.296	-0.0156
45	No_reach	2.008	0.000	0.0000	45	No_reach	2.008	0.000	0.0000
46	4	2.289	4.295	-0.0224	46	4	2.289	4.296	-0.0224
47	5	2.111	4.295	-0.0081	47	5	2.111	4.295	-0.0081
48	6	2.205	4.295	-0.0156	48	6	2.205	4.295	-0.0156
49	9	2.270	4.296	-0.0203	49	9	2.270	4.296	-0.0203
50	10	2.442	5.478	-0.0339	50	10	2.442	5.479	-0.0340
51	11	2.168	5.478	-0.0123	51	11	2.168	5.478	-0.0123
52	12	2.306	5.478	-0.0236	52	12	2.306	5.478	-0.0236
53	No_reach	2.009	0.000	0.0000	53	No_reach	2.009	0.000	0.0000
54	13	2.290	4.291	-0.0224	54	13	2.290	4.291	-0.0224
55	14	2.111	4.291	-0.0081	55	14	2.111	4.291	-0.0081
56	15	2.202	4.290	-0.0156	56	15	2.202	4.290	-0.0156
57	No_reach	2.010	0.000	0.0000	57	No_reach	2.010	0.000	0.0000
58	21	2.438	5.474	-0.0339	58	21	2.438	5.475	-0.0339
59	22	2.107	5.474	-0.0125	59	22	2.107	5.474	-0.0125
60	23	0.000	5.473	0.0000	60	23	0.000	5.474	0.0000
61	No_reach	1.000	0.000	0.0000	61	No_reach	1.000	0.000	0.0000
62	No_reach	1.971	0.000	0.0000	62	No_reach	1.971	0.000	0.0000
63	No_reach	2.000	0.000	0.0000	63	No_reach	2.000	0.000	0.0000
64	No_reach	2.001	0.000	0.0000	64	No_reach	2.001	0.000	0.0000
65	No_reach	2.000	0.000	0.0000	65	No_reach	2.000	0.000	0.0000
66	No_reach	2.000	0.000	0.0000	66	No_reach	2.000	0.000	0.0000
67	No_reach	2.000	0.000	0.0000	67	No_reach	2.000	0.000	0.0000
68	No_reach	2.004	0.000	0.0000	68	No_reach	2.004	0.000	0.0000
69	8	2.261	4.297	-0.0204	69	8	2.261	4.297	-0.0204
70	No_reach	2.008	0.000	0.0000	70	No_reach	2.008	0.000	0.0000
71	No_reach	2.000	0.000	0.0000	71	No_reach	2.000	0.000	0.0000
72	No_reach	2.002	0.000	0.0000	72	No_reach	2.002	0.000	0.0000
73	No_reach	2.000	0.000	0.0000	73	No_reach	2.000	0.000	0.0000
74	No_reach	2.000	0.000	0.0000	74	No_reach	2.000	0.000	0.0000
75	No_reach	2.000	0.000	0.0000	75	No_reach	2.000	0.000	0.0000
76	No_reach	2.001	0.000	0.0000	76	No_reach	2.001	0.000	0.0000
77	No_reach	2.000	0.000	0.0000	77	No_reach	2.000	0.000	0.0000
78	No_reach	2.000	0.000	0.0000	78	No_reach	2.000	0.000	0.0000
79	No_reach	1.942	0.000	0.0000	79	No_reach	1.942	0.000	0.0000

80	No_reach	0.000	0.000	0.0000	80	No_reach	0.000	0.000	0.0000
81	No_reach	1.000	0.000	0.0000	81	No_reach	1.000	0.000	0.0000
82	No_reach	1.971	0.000	0.0000	82	No_reach	1.971	0.000	0.0000
83	No_reach	2.000	0.000	0.0000	83	No_reach	2.000	0.000	0.0000
84	No_reach	2.000	0.000	0.0000	84	No_reach	2.000	0.000	0.0000
85	No_reach	2.000	0.000	0.0000	85	No_reach	2.000	0.000	0.0000
86	No_reach	2.000	0.000	0.0000	86	No_reach	2.000	0.000	0.0000
87	No_reach	2.000	0.000	0.0000	87	No_reach	2.000	0.000	0.0000
88	No_reach	2.001	0.000	0.0000	88	No_reach	2.001	0.000	0.0000
89	7	2.140	4.297	-0.0108	89	7	2.140	4.297	-0.0108
90	No_reach	2.005	0.000	0.0000	90	No_reach	2.005	0.000	0.0000
91	No_reach	2.000	0.000	0.0000	91	No_reach	2.000	0.000	0.0000
92	No_reach	2.000	0.000	0.0000	92	No_reach	2.000	0.000	0.0000
93	No_reach	2.000	0.000	0.0000	93	No_reach	2.000	0.000	0.0000
94	No_reach	2.000	0.000	0.0000	94	No_reach	2.000	0.000	0.0000
95	No_reach	2.000	0.000	0.0000	95	No_reach	2.000	0.000	0.0000
96	No_reach	2.000	0.000	0.0000	96	No_reach	2.000	0.000	0.0000
97	No_reach	2.000	0.000	0.0000	97	No_reach	2.000	0.000	0.0000
98	No_reach	2.000	0.000	0.0000	98	No_reach	2.000	0.000	0.0000
99	No_reach	1.961	0.000	0.0000	99	No_reach	1.961	0.000	0.0000
100	No_reach	0.000	0.000	0.0000	100	No_reach	0.000	0.000	0.0000

Table 33. Riverbed conductivity 0.001 m/s with 2m head at initial condition case results from SFR and Python module with a 1 second duration

11.4 Riverbed width 1meter

Python module					SFR				
cell_id	reach_id	gw head time 1.0 (m)	h stage time 1.0 (m)	q leak time 1.0 (m3/s)	cell_id	reach_id	gw head time 1.0 (m)	h stage time 1.0 (m)	q leak time 1.0 (m3/s)
1	No_reach	1.000	0.000	0.0000	1	No_reach	1.000	0.000	0.0000
2	No_reach	4.937	0.000	0.0000	2	No_reach	4.928	0.000	0.0000
3	No_reach	6.463	0.000	0.0000	3	No_reach	5.593	0.000	0.0000
4	No_reach	7.031	0.000	0.0000	4	No_reach	5.835	0.000	0.0000
5	No_reach	8.167	0.000	0.0000	5	No_reach	6.321	0.000	0.0000
6	No_reach	8.569	0.000	0.0000	6	No_reach	6.493	0.000	0.0000
7	No_reach	9.372	0.000	0.0000	7	No_reach	6.836	0.000	0.0000
8	No_reach	9.594	0.000	0.0000	8	No_reach	6.931	0.000	0.0000
9	No_reach	10.038	0.000	0.0000	9	No_reach	7.122	0.000	0.0000
10	No_reach	10.069	0.000	0.0000	10	No_reach	7.130	0.000	0.0000
11	No_reach	10.131	0.000	0.0000	11	No_reach	7.146	0.000	0.0000
12	No_reach	9.963	0.000	0.0000	12	No_reach	7.075	0.000	0.0000
13	No_reach	9.627	0.000	0.0000	13	No_reach	6.932	0.000	0.0000
14	No_reach	9.226	0.000	0.0000	14	No_reach	6.778	0.000	0.0000
15	No_reach	8.426	0.000	0.0000	15	No_reach	6.469	0.000	0.0000
16	No_reach	7.826	0.000	0.0000	16	No_reach	6.210	0.000	0.0000
17	No_reach	6.627	0.000	0.0000	17	No_reach	5.691	0.000	0.0000
18	No_reach	6.040	0.000	0.0000	18	No_reach	5.414	0.000	0.0000
19	No_reach	4.866	0.000	0.0000	19	No_reach	4.860	0.000	0.0000
20	No_reach	0.000	0.000	0.0000	20	No_reach	0.000	0.000	0.0000

21	No_reach	1.000	0.000	0.0000	21	No_reach	1.000	0.000	0.0000
22	No_reach	4.898	0.000	0.0000	22	No_reach	4.890	0.000	0.0000
23	No_reach	6.396	0.000	0.0000	23	No_reach	5.585	0.000	0.0000
24	No_reach	6.852	0.000	0.0000	24	No_reach	5.771	0.000	0.0000
25	No_reach	8.153	0.000	0.0000	25	No_reach	6.323	0.000	0.0000
26	No_reach	8.501	0.000	0.0000	26	No_reach	6.464	0.000	0.0000
27	No_reach	9.530	0.000	0.0000	27	No_reach	6.888	0.000	0.0000
28	No_reach	9.731	0.000	0.0000	28	No_reach	6.987	0.000	0.0000
29	No_reach	10.494	0.000	0.0000	29	No_reach	7.336	0.000	0.0000
30	No_reach	10.420	0.000	0.0000	30	No_reach	7.298	0.000	0.0000
31	No_reach	10.654	0.000	0.0000	31	No_reach	7.397	0.000	0.0000
32	17	10.530	8.034	0.0002	32	17	7.305	8.034	-0.0001
33	16	10.379	8.034	0.0003	33	16	7.208	8.034	-0.0001
34	18	10.091	8.034	0.0004	34	18	7.098	8.034	-0.0002
35	No_reach	9.169	0.000	0.0000	35	No_reach	6.795	0.000	0.0000
36	19	8.824	8.034	0.0002	36	19	6.679	8.034	-0.0004
37	20	7.455	8.034	-0.0001	37	20	6.177	8.034	-0.0002
38	No_reach	6.602	0.000	0.0000	38	No_reach	5.727	0.000	0.0000
39	No_reach	4.871	0.000	0.0000	39	No_reach	4.863	0.000	0.0000
40	No_reach	0.000	0.000	0.0000	40	No_reach	0.000	0.000	0.0000
41	No_reach	1.000	0.000	0.0000	41	No_reach	1.000	0.000	0.0000
42	1	4.907	8.034	-0.0007	42	1	4.900	8.034	-0.0007
43	2	6.410	8.034	-0.0001	43	2	5.640	8.034	-0.0002
44	3	6.983	8.034	-0.0002	44	3	5.907	8.034	-0.0003
45	No_reach	8.131	0.000	0.0000	45	No_reach	6.338	0.000	0.0000
46	4	8.758	8.034	0.0002	46	4	6.593	8.034	-0.0003
47	5	9.713	8.034	0.0001	47	5	6.942	8.034	-0.0001
48	6	10.156	8.034	0.0003	48	6	7.116	8.034	-0.0001
49	9	11.063	8.034	0.0006	49	9	7.508	8.034	-0.0001
50	10	11.404	11.146	0.0001	50	10	7.762	11.146	-0.0008
51	11	11.164	11.146	0.0000	51	11	7.686	11.146	-0.0003
52	12	11.135	11.146	0.0000	52	12	7.678	11.146	-0.0005
53	No_reach	10.339	0.000	0.0000	53	No_reach	7.256	0.000	0.0000
54	13	10.296	8.034	0.0005	54	13	7.193	8.034	-0.0002
55	14	9.219	8.034	0.0001	55	14	6.803	8.034	-0.0001
56	15	8.934	8.034	0.0001	56	15	6.723	8.034	-0.0002
57	No_reach	7.530	0.000	0.0000	57	No_reach	6.230	0.000	0.0000
58	21	7.279	11.146	-0.0009	58	21	6.243	11.146	-0.0011
59	22	4.884	11.146	-0.0005	59	22	4.874	11.146	-0.0005
60	23	0.000	11.146	0.0000	60	23	0.000	11.146	0.0000
61	No_reach	1.000	0.000	0.0000	61	No_reach	1.000	0.000	0.0000
62	No_reach	4.894	0.000	0.0000	62	No_reach	4.888	0.000	0.0000
63	No_reach	5.984	0.000	0.0000	63	No_reach	5.383	0.000	0.0000
64	No_reach	6.470	0.000	0.0000	64	No_reach	5.611	0.000	0.0000
65	No_reach	7.503	0.000	0.0000	65	No_reach	6.037	0.000	0.0000
66	No_reach	8.025	0.000	0.0000	66	No_reach	6.260	0.000	0.0000
67	No_reach	8.962	0.000	0.0000	67	No_reach	6.628	0.000	0.0000
68	No_reach	9.480	0.000	0.0000	68	No_reach	6.835	0.000	0.0000
69	8	10.283	8.034	0.0004	69	8	7.144	8.034	-0.0002
70	No_reach	10.478	0.000	0.0000	70	No_reach	7.253	0.000	0.0000
71	No_reach	10.281	0.000	0.0000	71	No_reach	7.215	0.000	0.0000

72	No_reach	10.369	0.000	0.0000	72	No_reach	7.281	0.000	0.0000
73	No_reach	9.751	0.000	0.0000	73	No_reach	7.009	0.000	0.0000
74	No_reach	9.630	0.000	0.0000	74	No_reach	6.958	0.000	0.0000
75	No_reach	8.680	0.000	0.0000	75	No_reach	6.558	0.000	0.0000
76	No_reach	8.408	0.000	0.0000	76	No_reach	6.449	0.000	0.0000
77	No_reach	7.056	0.000	0.0000	77	No_reach	5.878	0.000	0.0000
78	No_reach	6.628	0.000	0.0000	78	No_reach	5.712	0.000	0.0000
79	No_reach	4.871	0.000	0.0000	79	No_reach	4.863	0.000	0.0000
80	No_reach	0.000	0.000	0.0000	80	No_reach	0.000	0.000	0.0000
81	No_reach	1.000	0.000	0.0000	81	No_reach	1.000	0.000	0.0000
82	No_reach	4.891	0.000	0.0000	82	No_reach	4.887	0.000	0.0000
83	No_reach	5.699	0.000	0.0000	83	No_reach	5.244	0.000	0.0000
84	No_reach	6.103	0.000	0.0000	84	No_reach	5.422	0.000	0.0000
85	No_reach	7.029	0.000	0.0000	85	No_reach	5.818	0.000	0.0000
86	No_reach	7.492	0.000	0.0000	86	No_reach	6.016	0.000	0.0000
87	No_reach	8.407	0.000	0.0000	87	No_reach	6.392	0.000	0.0000
88	No_reach	8.864	0.000	0.0000	88	No_reach	6.580	0.000	0.0000
89	7	9.681	8.034	0.0002	89	7	6.908	8.034	-0.0001
90	No_reach	9.887	0.000	0.0000	90	No_reach	6.998	0.000	0.0000
91	No_reach	9.902	0.000	0.0000	91	No_reach	7.029	0.000	0.0000
92	No_reach	9.909	0.000	0.0000	92	No_reach	7.045	0.000	0.0000
93	No_reach	9.552	0.000	0.0000	93	No_reach	6.907	0.000	0.0000
94	No_reach	9.373	0.000	0.0000	94	No_reach	6.838	0.000	0.0000
95	No_reach	8.639	0.000	0.0000	95	No_reach	6.530	0.000	0.0000
96	No_reach	8.272	0.000	0.0000	96	No_reach	6.375	0.000	0.0000
97	No_reach	7.129	0.000	0.0000	97	No_reach	5.885	0.000	0.0000
98	No_reach	6.558	0.000	0.0000	98	No_reach	5.639	0.000	0.0000
99	No_reach	4.918	0.000	0.0000	99	No_reach	4.909	0.000	0.0000
100	No_reach	0.000	0.000	0.0000	100	No_reach	0.000	0.000	0.0000

Table 34. 1 meter river width case results from SFR and Python module with a 1 second duration

11.5 Riverbed width 0.1 meters

Python					SFR				
cell_id	reach_id	gw head time 1.0 (m)	h stage time 1.0 (m)	q leak time 1.0 (m3/s)	cell_id	reach_id	gw head time 1.0 (m)	h stage time 1.0 (m)	q leak time 1.0 (m3/s)
1	No_reach	1.000	0.000	0.0000	1	No_reach	1.000	0.000	0.0000
2	No_reach	4.930	0.000	0.0000	2	No_reach	4.930	0.000	0.0000
3	No_reach	5.792	0.000	0.0000	3	No_reach	5.792	0.000	0.0000
4	No_reach	6.111	0.000	0.0000	4	No_reach	6.111	0.000	0.0000
5	No_reach	6.747	0.000	0.0000	5	No_reach	6.747	0.000	0.0000
6	No_reach	6.968	0.000	0.0000	6	No_reach	6.968	0.000	0.0000
7	No_reach	7.411	0.000	0.0000	7	No_reach	7.411	0.000	0.0000
8	No_reach	7.534	0.000	0.0000	8	No_reach	7.534	0.000	0.0000
9	No_reach	7.779	0.000	0.0000	9	No_reach	7.779	0.000	0.0000
10	No_reach	7.802	0.000	0.0000	10	No_reach	7.802	0.000	0.0000
11	No_reach	7.848	0.000	0.0000	11	No_reach	7.848	0.000	0.0000
12	No_reach	7.762	0.000	0.0000	12	No_reach	7.762	0.000	0.0000
13	No_reach	7.591	0.000	0.0000	13	No_reach	7.591	0.000	0.0000
14	No_reach	7.369	0.000	0.0000	14	No_reach	7.369	0.000	0.0000

15	No_reach	6.924	0.000	0.0000	15	No_reach	6.924	0.000	0.0000
16	No_reach	6.579	0.000	0.0000	16	No_reach	6.579	0.000	0.0000
17	No_reach	5.887	0.000	0.0000	17	No_reach	5.887	0.000	0.0000
18	No_reach	5.545	0.000	0.0000	18	No_reach	5.545	0.000	0.0000
19	No_reach	4.861	0.000	0.0000	19	No_reach	4.861	0.000	0.0000
20	No_reach	0.000	0.000	0.0000	20	No_reach	0.000	0.000	0.0000
21	No_reach	1.000	0.000	0.0000	21	No_reach	1.000	0.000	0.0000
22	No_reach	4.892	0.000	0.0000	22	No_reach	4.892	0.000	0.0000
23	No_reach	5.763	0.000	0.0000	23	No_reach	5.763	0.000	0.0000
24	No_reach	6.018	0.000	0.0000	24	No_reach	6.018	0.000	0.0000
25	No_reach	6.755	0.000	0.0000	25	No_reach	6.755	0.000	0.0000
26	No_reach	6.941	0.000	0.0000	26	No_reach	6.941	0.000	0.0000
27	No_reach	7.506	0.000	0.0000	27	No_reach	7.506	0.000	0.0000
28	No_reach	7.609	0.000	0.0000	28	No_reach	7.609	0.000	0.0000
29	No_reach	8.011	0.000	0.0000	29	No_reach	8.011	0.000	0.0000
30	No_reach	7.979	0.000	0.0000	30	No_reach	7.979	0.000	0.0000
31	No_reach	8.113	0.000	0.0000	31	No_reach	8.113	0.000	0.0000
32	17	8.066	26.022	-0.0002	32	17	8.066	26.022	-0.0001
33	16	8.012	26.022	-0.0004	33	16	8.012	26.022	-0.0002
34	18	7.863	26.022	-0.0006	34	18	7.863	26.022	-0.0004
35	No_reach	7.355	0.000	0.0000	35	No_reach	7.355	0.000	0.0000
36	19	7.172	26.022	-0.0008	36	19	7.172	26.022	-0.0005
37	20	6.383	26.022	-0.0004	37	20	6.383	26.022	-0.0003
38	No_reach	5.879	0.000	0.0000	38	No_reach	5.879	0.000	0.0000
39	No_reach	4.864	0.000	0.0000	39	No_reach	4.864	0.000	0.0000
40	No_reach	0.000	0.000	0.0000	40	No_reach	0.000	0.000	0.0000
41	No_reach	1.000	0.000	0.0000	41	No_reach	1.000	0.000	0.0000
42	1	4.898	26.022	-0.0007	42	1	4.898	26.022	-0.0005
43	2	5.788	26.022	-0.0002	43	2	5.788	26.022	-0.0002
44	3	6.124	26.022	-0.0004	44	3	6.124	26.022	-0.0003
45	No_reach	6.762	0.000	0.0000	45	No_reach	6.762	0.000	0.0000
46	4	7.119	26.022	-0.0007	46	4	7.119	26.022	-0.0004
47	5	7.633	26.022	-0.0002	47	5	7.633	26.022	-0.0001
48	6	7.866	26.022	-0.0005	48	6	7.866	26.022	-0.0003
49	9	8.332	26.022	-0.0006	49	9	8.332	26.022	-0.0004
50	10	8.480	38.411	-0.0014	50	10	8.480	38.411	-0.0007
51	11	8.351	38.411	-0.0005	51	11	8.351	38.411	-0.0002
52	12	8.341	38.410	-0.0009	52	12	8.341	38.410	-0.0004
53	No_reach	7.949	0.000	0.0000	53	No_reach	7.949	0.000	0.0000
54	13	7.953	26.021	-0.0007	54	13	7.953	26.021	-0.0004
55	14	7.362	26.021	-0.0002	55	14	7.362	26.021	-0.0001
56	15	7.212	26.021	-0.0005	56	15	7.212	26.021	-0.0003
57	No_reach	6.397	0.000	0.0000	57	No_reach	6.397	0.000	0.0000
58	21	6.252	38.410	-0.0013	58	21	6.252	38.410	-0.0007
59	22	4.871	38.410	-0.0005	59	22	4.871	38.410	-0.0002
60	23	0.000	38.410	0.0000	60	23	0.000	38.410	0.0000
61	No_reach	1.000	0.000	0.0000	61	No_reach	1.000	0.000	0.0000
62	No_reach	4.890	0.000	0.0000	62	No_reach	4.890	0.000	0.0000
63	No_reach	5.521	0.000	0.0000	63	No_reach	5.521	0.000	0.0000
64	No_reach	5.805	0.000	0.0000	64	No_reach	5.805	0.000	0.0000
65	No_reach	6.389	0.000	0.0000	65	No_reach	6.389	0.000	0.0000

66	No_reach	6.687	0.000	0.0000	66	No_reach	6.687	0.000	0.0000
67	No_reach	7.209	0.000	0.0000	67	No_reach	7.209	0.000	0.0000
68	No_reach	7.497	0.000	0.0000	68	No_reach	7.497	0.000	0.0000
69	8	7.940	26.022	-0.0006	69	8	7.940	26.022	-0.0004
70	No_reach	8.030	0.000	0.0000	70	No_reach	8.030	0.000	0.0000
71	No_reach	7.908	0.000	0.0000	71	No_reach	7.908	0.000	0.0000
72	No_reach	7.945	0.000	0.0000	72	No_reach	7.945	0.000	0.0000
73	No_reach	7.614	0.000	0.0000	73	No_reach	7.614	0.000	0.0000
74	No_reach	7.552	0.000	0.0000	74	No_reach	7.552	0.000	0.0000
75	No_reach	7.029	0.000	0.0000	75	No_reach	7.029	0.000	0.0000
76	No_reach	6.884	0.000	0.0000	76	No_reach	6.884	0.000	0.0000
77	No_reach	6.116	0.000	0.0000	77	No_reach	6.116	0.000	0.0000
78	No_reach	5.874	0.000	0.0000	78	No_reach	5.874	0.000	0.0000
79	No_reach	4.864	0.000	0.0000	79	No_reach	4.864	0.000	0.0000
80	No_reach	0.000	0.000	0.0000	80	No_reach	0.000	0.000	0.0000
81	No_reach	1.000	0.000	0.0000	81	No_reach	1.000	0.000	0.0000
82	No_reach	4.888	0.000	0.0000	82	No_reach	4.888	0.000	0.0000
83	No_reach	5.353	0.000	0.0000	83	No_reach	5.353	0.000	0.0000
84	No_reach	5.585	0.000	0.0000	84	No_reach	5.585	0.000	0.0000
85	No_reach	6.113	0.000	0.0000	85	No_reach	6.113	0.000	0.0000
86	No_reach	6.377	0.000	0.0000	86	No_reach	6.377	0.000	0.0000
87	No_reach	6.894	0.000	0.0000	87	No_reach	6.894	0.000	0.0000
88	No_reach	7.152	0.000	0.0000	88	No_reach	7.152	0.000	0.0000
89	7	7.612	26.022	-0.0003	89	7	7.612	26.022	-0.0002
90	No_reach	7.720	0.000	0.0000	90	No_reach	7.720	0.000	0.0000
91	No_reach	7.714	0.000	0.0000	91	No_reach	7.714	0.000	0.0000
92	No_reach	7.711	0.000	0.0000	92	No_reach	7.711	0.000	0.0000
93	No_reach	7.509	0.000	0.0000	93	No_reach	7.509	0.000	0.0000
94	No_reach	7.408	0.000	0.0000	94	No_reach	7.408	0.000	0.0000
95	No_reach	7.000	0.000	0.0000	95	No_reach	7.000	0.000	0.0000
96	No_reach	6.796	0.000	0.0000	96	No_reach	6.796	0.000	0.0000
97	No_reach	6.155	0.000	0.0000	97	No_reach	6.155	0.000	0.0000
98	No_reach	5.834	0.000	0.0000	98	No_reach	5.834	0.000	0.0000
99	No_reach	4.911	0.000	0.0000	99	No_reach	4.911	0.000	0.0000
100	No_reach	0.000	0.000	0.0000	100	No_reach	0.000	0.000	0.0000

Table 35. 0.1 meters river width case results from SFR and Python module with a 1 second duration