Hydrochemical facies analysis for large coastal groundwater model validation

MSc Thesis

Student name	Student number
J.J. (Justus) Krantz	4472713

to obtain the degree of

Master of Science

in Geotechnical Engineering

at the Delft University of Technology,

Faculty of Civil Engineering and Geosciences

To be defended on June 6th, 2023, at 13:00

in lecture hall G at the Faculty of Civil Engineering and Geosciences

Assessment committee

Preface

This thesis concludes more than a year of researching and groundwater modeling at Deltares. Above all, however, it marks the end of my time as an MSc student at the faculty of Civil Engineering and Geosciences, and as a student at the Delft University of Technology in general. I feel that I've grown a lot. I gained an analytical thinking approach that I can apply to many problems, perhaps even ones that aren't in my field of study.

I would like to express my gratitude to my supervisors: Timo Heimovaara, Huite Bootsma and Guillaume Rongier. Timo's guidance has been enthusiastic, inspiring, and friendly, throughout both my BSc as my MSc thesis. Guillaume's detailed feedback, and guidance on the larger context of my thesis have had a stimulating effect on this thesis's quality.

Huite's role as my daily supervisor has been a pleasant experience for me during my time at Deltares. I reached out to him because I wanted to learn groundwater modeling at a company with expertise, from a person with expertise. He was always approachable for advice, supervision, and guidance, and I tried my best to incorporate everything he advised me to do. On the way I learned a thing or two besides technical skills, on my transition to an engineer in the field of geohydrology. Deltares has been a good host to me during my graduation thesis, I would like to express my gratitude to the company.

I would also like to thank Pieter Jan Stuyfzand, for providing me his hydrochemical cross section perpendicular to the coastline, through the study area. His thorough and detailed work on groundwater in the dunes along the coastline of the Netherlands have been a solid foundation inspiration for this study, and hopefully many more to come.

My final, though not least, gratitude goes out to my girlfriend Jessy, my parents and my brother and sister in-law, my friends, and of course my colleague-friends at Clone Records. Thank you for your support, for distracting my mind from codes, and thank you for bearing with me.

I would like to conclude with a thought that I could not express in a technical report. During one of our first meetings to discuss the progress of my thesis, Timo Heimovaara asked me what the ultimate goal of a (geohydrological) model is. In a rush to the answer, I said that in the future, field observations would not be necessary again. In the future, we would have models whose predictive capabilities would match physical processes precisely.

A thing I learned along the way, however, is that uncertainty is an inherent quality to groundwater modeling, and I think that this is also what intrigues me now, at the end of my thesis. This quality intrigues me to continue my career in this field. I state uncertainty as a quality, because it reminds me that there will always be a challenge, an uncertainty left to deal with. When I started my first year as a student, I would never have thought that there would be a lesson to be learnt about ambiguity in general in life, through groundwater modeling. I am not sure it is the ultimate goal of a geohydrological model, although it's an interesting thought.

Abstract

Variable density groundwater models are essential for managing coastal groundwater resources. However, their practical applicability can be questioned due to limited validation opportunities on long timescales associated with the development of fresh-saline distributions. This study addresses this challenge by applying upscaled metamodeling techniques to a state-of-the-art variable density groundwater model (the original model) for the Meijendel-Berkheijde drinking water reservoir. Model validation is performed using a Hydrochemical Facies Analysis (HyFA) conducted by Stuyfzand (1993) in the same area.

The primary objective of this study is to enhance validation techniques for variable density groundwater models by incorporating the HyFA. Unlike traditional snapshot-based validation, the HyFA enables validation of groundwater pathways calculated by the model. The applied metamodeling approach significantly reduces calculation times by implementing an upscaled horizontal grid size, parameter rescaling, and linear boundary conditions, thereby enhancing computational efficiency.

Although the original model lacks long-term salinity validation, it has not been invalidated based on the similarity of metamodel outputs to the HyFA. However, in the northern part of the study area, a potentially excessive conductance term may result in higher infiltration rates. Incorporating the HyFA into the metamodel is straightforward by adding a "species" dimension in the SEAWAT structure. This validation technique proves valuable for assessing variable density groundwater models on shorter timescales, particularly in areas affected by extensive human interventions.

This study contributes to collaborative efforts by Dunea, Deltares, and Arcadis (2021), aimed at advancing efficient modeling for the coastal groundwater reserve of Meijendel-Berkheijde. Transparent documentation of detailed model scripts ensures reproducibility and provides a valuable resource for future research. The insights gained from this study have implications for global advancements in coastal groundwater management.

Keywords: *Hydrochemical Facies Analysis, variable density groundwater modeling, upscaled metamodeling, coastal groundwater management.*

Contents

1. Introduction

Fresh groundwater resources in coastal regions are a topic of study that will become increasingly important in the face of challenges of modern times: environmental, socio-economic, and geopolitical. In coastal areas, fresh groundwater resources are susceptible to degradation due to their proximity to saline seawater, high demands, land use change, climate change and sea level fluctuations (Werner et. al., 2012). Coastal regions are home to a significant share of humanity: in 2016, 44% of the world population was living within a 150km range of the coastline (J. Akrofi, 2016), where people largely rely on fresh groundwater resources, for domestic, industrial, and agricultural use (Delsman, 2015).

The Netherlands has an elaborate history of water management. First accounts date back to 800AD, when drainage of coastal salt marches commenced (Delsman, 2015). Its rural landscape is characterized by polders, canals, and reclaimed lakes, initially drained using windmills. This pursuit dates to 1000AD (van der Ven, 1993), and is keeping water tables sufficiently low for agricultural use. Consequently, this drainage also resulted in compaction of soils due to oxidation of organic matter in peat soils, which were now exposed to air (Hoogland et. al., 2012). Consequently, part of the Netherlands is now situated below the mean sea level. Figure 1 gives an overview of threats to coastal aquifers and fresh groundwater resources (Delsman, 2015).

This study aims to assess and enhance state of the art groundwater modeling techniques for freshwater reservoirs in dunes. In Section 1.1, the area of Meijendel-Berkheijde, situated along the coastline of the Netherlands, is judged a suitable study area for this aim, by virtue of desirable study conditions. Section 1.2 describes the state-of-the art: a groundwater model that incorporates state-of-the-art groundwater modeling techniques has been developed for the area, although its applicability can be questioned. Section 1.3 describes the scope of this study, motivated by the conditions described in Sections 1.1 and 1.2, and states the objectives and sub-objectives. Section 1.4 provides the reader with an overview of the report.

The insights and analyses from this study may contribute to developing more efficient groundwater models (i.e., models that require less effort to construct and less time to run), while capturing uncertainties. These models could contribute to global efforts to guarantee the robustness and sustainability of drinking water resources in coastal areas globally.

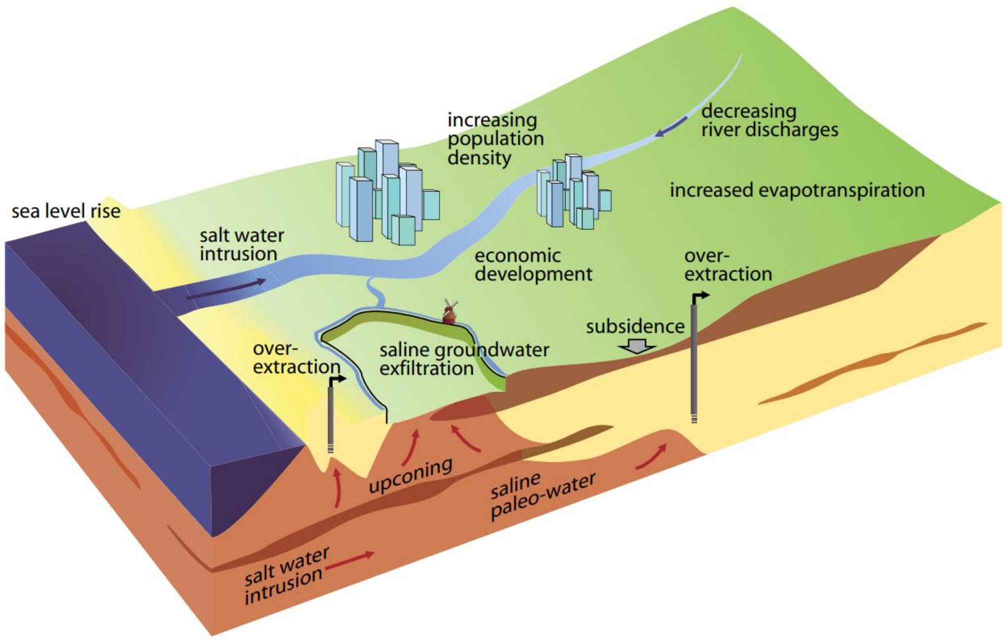


Figure 1: Threats to coastal aquifers and fresh groundwater reserves in the Netherlands. Retrieved from Delsman (2015)

1.1. The coastal dunes of Meijendel-Berkheide as a study area

The Netherlands is the third most densely populated country in the world (U.N. Statistics Division, 2021). The densely populated coastal dunes and adjacent polders of the Western Netherlands suffer from all environmental ills of modern times. They are faced with a wide spectrum of natural variations and anthropogenic impacts (Stuyfzand, 1993a). Intrusion of saline groundwater, over extraction of freshwater resources, increase in demand due to growth, these are threats that coastal dunes of the Western Netherlands suffer.

In the dune area of Meijendel-Berkheijde, situated along the coastline of the Netherlands, water management has a particular history, making it ideal to study the development of fresh-saline groundwater interactions over time, with human intervention. In the period of 1874 to 1955, excessive groundwater exfiltration caused significant desiccation and groundwater salinization in the dune area of Meijendel-Berkheide (Stuyfzand et al., 1993b). To recharge the freshwater reserves, infiltration ponds were installed and river water from the Rhine was pumped into the coastal aquifers since 1956. Since 1976 additional water from the Meuse River is pumped into the dune reserves to further enhance the reservoir's recharge (see Figure 3).

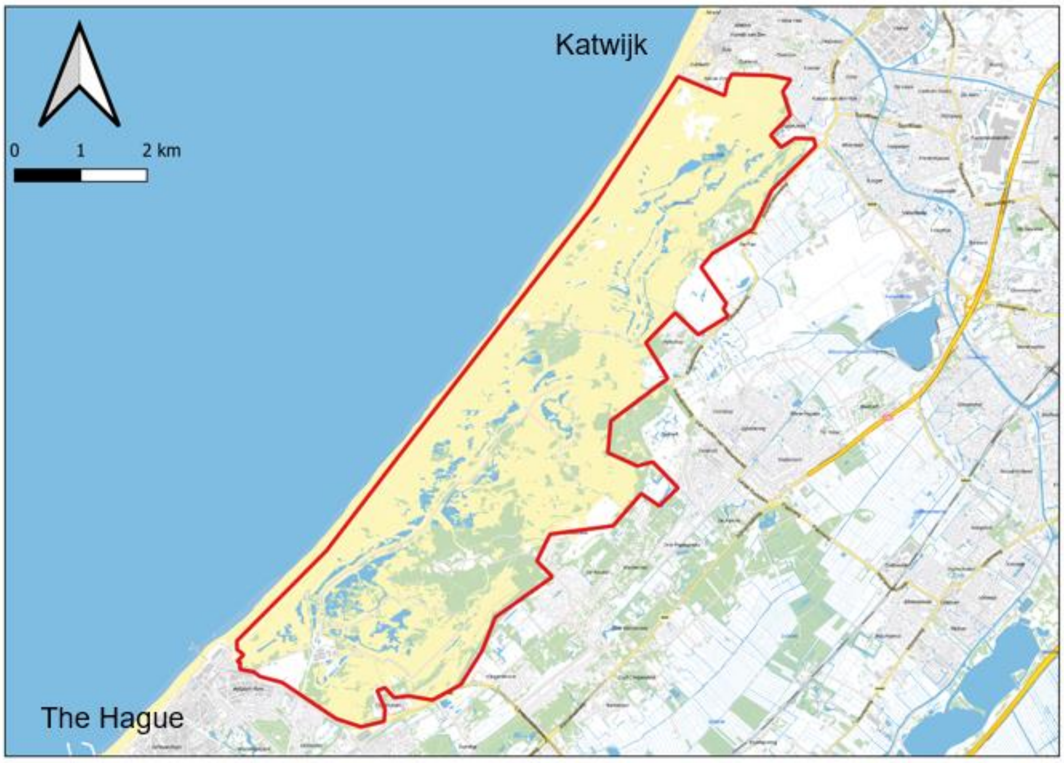


Figure 2: Study area (in red) of Meijendel-Berkheijde, near The Hague, Netherlands.

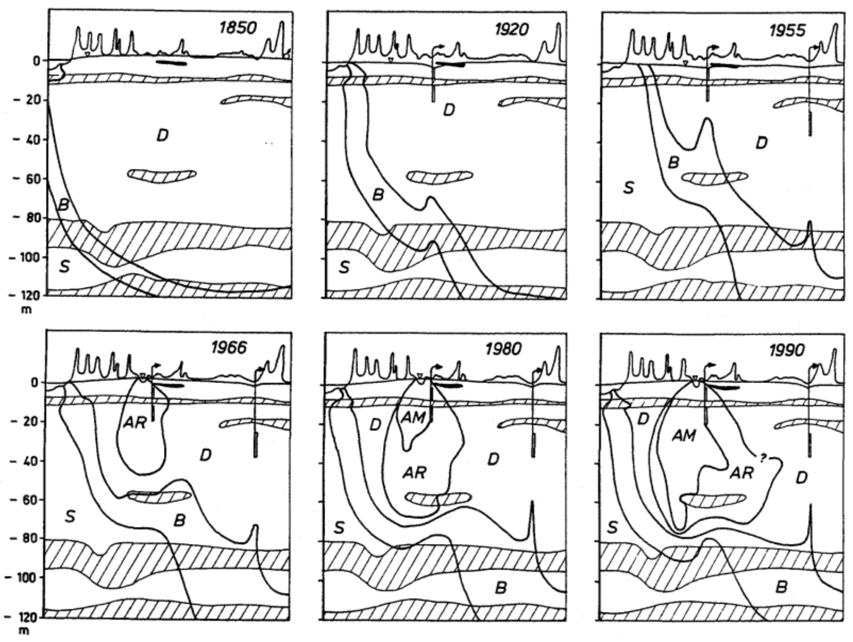


Figure 3: Saline groundwater intrusion and overextraction in Meijendel-Berkheide, interpreted by (Stuyfzand et. al, 1993b). Cross section perpendicular to coastline, from East to West (left-right). AM = Artificially infiltrated Meuse water, AR = Artificially infiltrated Rhine water, D = Dune freshwater, B = Brackish groundwater.

It is found that long timescales are involved in establishing the current fresh-saline distribution in deltas (van Engelen, 2020), and groundwater salinity in coastal aquifers of the Netherlands predominantly derives from sea water infiltration during Holocene marine transgressions (Post and Kooi, 2003). However, human intervention in may disrupt this long timescale transience. As recognized by Oude Essink (2001), in areas where extensive human intervention (like extraction and injection of groundwater) takes place, the anthropogenic influence on fresh-saline distributions may become primary, on a much smaller timescale. This is underlined for the area of Meijendel-Berkheijde, by the Hydrochemical Facies Analysis of Stuyfzand (1993a).

Moreover, most data available are “snapshots” and provide little information on the long-term development of fresh-saline groundwater distributions. Modern techniques like the use of airborne electromagnetics to map groundwater salinity are on a relatively small timescale compared to the development of fresh-saline groundwater distributions. This lack of historical data causes model validation to be based on “snapshots”. A Hydrochemical Facies Analysis (HyFA) like Stuyfzand, (1993a) could provide a contribution to model validation, as origin tracing provides insight in the “path” of water flow in the subsurface. Its use for variable density groundwater model validation requires research.

To summarize, the area of the Meijendel-Berkheijde is judged as suitable for studying coastal fresh-saline groundwater distributions by virtue of (a) the elaborate geochemical analysis of Stuyfzand (1993a) giving insight in the path of groundwater flow in the area, providing a research opportunity for fresh-saline groundwater model validation; (b) numerous studies with a successful numerical groundwater modeling approach in the area and (c) site specific human-induced conditions that disrupt the long timescales over which fresh-saline interfaces develop, consequently making the area suitable for modeling fresh-saline groundwater distributions efficiently, over a shorter period of time.

1.2. State of the Art

A prerequisite to sustainable management of coastal fresh groundwater reserves is an accurate description of their present-day distribution, which is difficult to obtain due to the sparse measurements at depth (Delsman, 2015). The area of Meijendel-Berkheijde may be an exception to this statement, as numerous studies have been conducted on its geohydrology, which provide a basis for further research. Stuyfzand (1993a) conducted a Hydrochemical Facies Analysis (HyFA) to identify the origins of over 2,000 groundwater samples collected along the coastline; Oude Essink (2001) introduced the concept of a fresh-saline groundwater interface to a groundwater model for the entire coastline and discussed problems that arise in the Dutch coastline as a result of variable density flow; Delsman (2015) used a two-dimensional numerical model to perform a paleo-hydrological reconstruction, to investigate intrusion of saline groundwater from the sea over a longer timescale, throughout the Holocene. Moreover, Delsman (2015) validated the respective model's output to the HyFA of Stuyfzand (1993a), see Figure 4.

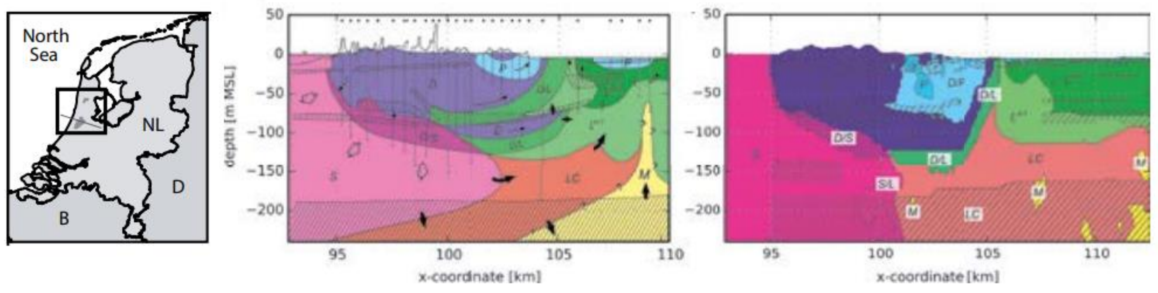


Figure 4: Location of two dimensional transect (left); Position of hydrosomes, inferred from hydrochemical facies analysis (middle, adapted from Figure 4.6 in Stuyfzand (1993a); and from modeled origin tracers. (Delsman, 2015)

Dunea manages drinking water supply to The Hague from the fresh groundwater reservoir in Meijendel-Berkheijde. They seek to increase their supply with 30% from 2021 to 2025 to account for future increases in demand due to population growth, and deficits in precipitation due to climate change (Bootsma et al., 2021). In a preliminary study to investigate the feasibility of brackish groundwater as an additional source for producing drinking water, Bootsma et. al. (2021) used a numerical three-dimensional state-of-the-art variable density groundwater flow model to find that the extraction of brackish groundwater as a drinking water resource may be feasible, but that further research is necessary.

However, the state-of-the-art model's applicability and reliability can be questioned, since it has only been validated on hydraulic head data (Bootsma et.al., 2021) and water balances of the infiltration and extraction system. Using this model to also predict the behavior of the salinity distribution in the future may be risky extrapolation. Moreover, long term modeling exercises to gauge how this model simulates the salinity distribution development is impractical and costly: even on a 32-core (Amazon Web Services) machine, a simulation of 1000 years takes over two months to complete (Bootsma, H., personal communication).

Another perspective on the state of the art in hydrological modeling is the restraining effect of poor reproducibility. Advances in hydrological modeling and on groundwater resource research is typically poorly reproducible. Stagge et al. (2019) estimate that results may only be reproducible for between 0.6% and 6.8% of nearly 2000 peer-reviewed manuscripts published in six hydrology and water resources journals, due to a lack of sufficiently clearly described methods and a lack of the necessary input data or processing code (Knoben et. al., 2022).

In summary, an assessment of state-of-the-art modeling techniques for the Meijendel-Berkheijde groundwater reservoir is called for by i) the state-of-the-art groundwater model's questionable validation; ii) its long calculation times, making it impractical to study the development of fresh-saline groundwater behaviour. This assessment may contribute to efforts in coastal drinking water resource management globally, through transparency on its methods and code.

1.3. Scope of study

3D modeling techniques for fresh-saline groundwater interactions in coastal regions can be seen as the central concept of this study. The starting point for this study is the state-of-the-art groundwater flow model currently used for coastal groundwater management by The Hague's fresh drinking water supplier Dunea (Bootsma et. al., 2021). This model will be referred to as the "original model" throughout the remainder of this report. Motivated by the conditions named in Sections 1.1 and 1.2, the aim of this study is:

To assess the role of the Hydrochemical Facies Analysis as a model validation technique for variable density groundwater flow models used to study coastal drinking water resources.

Associated with this statement are the following sub-objectives:

1. Assess the role of the Hydrochemical Facies Analysis (HyFA) conducted by Stuyfzand (*1993*) on the study area of Meijendel-Berkheijde.
    a. Investigate the hydrochemical section that crosses the study area perpendicular to the coastline for model validation.
    b. Investigate the hydrochemical section that crosses the study area along the coastline for model validation.
2. Assess the role of the conceptual fresh-saline interface for model validation.
    a. Integrating the effects of variable density flow for model validation, by incorporating the conceptual fresh-saline groundwater interface as a sharp interface between fresh and brackish groundwater.

The main method to reach the objective of this study is by upscaled metamodeling. Although various definitions and applications of metamodeling are found in literature, this study follows a general definition by Barton (1998): A metamodel can be broadly defined as a model of a model, characteristically more computationally efficient than the original model and with fewer parameters. Beyond this general definition, the form of the metamodel in different applications varies depending on the nature of desired model output and the characteristics of the problem (Fraser et. al., 2013).

In this study, the original model that is used as a starting point for the metamodel, is the state-of-the-art variable density groundwater model by Bootsma et al. (2021), whose calculation times are impractical and whose role as a well validated variable density groundwater model can be questioned.

Studying model efficiency and accuracy are consequential sub-objectives associated to increasing the original model's efficiency and assessing the role of HyFA as a model validation technique. Therefore, the following sub-objective is established:

3. Use appropriate rescaling methods that increase model efficiency, but retain sufficient accuracy, whilst ensuring reproducibility for global efforts in coastal groundwater management.
    a. Rescaling methods: rescaling parameters to a new grid size, define and calibrate boundary conditions accordingly.
    b. Increase model efficiency: Efficiency is defined in this study by the calculation time of the groundwater model: the shorter the model's calculation time, the higher the model efficiency.
    c. Retain sufficient model accuracy: Accuracy is determined in this study by the discrepancy between the constructed metamodel and the state-of-the-art groundwater model.
    d. Ensure reproducibility of findings of this study through transparency on methods, input data, results, and detailed model scripts.

1.4. Report overview

This study is divided into four chapters, to present the methods and results in a concise and linear manner. Chapters 2 and 3 are used to present methods and results, both addressing the objectives: “Metamodeling” (Chapter 2) addresses sub-objective 1, which arises when addressing sub-objectives 1 and 2; “Model validation on groundwater salinity” (Chapter 3) addresses sub-objectives 1 and 2. The chapters aim to approach the objectives from a conceptual perspective, as well as from a practical point of view. The findings are recollected into an overarching discussion and conclusion in Chapter 4, to address the aim of this thesis. All the supporting material is provided in the appendices to be referred to throughout this paper.

Chapter 1 shares the motivation for this study in an introduction. First, the state of the art in the context of the study is discussed in Section 1.1. In section 1.2, the suitability of the coastal dunes of Meijendel-Berkheijde as a study area is discussed. Subsequently, the scope and objectives of the study are presented (Section 1.3). The structure of the paper is presented in Section 1.4.

Chapter 2 discusses metamodeling as the method to reach the 3rd sub-objective of this study, as discussed in Section 1.3. This chapter describes how the metamodel is constructed and how upscaling is performed in section 2.1. In Section 2.2, the results are presented and discussed.

Chapter 3 addresses sub-objectives 1 and 2 to fulfil the aim of this study. The metamodel developed in Chapter 2 will be used to: i) study the feasibility of HyFA and the conceptual fresh-saline interface for model validation; ii) assess the original model’s validation on groundwater salinity. The methodology is discussed in Section 3.1 and the results are presented and discussed in Section 3.2.

Chapter 4 serves as an overarching discussion and conclusion. In this chapter, the findings from Chapters 2 and 3 are recollected to a discussion that regards the objectives of this study and approaches the results from a global perspective, to contribute to efforts in the field of drinking water management in coastal areas. The conclusions of this study are summarized in Section 4.3, and in Section 4.4 recommendations for further research are given.

2. Metamodeling

2.1. Upscaling: methodology

To fulfil the aim of this study and 1st and 2nd sub-objectives, the starting point for this study must first be considered. Using appropriate rescaling methods is considered as the 3rd sub-objective that is consequential to reaching the first two sub-objectives. This section discusses one of the main methods of this study: upscaling through metamodeling. Specifically, this section is devoted to i) increasing model efficiency; ii) retaining model accuracy; iii) ensuring reproducibility through transparency on methods, input data and detailed script structure.

2.1.1. The original model

The state-of-the-art groundwater model for the groundwater reservoir of Meijendel-Berkheijde, is the result of previous modeling efforts in the area. In this study this model is referred to as the original model, as it provides the base on which the metamodel is constructed. The starting point for this study is the original model's input and output datasets. It should be noted that the original model's scripts and script structure were not accessed. The original model's input data are used for constructing and running the metamodel, the original model's output data are used for analyzing the discrepancies between the two model outputs, and consequently for calibration.

The original model domain, discretization, and according parametrization are based on the preceding "bridging model" constructed for the drinking water reservoir by Arcadis, Deltares and KWR (2019), used to investigate the drinking water reservoir's robustness against a period of drought (reduced recharge and infiltration). The model domain (of the original model, and consequently, the metamodel) in Figure 5. In it, the subsurface is represented by two aquifers and two aquitards, for illustration purpose only.

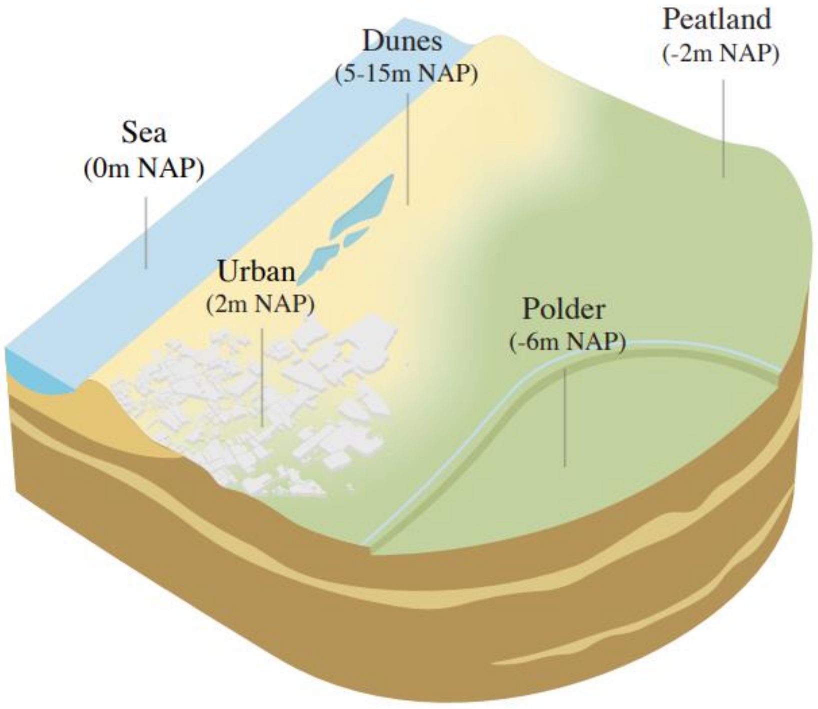


Figure 5: Original model domain and piezometric groundwater heads. In the dunes, the infiltration ponds in the study area of Meijendel-Berkheijde can be seen. The subsurface is represented (only visually) by two aquifers, a simplified representation, for illustration purpose (In the actual models, there is more variability in the subsurface). The urban area shown represents The Hague.

Along the coastline, the original model domain ranges from Loosduinen (south), to Katwijk (north). Inlands it reaches until Zoetermeer (west). The original model has a horizontal cell size of 25x25m (Bootsma et. al. 2021). Vertically, the model top reaches the follows the topography of the dunes, (15m NAP) and reaches a depth of 250m below NAP. The top 30m of the model domain consist of 15 cells that are each 2m thick, followed by 22 cells of 5m thick and 12 cells of 10m thick. (Bootsma et. al., 2021)

Corresponding to the original model's discretization is its parametrization, i.e., the soil parameters that the cells contain to represent a volume of soil: groundwater salinity and hydraulic conductivities. The starting groundwater salinity results from a 100y simulation of the original model, which in turn used a starting groundwater salinity provided by TNO. This starting salinity was assumed to be steady state under the constant fluxes in the original model. The hydraulic conductivities originate from three data sources: (i) REGIS, a 3D layer model of the subsurface of the Netherlands with a horizontal resolution of 100m, and variable vertical resolution up to a maximum depth of 500m (TNO, Ministry of Internal Affairs) is used for the deep subsurface; (ii) GeoTOP, with a smaller vertical resolution than REGIS (TNO, Ministry of Internal Affairs) for the shallow subsurface *outside* the study area of Meijendel-Berkheijde (but in the model domain); (iii) The parametrization used for the TRIWACO model by Royal Haskoning DHV (2018), is also applied to the study area (Meijendel-Berkheijde), due to a higher borehole density in this specific area.

Although the original model is the result of numerous efforts that use state-of-the-art techniques and data sources, it should be noted that uncertainties are to be expected in this model's in and outputs, as it is based on other models (each with their uncertainty), observation data (with a certain sampling interval and measurement error) and finally because a complex geohydrological system is now represented by a grid with a certain cell size (25m horizontally, variable vertically), with a reduced ability to simulate small scale groundwater flows. It should be kept in mind that although it is the starting point for this study, the original model in itself is also prone to uncertainty.

2.1.2. Metamodeling

In hydrology, a model is a simplified representation of a complex system (Clarke, 1973). A heavy computational load is often caused by expensive (in time and cost) analysis and simulation processes to reach a comparable level of accuracy as physical testing data, a challenge for which metamodeling is often used (Gary Wang and Shan, 2006). Keeping these metamodels simple facilitates the involvement of stakeholders in the modeling process, the communication of associated uncertainty, and improves the credibility of its results, as recognized by Basco Carrera et.al (2018). Moreover, their results need not fully correspond to an operationally large discrepancy in data obtained from field observations, and so models can be retained as working hypotheses (even though the data do not agree with them) so as long as a more acceptable model cannot be found (Clarke, 1973).

A metamodel is constructed from the original model with the purpose to increase efficiency while retaining sufficient accuracy. The model is developed using SEAWAT, which is MODFLOW based and simulates 3D variable-density groundwater flow coupled with multi-species solute transport (Langevin and Guo, 2006). It is used to create cells that interact in a 3D grid with a certain cell size, each representing a pore volume in a soil. These cells interact as water flows from one cell to another, based on groundwater flow equations, boundary conditions and cell parameters (horizontal and vertical permeabilities, heads, salinity) per timestep. This will be further discussed in Section 2.1.5.

As defined in Section 1.3, in this study, model accuracy is determined by the discrepancy between the original model and the metamodel, not the metamodel with direct observations or data, although the original model has been calibrated on observations (as discussed in Section 2.1.1). The original model's input and output is used as a starting point for the metamodel's calibration (its exact scripts and script structure were not accessed). An iterative approach is required to ensure the efficiency of a metamodel, whilst still retaining sufficient accuracy for the purpose of the original model: fresh groundwater resource management. Here, the discrepancy between the results of the metamodel and the original model can be quantified and analyzed statistically. Based on an analysis of the discrepancies between the metamodel and the original model, the metamodel can be justified to be sufficiently calibrated to represent the state-of-the-art original model.

Subsequently, the metamodel's purpose is to assess the potential of HyFA and the fresh-saline interface as model validation techniques, this will be further discussed in Chapter 3.

2.1.3. Regridding

Increasing efficiency of a model by decreasing calculation time is one of the sub-objectives of this study. Decreasing the number of equations to be solved in the system of equations could increase model efficiency by increasing horizontal cell sizes but maintaining the same model domain. The number of cells decreases, and consequently the number of times the equations in the system of differential equations need to be solved, decreases. The hypothesis is that regridding to a larger cell size reduces the calculation times (if all other features remain unchanged). This study investigates the effect of changing all model parameters and boundary conditions from a horizontal grid size of 25x25m to a grid size of 250x250m.

This principle is applied in this study to the initial parameters that cells represent in the model domain: groundwater salinity and hydraulic conductivities (vertical and horizontal). These parameters change during the simulation, as a result of variable density flow, as incorporated in SEAWAT. Initial groundwater salinity is regridded to a larger cell size by taking the mean regridding method: the mean salinity of the original cells that the corresponding larger cell represents, is used as starting salinity for the replacing larger cell. This method is determined in an aim to retain the total groundwater salinity.

For vertical hydraulic conductivity, the harmonic mean is used for rescaling one-dimensional, parallel resistances represented by the vertical hydraulic conductivity k_v. For horizontal flow, the geometric mean is applied for rescaling serial resistances along cells, represented by the horizontal hydraulic conductivity k_h. The geometric mean is an efficient method to rescale cell blocks and consequently obtaining single values of model parameters in geohydrology (instead of a ranges of values), that results in satisfactory representative groundwater flows (Bierkens, 1994). It is used to replace horizontal conductivities of a smaller cell size into one hydraulic conductivity. This method is integrated in the regridder function in iMOD Python (Deltares, online consultation).

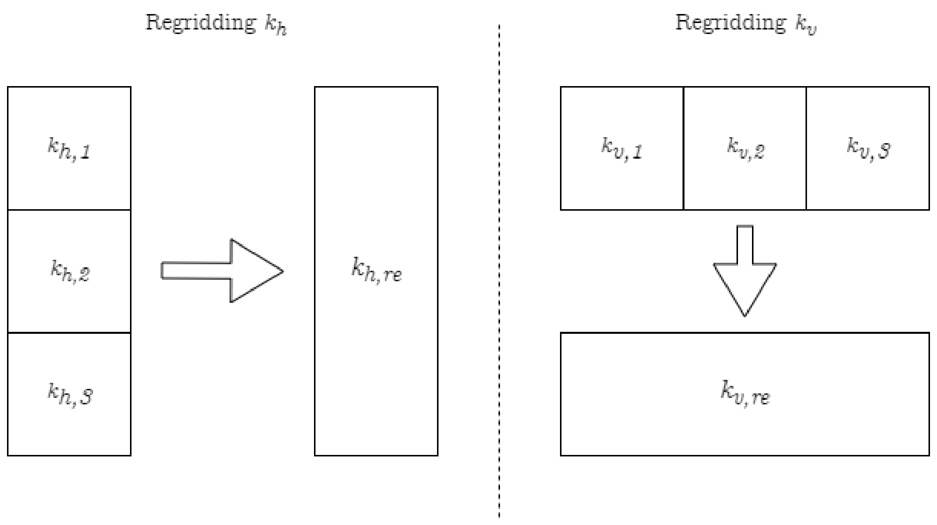


Figure 6: Conceptualization of upscaling (regridding) hydraulic conductivities to a new cell size: using the geometric mean for the horizontal direction (left, top view); and the harmonic mean for the vertical direction (right, side view).

When regridding, hydraulic conductivities that govern the intercell groundwater flow, should also be accounted for. Figure 6 shows two simplified cases for regridding cell block sizes to a larger size and letting the cell parameters change correspondingly. Replacing 3 cell parameters with one cell parameter corresponds to 3 groundwater flow differential equations to be replaced by one.

$$\sum_{i=1}^{3} Q_i = Q_{re},$$

Where Q_i represents individual cell flows in the initial cell discretization (L^3 T^{-1}) and Q_{re} represents flow through the replacing regridded cell (L^3 T^{-1}). To increase efficiency, whilst retaining accuracy, the regridded flow should be equal to the initial flow when the hydraulic head gradient is constant. The relation that should be satisfied becomes:

$$\nabla h * \sum_{i=1}^{3} k_i = \nabla h * k_{re}$$

Rearranging gives:

$$k_{re} = \sum_{i=1}^{3} k_i$$

In the case described above, the upscaled, regridded hydraulic conductivity should be the sum of all hydraulic conductivities it replaces, to represent the same flow of groundwater. A geometric mean is the most robust method to rescale hydraulic conductivities (Bootsma et. al., 2021). As the number of cells decrease, the model becomes a more simplified representation of a physical process, which may come along with the risk of losing accuracy. The effect of horizontal cell size on model calculation times is investigated (by changing from a cell size of 25m for the original model to 250m for the metamodel) but changing the vertical cell size and using variable horizontal cell size throughout the domain are outside the scope of this study.

Worth noting is that upscaling hydraulic conductivities to a larger cell size by applying the regridding method, will likely induce changes in groundwater flow. The now larger cells may cause interconnection of high conductivity zones, thereby potentially overestimating the groundwater flow, as visualized in Figure 7. In any case, regridding is expected to increase uncertainty in the metamodel output.

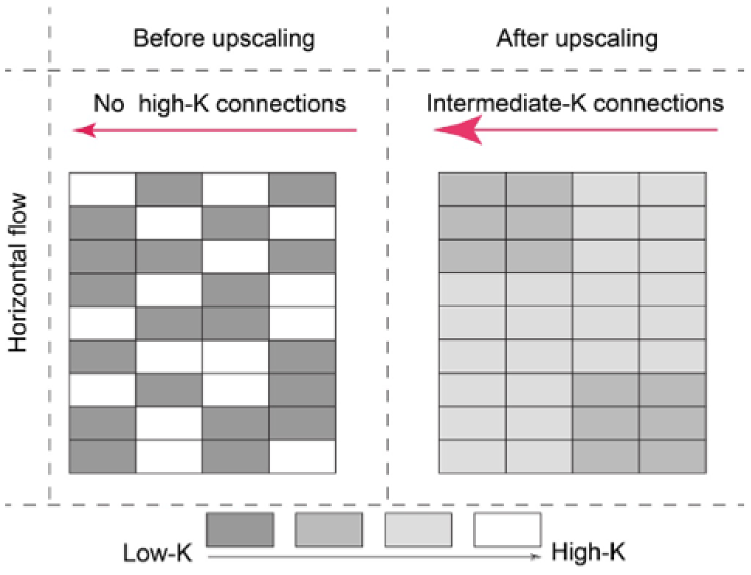


Figure 7: The effects of upscaling (regridding) horizontal hydraulic conductivities on flow paths (Yu and Michael, 2022)

2.1.4. Model calibration

As stated in Section 2.1.2, the upscaled metamodel is calibrated to the original model. This method is an iterative process: a feedback loop between data for calibration (output and input of the original model) and the metamodel (See Figure 8). Referring to the modeling approach discussed in the introduction of Section 2.1, this process focuses on accuracy of the metamodel, i.e., decreasing the discrepancy between the model outputs. The input and output data of both models are compared, the discrepancy is analyzed, subsequently changes are made to improve accuracy, and then the model is run again, this process is schematized in Figure 8. To compare the original model to the metamodel output in a clear manner, the original model input and outputs are regridded to the same cell size as the metamodel (see Section 2.1.3).

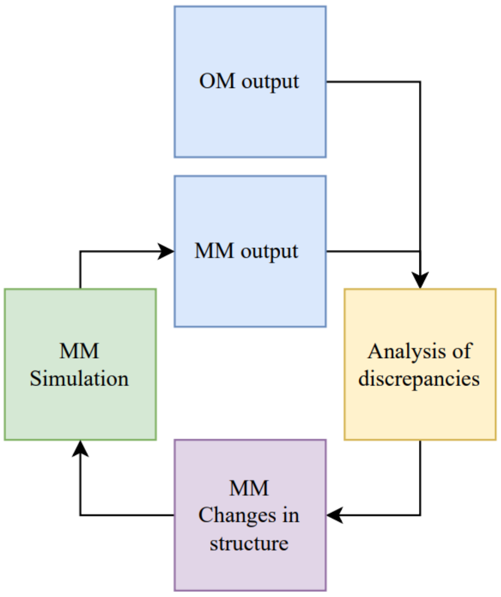


Figure 8: Conceptual model calibration procedure used for the metamodel. MM = Metamodel, OM = Original model

The process visualized in Figure 8 requires specification of “Analysis of discrepancies”: what features of the metamodel and original model should be analyzed? Typically, model calibration is done based on simulated heads and fluxes compared to field measurements, in order to have confidence in a model’s predictive capability (Hunt et. al., 2005). In this study, the metamodel calibration is done based on data from the original model instead of field observations, although the approach is the same. However, heads and fluxes may not be sufficient for model calibration (Hunt, 2002), hence other insight or other data in addition to heads and fluxes may be important for improved calibration and system understanding (Hunt, 2002; Seibert and McDonnel, 2002; Hunt, 2005).

For the first purpose of this the metamodel, i.e., to increase modeling efficiency while retaining accuracy, the calibration is done based on four “targets”: (I) hydraulic heads (steady state and transient); (II) the water balance of the study area (the dune area of Meijendel-Berkheijde where the groundwater reservoir is situated) and boundary conditions; (III) the vertical flow downward in the study area (Flow Lower Facies, FLF, see end of this section); (IV) groundwater salinity, represented by chloride as the dominant anion, see Section 3.1.1) . These targets were selected based on availability from the original model's input and output and judgement on relevant parameters for fresh-saline groundwater distributions.

To calibrate the model on a model domain scale (larger than the study area) that both the original and the metamodel share, targets I and IV are compared in a series of cross sections perpendicular to the coastline, throughout the model domain (see Figure 9). These cross sections cover the model domain and are used to inspect if the large-scale groundwater flow corresponds between the two models (for hydraulic heads) and to inspect if there are major errors in the groundwater salinity and hydraulic head outputs of the metamodel.

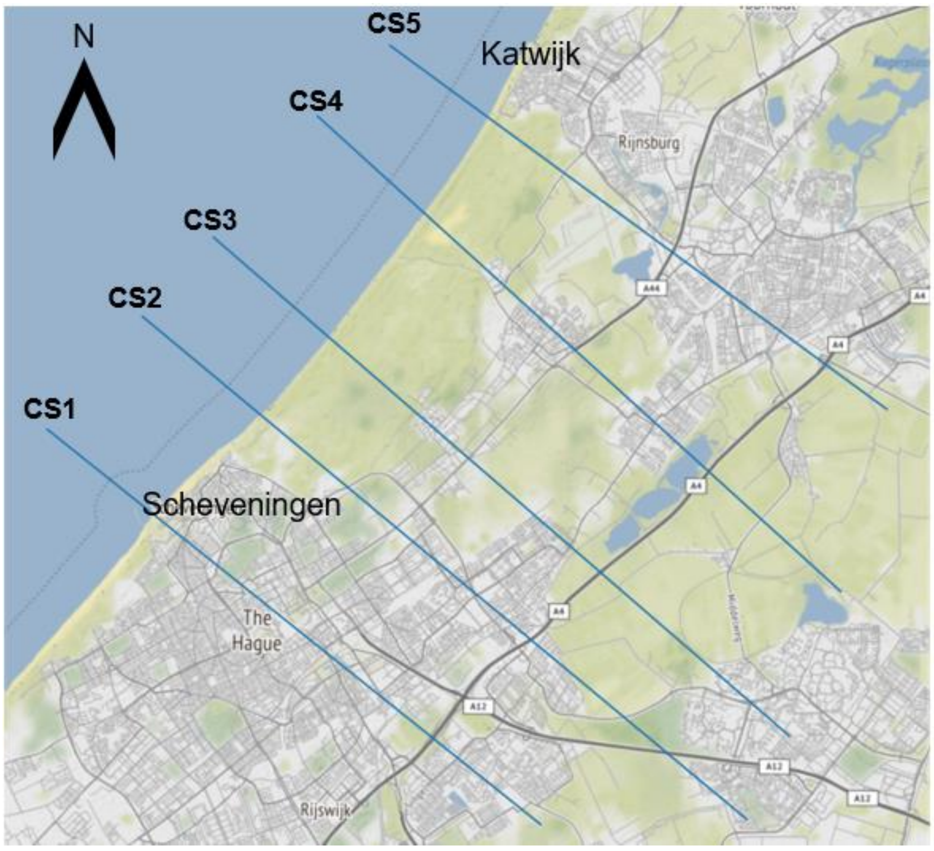


Figure 9: Extent of cross sections over the study area, to be used when analyzing the discrepancy between model outputs (See Figures 16, 19 and 39)

Besides visual inspection, the discrepancies in hydraulic head and groundwater salinity datasets between the two models will also be inspected explicitly. The discrepancy can be expressed as an error for each location i in the new (metamodel) discretization:

$$error_i = OM_{i,re} - MM_i, \qquad \text{[Equation 1]}$$

Where $OM_{i,re}$ is the original (rescaled) model output at location i and MM_i is the metamodel output at location i. The original model's output is rescaled to the metamodel cell size by taking the mean over the replacing area, using the regridding method described in Section 2.1.3.

On a smaller scale, head and flux targets may not be sufficient for constraining model parameters like (stream)bed conductance (Hunt, 2002). For this model, this applies to infiltration pond bed conductance. Therefore, calibration of the conductance of the infiltration ponds is approached through targets II and III. Calibration on water balance is done through outputs of the metamodel and original model (steady-state and transient).

Another method used to compare modeled groundwater flow in the study area is the use of Flow Lower Face budgets. In MODFLOW, these budgets define the vertical flow in or out of a cell through its lower face (L^3T^{-1}), where the convention is that upward flow into the cell is positive, and downward flow out of the cell is negative, as shown in Figure 10.

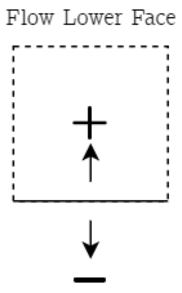


Figure 10: Convention used in iMOD, MODFLOW, SEAWAT for Flow Lower Face of cells, side view. Downward flow through the lower face of a cell is negative, upward flow is positive

2.1.5. Boundary Conditions

Boundary conditions, in hydrological modeling are used either to represent a hydrological feature, or as a computational tool (Chen, 1973) to increase calculation efficiency. In both cases, they are incorporated in MODFLOW as packages (Langevin et al., 2017) and applied to a specific cell (with specific dimensions) and time in the model domain. They are used to represent the influence of hydrological features like lakes, infiltration ponds, or the sea, on the groundwater flow. Moreover, they can be used to increase the numerical model's efficiency, while still yielding a good numerical solution (Chen, 1973). Following this logic, when making assumptions (i.e., applying boundary conditions), one is simplifying a complex system into something less complex, and should beware of an increase in uncertainty of a model's resulting prediction. Since boundary conditions apply to cells with a specific dimension, they need to be calibrated when upscaling. Scaling and defining boundary conditions is an empirical practice that involves calibration. In short: boundary conditions can increase a model's efficiency, but their negative effect on model accuracy should be mitigated as much as possible through calibration, although uncertainties are to be expected when upscaling and making assumptions.

To study the effects of boundary conditions on the metamodel's efficiency and accuracy requires looking into how they are incorporated in MODFLOW. The flow between a boundary condition cell and an adjacent cell n representing the groundwater system, is governed by a conductance term between the two, as schematized in Figure 11 for the GHB (Robin) boundary condition in one dimension. The higher the conductance, the higher the flow between the boundary condition and the adjacent cell.

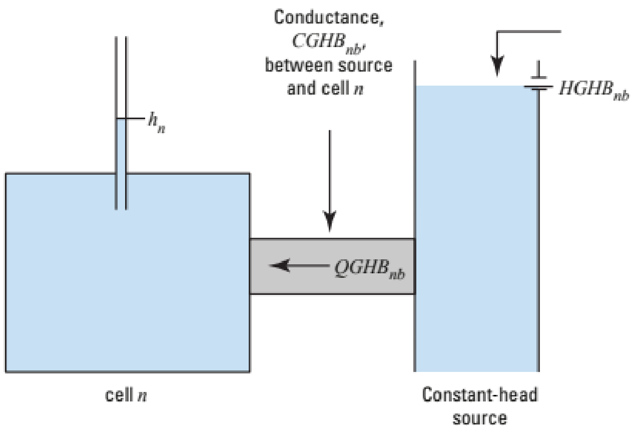


Figure 11: Diagram showing the flow between a General Head Boundary (Robin) boundary condition and its adjacent groundwater cell (Harbaugh, 2005), (Langevin et. al., 2017).

The use of boundary conditions involves empirical calibration as their use implies a simplified representation of the hydrological feature in the model. The conductance term is used to calibrate groundwater flow around the hydrological feature that the boundary condition represents. It should be recognized that use of a single conductance term to account for the resulting three-dimensional flow process is inherently an empirical exercise, and that adjustment during calibration is almost always required (Langevin et al., 2017).

The metamodel is structured to process the available data that is available from the original model. They both follow the iMOD SEAWAT package structure. Table 1 gives an overview of the packages used in this model and provides the due references.

Table 1: MODFLOW and MT3DMS packages used in SEAWAT (Langevin and Guo, 2002)

Package	Acronym	Reference
Basic	BAS	McDonald and Harbaugh (1988)
Well	WEL	McDonald and Harbaugh (1988)
Drain	DRN	McDonald and Harbaugh (1988)
River	RIV	McDonald and Harbaugh (1988)
General Head Boundary	GHB	McDonald and Harbaugh (1988)
Recharge	RCH	McDonald and Harbaugh (1988)
Preconditioned conjugate Gradient solver	PCG2	Hill (1990)
Output Control	OC	McDonald and Harbaugh (1988)
LinkMT3D	LKMT3D	Zheng and Wang (1999)
Basic Transport	BTN	Zheng and Wang (1999)
Advection	ADV	Zheng and Wang (1999)
Dispersion	DSP	Zheng and Wang (1999)
Generalized Conjugate Gradient solver	GCG	Zheng and Wang (1999)

General Head Boundary vs River package

As discussed in section 1.1, artificially enhanced infiltration in the study area commenced in the 1950s, by engineering infiltration ponds and pumping supplementary Rhine and Meuse water into the dunes. As mentioned in the introduction in this chapter, a boundary condition can be applied for cells representing infiltration ponds, rivers, and canal-and-polder areas. In this study, the difference between a fixed head (Robin) boundary condition (GHB) and a variable head and drainage condition (RIV) in their effect on model efficiency and accuracy is investigated. This is done for the infiltration ponds in the study area, as for the polder area towards the west end of the metamodel domain show in Figure 5.

When implementing a General Head Boundary (Robin boundary condition), a linear relation between the head of the adjacent cell and that of the GHB is established. The following relation is then solved in the system of differential equations:

$$QGHB_{nb} = CGHB_{nb}(HGHB_{nb} - h_n),$$

Where $QGHB_{nb}$ is the flow into cell n from the boundary expressed as a fluid volume per unit time ($L^3\ T^{-1}$), $CGHB_{nb}$ is the boundary conductance ($L^2\ T^{-1}$) and $HGHB_{nb}$ is the head assigned to the boundary condition (Langevin et. al., 2017). The flow between a GHB cell and its adjacent cell is visualized in Figure 11. In this case, intercellular groundwater flow is linearly dependent on the head of the cell, the head of the boundary condition cell and the conductance term between the two cells, as schematized in Figure 12.

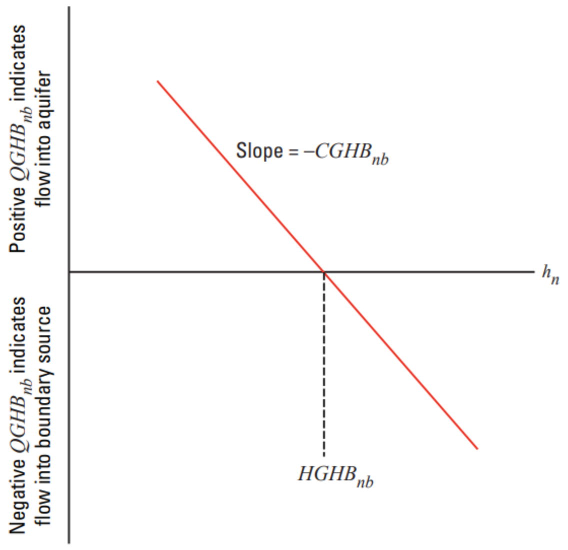


Figure 12: QGHB flow for a general-head boundary (Robin) as a function of head, h, in cell n, where HGHB is the source head and C is the conductance term for the GHB (Harbaugh, 2005), (Langevin et. al., 2017).

For a RIV boundary condition, the flow between a RIV cell and its adjacent cell is schematized in Figure 13. In this case the groundwater flow is either constant or follows a linear relation between the head of the adjacent cell and the river head. The following relation is established:

$$QRIV_{nb} = CRIV_{nb}(HRIV_{nb} - h_n), \qquad h_n > RBOT_{nb}$$

$$QRIV_{nb} = CRIV_{nb}(HRIV_{nb} - RBOT_{nb}), \qquad h_n \leq RBOT_{nb}$$

Where $QRIV_{nb}$ is the flow between the river and the groundwater system, positive if flowing out of the river (L^3 T^{-1}), $HRIV_{nb}$ is the water level (stage) in the river (L); $CRIV_{nb}$ is the hydraulic conductance of the river-aquifer interconnection (L^2 T^{-1}); and h_n is the head at the node in the cell underlying the river reach (L) (Langevin et. al., 2017). A RIV boundary condition functions as a GHB (or Robin) boundary condition when $h_n > RBOT$ and as a fixed flux (Neumann) boundary condition when $h_n < RBOT$.

As shown in Figure 13, RIV is a non-linear boundary condition that is expected to result in longer calculation times than the GHB boundary condition, as the Preconditioned Conjugate Gradient solver (PCG) requires an additional step of checking h_n to find $QRIV_{nb}$ (Hill, 1990).

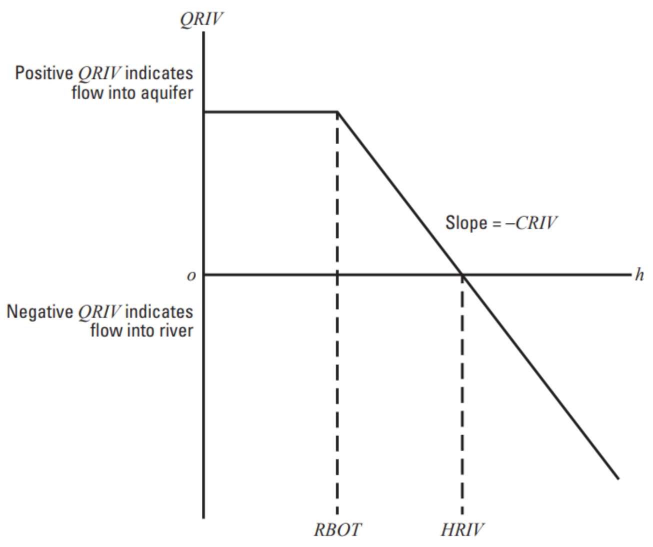


Figure 13: QRIV as a function of head, h, in the cell, where RBOT is the elevation of the bottom of the riverbed and HRIV is the head in the river (Harbaugh, 2005), (Langevin et. al., 2017).

When applying a GHB condition instead of a RIV condition, a linear relation of groundwater flow on h_{GHB} and h_n can be used to solve a differential groundwater flow equation for a specific timestep. On the other hand, when a RIV condition is used, the flow from the boundary condition into the groundwater system is either linear, or constant, depending on h_{RIV}, h_{BOT} and h_n. Therefore, the hypothesis is that the use of a GHB boundary condition for the infiltration ponds and polder areas instead of a RIV boundary condition will increase efficiency as the differential equations require less calculation time to be solved.

Drainage

The drainage package in MODFLOW is designed to simulate the effect of hydrological features that remove water from the groundwater system. (Langevin et. al., 2017). This package is used to simulate surface runoff and piezometric pipe drainage in the groundwater reservoir in Meijendel-Berkheide. This boundary condition's dependency on adjacent cell head is shown Figure 14. DRN is, just like RIV, a nonlinear boundary condition.

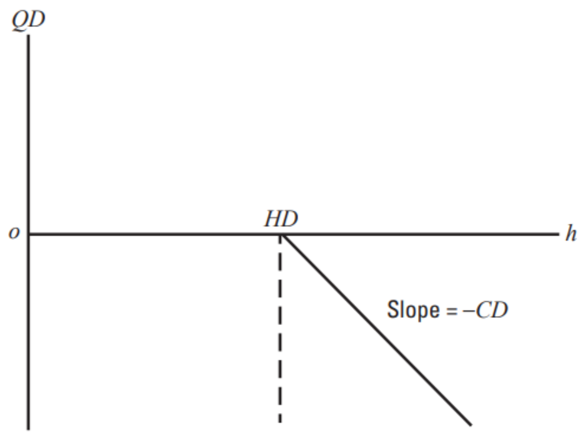


Figure 14: QDRN as a function of head in the adjacent cell. Negative QD denotes flow into the drain, out of the groundwater system (Harbaugh, 2005), (Langevin et. al., 2017).

2.1.6. Data and script structure

As stated in Section 1.3, reproducibility is valued in this study. Therefore, a detailed documentation of methods, data and scripts is provided, in an aim to ensure reproducibility of the findings of this research, and to allow for peer reviewing. The metamodel scripts can be found in Appendix D. This section describes the general data and script structure of the project and the metamodel.

Processing of the input and output data and structuring the scripts to run the model is done in Python. Scripts and data are externally stored, and version history is managed on Github (https://github.com/justkrantz/msc-thesis), where the project is organized as follows:

```
├── README.md
├── bin            <- Compiled model code can be stored here (not tracked by git)
├── data
│   ├── 1-external     <- Data external to the project.
│   ├── 2-interim      <- Intermediate data that has been altered.
│   ├── 3-input        <- The processed data sets, ready for modeling.
│   ├── 4-output       <- Data dump from the model.
│   └── 5-visualize    <- Post-processed data, ready for visualization.
├── reports        <- For a manuscript source, e.g., LaTeX, Markdown, etc., or any project reports
│   └── figures        <- Figures for the manuscript or reports
└── scripts        <- Source code for the project
    ├── 1-prepare      <- Scripts and programs to process data, from 1-external to 2-interim.
    ├── 2-model        <- Scripts to create and run model specific input from 2-interim to 3-input and
                          4-output
    ├── 3-analyze      <- Scripts to post-process model results, from 4-output to 5-visualization.
    └── 4-visualize    <- Scripts for visualization of results, from 5-visualization to ./report/figures.
```

The source code of the model follows a linear structure and (intermediate) data is stored in the manner described above. A more detailed description of the source code is provided in Figure 15, the exact scripts can be found in Appendix D.

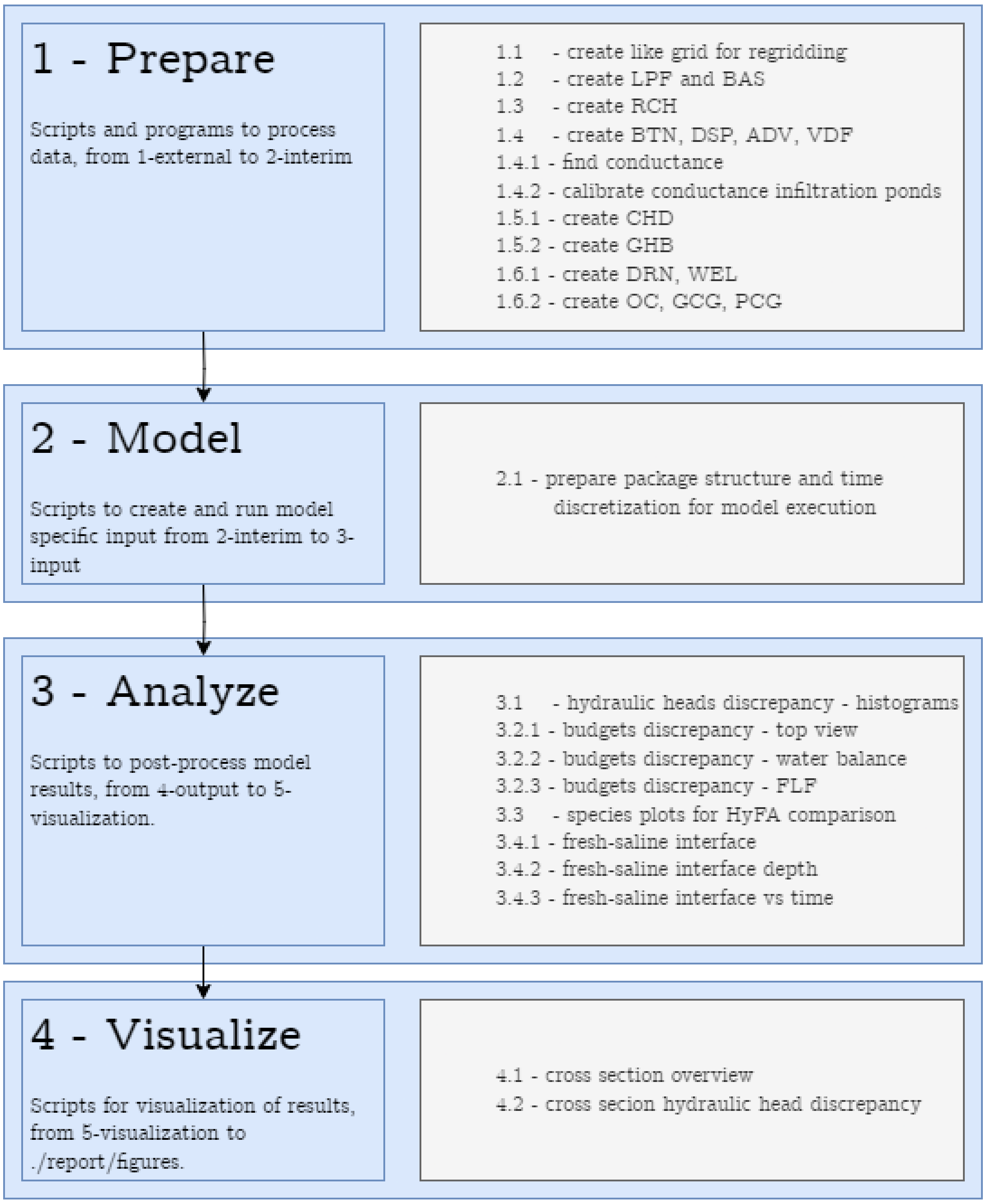

1 - Prepare
Scripts and programs to process data, from 1-external to 2-interim
1.1 - create like grid for regridding
1.2 - create LPF and BAS
1.3 - create RCH
1.4 - create BTN, DSP, ADV, VDF
1.4.1 - find conductance
1.4.2 - calibrate conductance infiltration ponds
1.5.1 - create CHD
1.5.2 - create GHB
1.6.1 - create DRN, WEL
1.6.2 - create OC, GCG, PCG
2 - Model
Scripts to create and run model specific input from 2-interim to 3-input
2.1 - prepare package structure and time discretization for model execution
3 - Analyze
Scripts to post-process model results, from 4-output to 5-visualization.
3.1 - hydraulic heads discrepancy - histograms
3.2.1 - budgets discrepancy - top view
3.2.2 - budgets discrepancy - water balance
3.2.3 - budgets discrepancy - FLF
3.3 - species plots for HyFA comparison
3.4.1 - fresh-saline interface
3.4.2 - fresh-saline interface depth
3.4.3 - fresh-saline interface vs time
4 - Visualize
Scripts for visualization of results, from 5-visualization to ./report/figures.
4.1 - cross section overview
4.2 - cross secion hydraulic head discrepancy

Figure 15: Source code structure of the model. The structure follows the overall project structure mentioned at the beginning of this section. Acronyms refer to specific packages, see Table 1. All scripts can be found in Appendix D.

It should be noted that the execution of the model (resulting from initialization in Python using the structure shown above) is done through the iMOD WQ SEAWAT executable. The executables are run on an Intel(R) Core(TM) i5-5300U CPU at 2.30 GHz, with an RAM of 8,00 GB. When calculation times are mentioned in the results section, these are dependent not only on modeling structure but also on computer performance. However, when analyzing relative changes in calculation times for model efficiency, the effect of fluctuations in computer performance is assumed to be negligible, compared to the influence by changes in modeling structure.

2.2. Results and Discussion

This section presents the metamodel output and compares it to the original model in terms of efficiency (Section 2.2.1) and accuracy (Section 2.2.2). The metamodel outputs presented in this section are the result of the iterative approach described in Section 2.1. For clarity's sake, the intermediate results owing to the applied iterative approach (Section 2.1.4) are omitted in this section and can be found in Appendix C. Note that an overarching discussion addressing the objectives of this study is held in Chapter 4.

2.2.1. Model efficiency

The effects of rescaling to a horizontal grid size of 250x250m and changing all parameters and boundary conditions correspondingly, on model calculation time is presented in this section (Section 2.1.4). The cumulative effect of the applied methods on model calculation time, and thereby efficiency, is summarized in Table 2. As discussed in Section 1.2, the original model's calculation time on a 32-core (Amazon Web Services) machine is over 2 months for a simulation of 1000 years (Bootsma, H., personal communication).

Table 2: Calculation time as a result of applied methods, for acronyms refer to section 2.1.5.

Horizontal cell size (L*W) [m]	Boundary conditions		Simulation duration [years]	Calculation time [hours : mins]	Calculation time factor [-]
	Domain edge	Infiltration ponds and polder area			
250*250	Impermeable	GHB	100	02 : 53	-
250*250	Fixed Head	GHB	100	14 : 05	4.9
250*250	Impermeable	GHB	39	01 : 33	-
250*250	Impermeable	RIV	39	03 : 55	2.5

Applying an impermeable boundary to the model domain edges is results in a calculation time that is 4.9 times faster than a fixed head boundary condition. This difference in calculation time could be related to adaptive time stepping. Adaptive time stepping is a method that allows MODFLOW to determine the appropriate step length when the solution fails to converge: the timestep is decreased and attempted to be solved again (USGS, online consultation). Consequently, this would result in a longer calculation time as the solution is now required to be solved for more timesteps.

For the infiltration ponds and the polder area, applying a GHB instead of a RIV boundary condition results in a decrease in calculation time. This may be explained by the difference between linear and non-linear equations to be solved by the Preconditioned Conjugate Gradient solver (PCG), as discussed in Section 2.1.5, nonlinearities occur if a boundary condition is nonlinear (like the RIV package) and depending on the hydraulic head in the aquifer adjacent to the boundary, the equation to be solved must be updated (Hill, 1990). In the case of the infiltration ponds, applying a linear boundary condition like the GHB could therefore been seen as an effective method to increase computational efficiency, on the condition that the effect of variations in ponds stages are small enough for the ponds to be represented by a single stage.

2.2.2. Model accuracy

In this section, the discrepancies between the original model and metamodel output are presented and discussed. As stated in Section 2.1, the model outputs are compared based on a series of cross sections for the hydraulic heads and groundwater salinity shown in Figure 9.

Hydraulic head distribution

The hydraulic head distribution of a parallel metamodel and original model simulation of 39 years can be seen in Figure 16. In CS1, CS2, CS3, the regional groundwater flow from the sea and dunes (high hydraulic heads) on the left, to the polder areas (low hydraulic head) is represented in both the original model and metamodel output. Additionally, the range of hydraulic heads visible in the model domain are of the same magnitude.

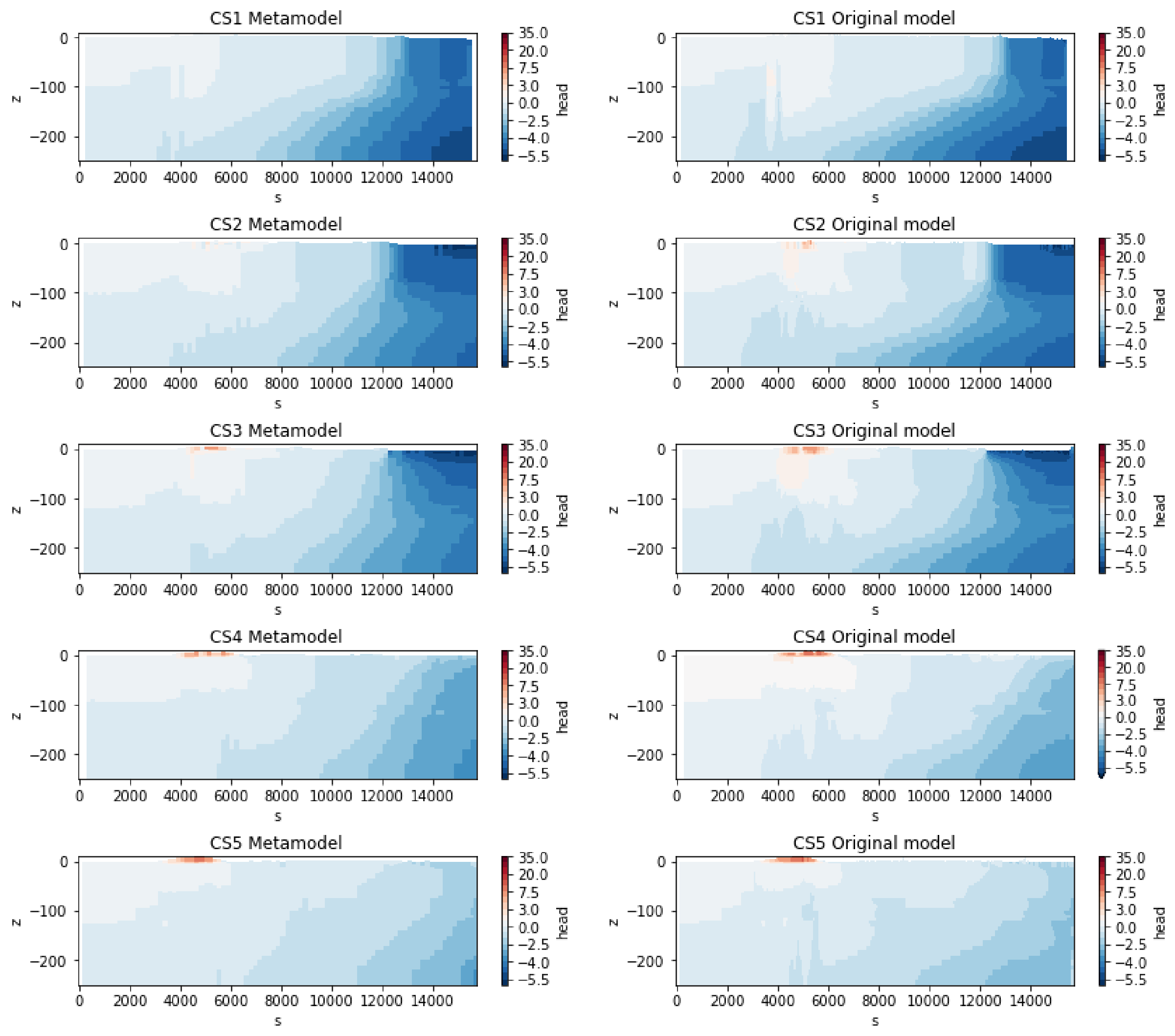


Figure 16: Cross sections (see Figure 9) perpendicular to coastline, after 40 years simulations in both the metamodel and the original model. Both simulations have the same starting heads, to investigate the discrepancy between the two. Note that the original model output has been regridded to the metamodel cells size of 250x250m for comparison.

To investigate the discrepancy between the models statistically, the hydraulic head output of the original model was regridded to the same cell size of the metamodel, after which an error could be calculated as the difference between the two (see Section 2.1.4). It should be mentioned that in this analysis, the hydraulic heads that are mentioned are point heads, as corresponding to the distinction introduced by Lusczynski (1961) between point heads, environmental heads, and freshwater heads. The point water head is the hydraulic head that would be measured in a piezometer screened at a specific point, having a certain density (Post et. al., 2007).

The global mean error on hydraulic (point) heads through the entire domain is calculated to be $\mu = \ 5.83 * 10^{-2}\ m$, with a standard deviation of $\sigma = 3.03 * 10^{-1}\ m$.

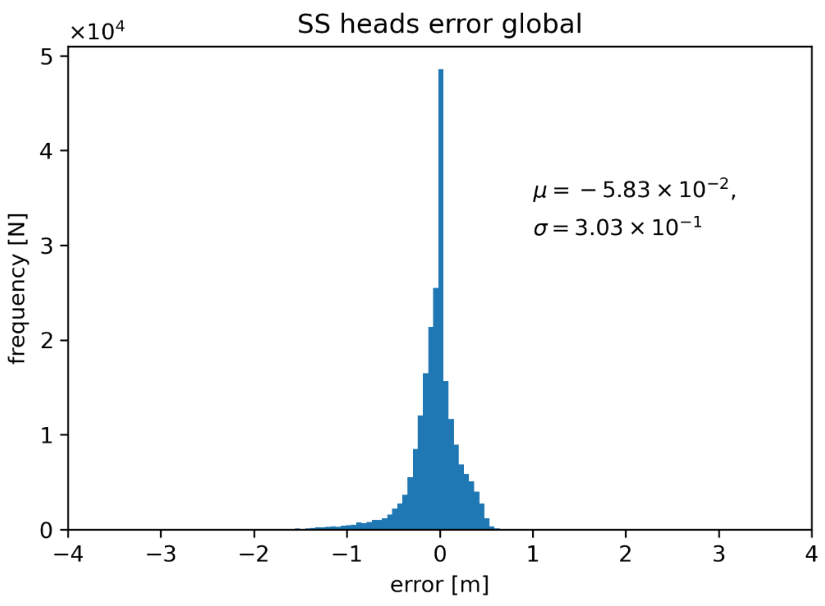


Figure 17: histogram showing the discrepancy (error) in steady-state hydraulic (point) head of the metamodel, compared to the original model.

For the study area (See Figure 2), the mean error on hydraulic heads is $\mu = \ -9.55 * 10^{-2}\ m$, and $\sigma = 6.06 * 10^{-1}\ m$. The histogram of the error between the metamodel and the original model the study is shown in Figure 17. As can be seen in the histogram, most cells of the metamodel have a lower hydraulic head than the cells at the same location in the original model. The highest largest number of cells of the metamodel have a hydraulic head that is around 0.25m lower than the original model. The positive peak in the histogram shows that the metamodel has a fraction of cells which hydraulic heads are about 0.5m higher than the heads of the same location in the original model.

In Figure 18, the discrepancy between the metamodel the original model steady-state heads is shown for the top 30m (Figure 18b) and for the entire depth (Figure 18a). It shows that the largest errors in the study area are found in the top 30m. Here, the steady-state hydraulic heads calculated by the metamodel are lower than those calculated by the original model.

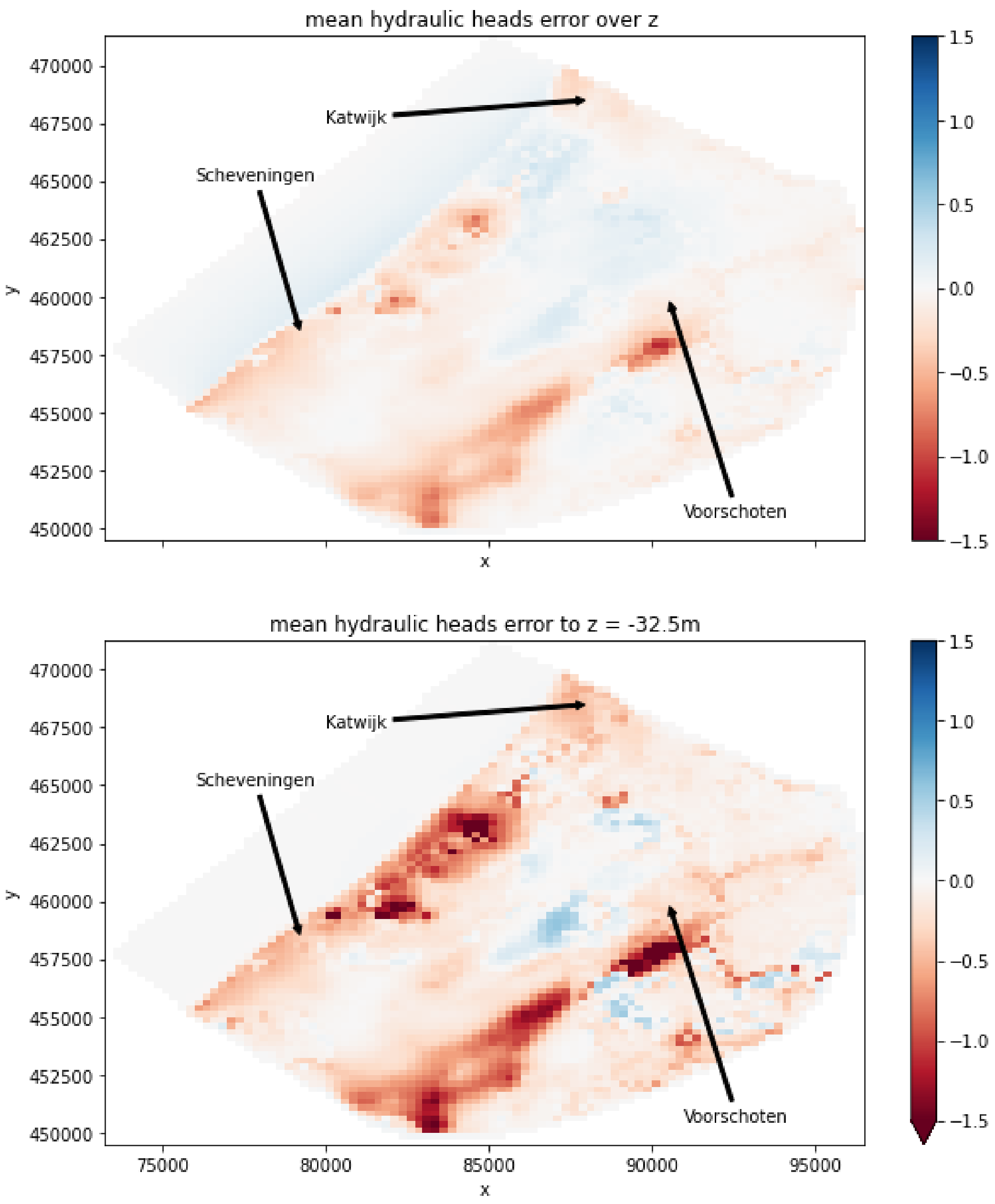


Figure 18a,b: Top view plot showing mean absolute errors over depth for the steady state hydraulic heads, of the metamodel with the original model. Mean absolute error over entire depth (top, a), and mean absolute error over top 32.5m (bottom, b). The study area lies along the coastline between Scheveningen and Katwijk.

Groundwater salinity

The groundwater salinity of the original model and metamodel are shown in parallel cross sections, perpendicular to the coastline (as described in Section 2.1.4) in Figure 19. The two cross sections that cover the study area are CS3 and CS4, as indicated in Figure 9. The results that are compared in this section, are the final groundwater salinities that the original model and the metamodel produce after a 39y simulation, when starting with the same starting salinity. In Figure 19, what can be seen is how much the calculated groundwater salinities diverge after a parallel 39y simulation. As mentioned in Section 2.1.4., the groundwater salinity is represented by the concentration of Chloride anions in the model output. Note that the original model's output was regridded to the metamodel grid size, as discussed in Sections 2.1.4.

In the cross sections, the effects of variable density flow can be seen. Generally, the fresh groundwater (blue) lies on top of the heavier saline groundwater (yellow – red), although near the coastline in CS1, CS3 and CS4, both models show a saline groundwater intrusion, below which fresh groundwater is situated. This will be further analyzed in the overarching discussion in Chapter 4.

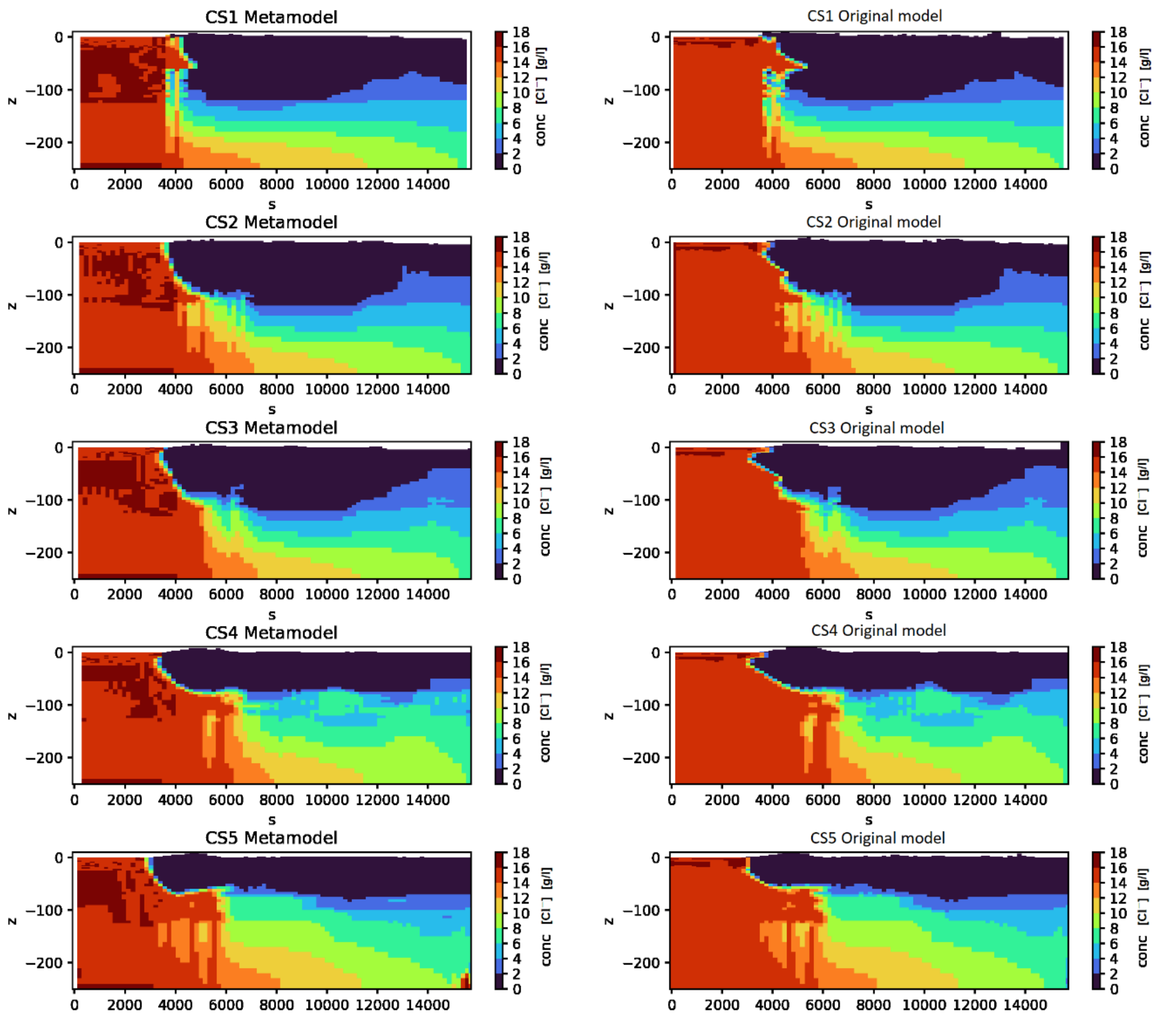


Figure 19: Final groundwater salinity of metamodel and original model after a 40y simulation, when starting with the same initial groundwater salinity, to investigate the discrepancy between the two model outputs. Note that the original model output has been regridded to the metamodel cells size of 250x250m for comparison.

Water balance

As discussed in Section 2.1.4, after the global calibration is deemed sufficient through visual inspection, the cell budgets and intercell flows are used to calibrate the conductivity below the infiltration ponds (represented as General Head Boundary condition, see Section 2.1.5) to the water balance in the study area. Figures 20 and 21 shows the water balance of the study area in the original model and the metamodel whose conductance of the infiltration ponds is calibrated to the original model (see Figure 2 for an overview study area). After six manual iterations, the conductance term below all GHB cells inside the study area in the MM have been decreased to a factor of 0.625 to match the water balances.

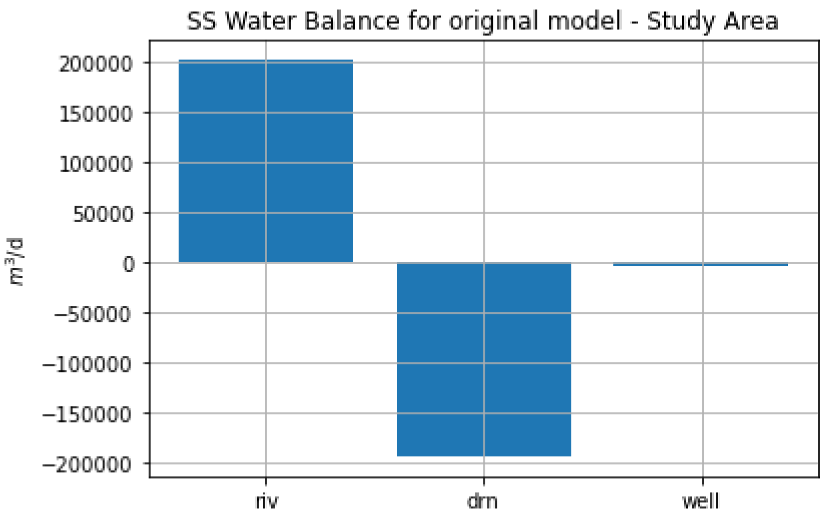


Figure 20: water balance of the study area in the original model (see Figure 2 for overview of study area)

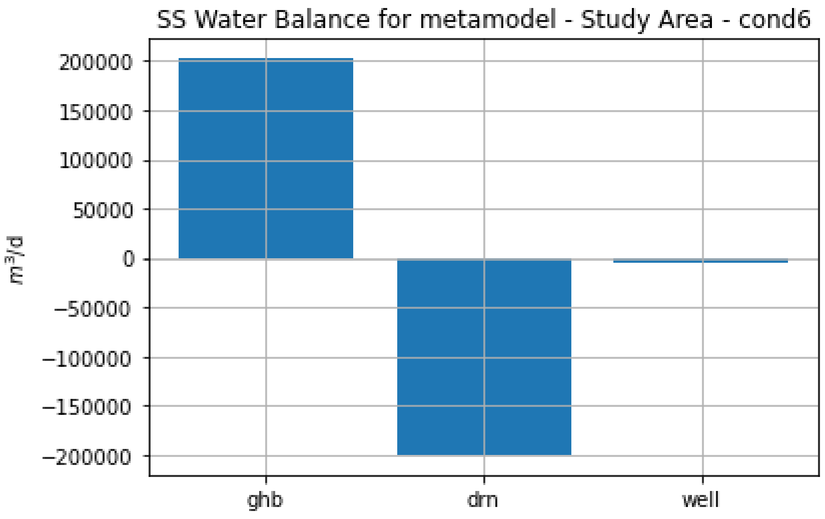


Figure 21: Steady state water balance for the study area (see Figure 2 for an overview of the study area) for the metamodel after 6 iterations on conductance (cond6). The complete calibration process is given in Appendix C.

Vertical flow through the lower face of a cell can be obtained from the model output through flow lower face budget values. As described in Section 2.1.4, the convention describes that upward flow through the lower face of a cell is positive, and accordingly downward flow through the lower face of a cell is negative. Figures 22 and 23 show the error that the metamodel has for flow lower face budget values, compared to the original model (see figure 2 for study area). Globally, the mean vertical flow downward is lower in the metamodel than in the original model. In the study area, the mean vertical flow downward is higher in the metamodel than in the original model. The standard deviation is higher in the study area than in the global model domain.

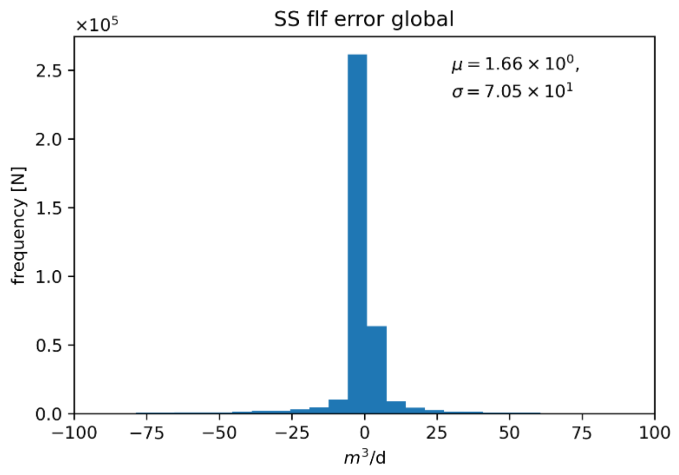


Figure 22: Discrepancy (error) between the original and metamodel's calculated Flow Lower Face (FLF) over the entire model domain. (See Figure 10 for convention of FLF)

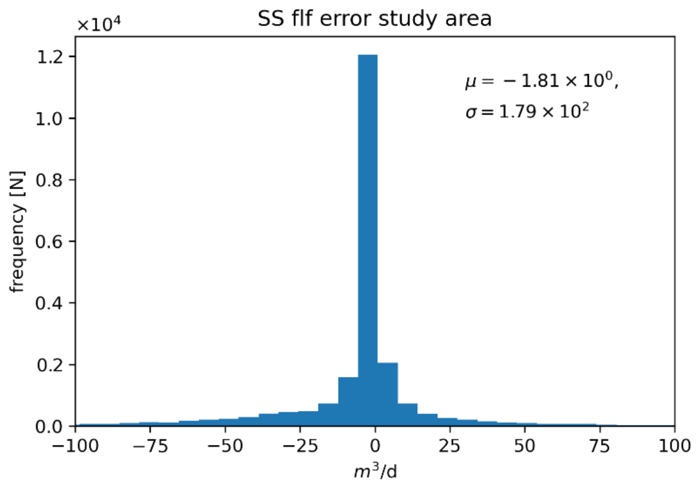


Figure 23: Discrepancy (error) between the original and metamodel's calculated Flow Lower Face (FLF) over the study area. (See Figure 10 for convention of FLF, see Figure 2 for an overview of the study area)

Domain boundaries

When changing the boundary conditions at the model domain boundaries from an impermeable to a fixed head condition, the hydraulic head error in the study area remains unchanged. Therefore, from an accuracy perspective, the model domain edges are assumed to be placed sufficiently far away to have negligible effect on the accuracy of the model.

3. Model validation on groundwater salinity

This chapter focuses on the second objective of this study: assessing conventional model validation techniques by introducing the metamodel to two new concepts: the Hydrochemical Facies Analysis (HyFA) by Stuyfzand (1993a), and the fresh-saline interface. In this chapter, the applicability of these concepts as efficient modeling practices is investigated. To do so, the methodology is described in Section 3.1 by first introducing the metamodel to the HyFA in Section 3.1.1 before applying the conceptual fresh-saline interface to the metamodel in Section 3.1.2. The results are described and briefly discussed from a modeling perspective. Note that a further, overarching discussion addressing the research objectives is held in Chapter 4.

3.1. Methodology

In Chapter 2, the metamodel's accuracy and efficiency are studied, providing a research opportunity to test conventional modeling techniques to validation on groundwater salinity. Effective model use in the future will not only be to seek solutions but also to analyze data significance and guide further collection efforts towards minimizing uncertainty in predictions (Langevin and Panday, 2012). Besides "hard data", using "soft data" (qualitative knowledge that cannot be directly used as numbers) to model validation has added value as they provide insight in the process knowledge (Seibert and McDonnell, 2002). Therefore, the ambition in this chapter is to assess the significance of two concepts that may be regarded as soft data: fresh-saline interface and the HyFA, when validating a geohydrological model in an efficient manner.

3.1.1. Hydrochemical Facies Analysis (HyFA)

As mentioned in section 1.2, the HyFA is an interpretation of field data, which provides insight in the "path" that groundwater particles travel through the soil. Potentially, this type of analysis could be valuable for model validation, especially since most data used for model validation are "snapshots". In his study (the HyFA), Stuyfzand (1993a) analyzed over 2000 samples obtained from boreholes along the coastline of the Netherlands. By sampling at multiple depths per borehole, and through interpolation and interpretation, he could visualize the origin of various groundwater regions by their chemical composition. Two hydrochemical sections by Stuyfzand (1993a, 1993b) cross the study area: one perpendicular to the coastline and one along the coastline, an overview is given in Figure 24.

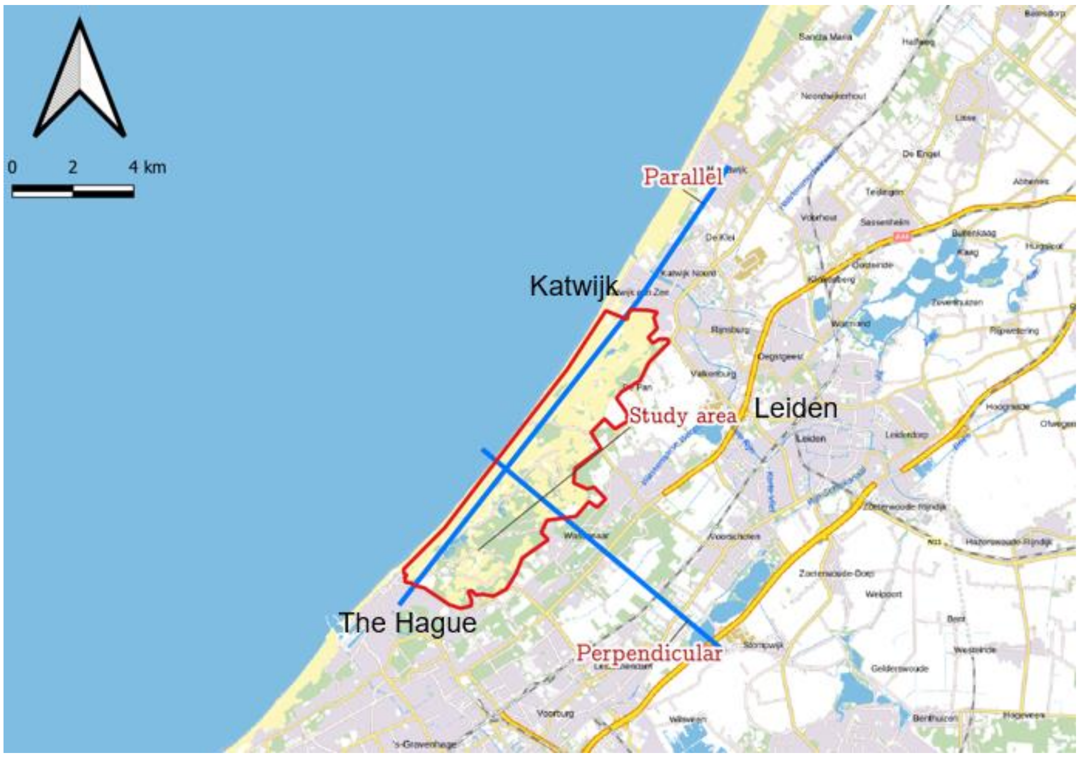


Figure 24: Overview of Stuyfzand hydrochemical cross sections (1993a, 1993b) across the study area of Meijendel-Berkheijde: Perpendicular to the coastline (Stuyfzand, 1993b) ; Along coastline (Stuyfzand 1993a)

In Stuyfzand's cross sections (1993a, 1993b), a visual representation of the geohydrology is seen based on groundwater origins. Figure 25 shows the sampling interval of the two cross sections through the study area. Figures 26 and 27 show interpreted versions of these cross sections that highlight the artificial infiltration and groundwater salinity.

Although unmentioned in his study, the horizontal sampling interval can be derived from the cross sections in Figure 25. 12 boreholes over the (conservative approximation of) a distance of 12km between Scheveningen and Katwijk, corresponds to a conservative sampling interval of 1km. For the cross-section perpendicular to the coastline (Stuyfzand, 1993b), the conservatively derived sampling interval in the study area is 500m. Worth noting is that Figure 25 only serves to highlight the sampling intervals.

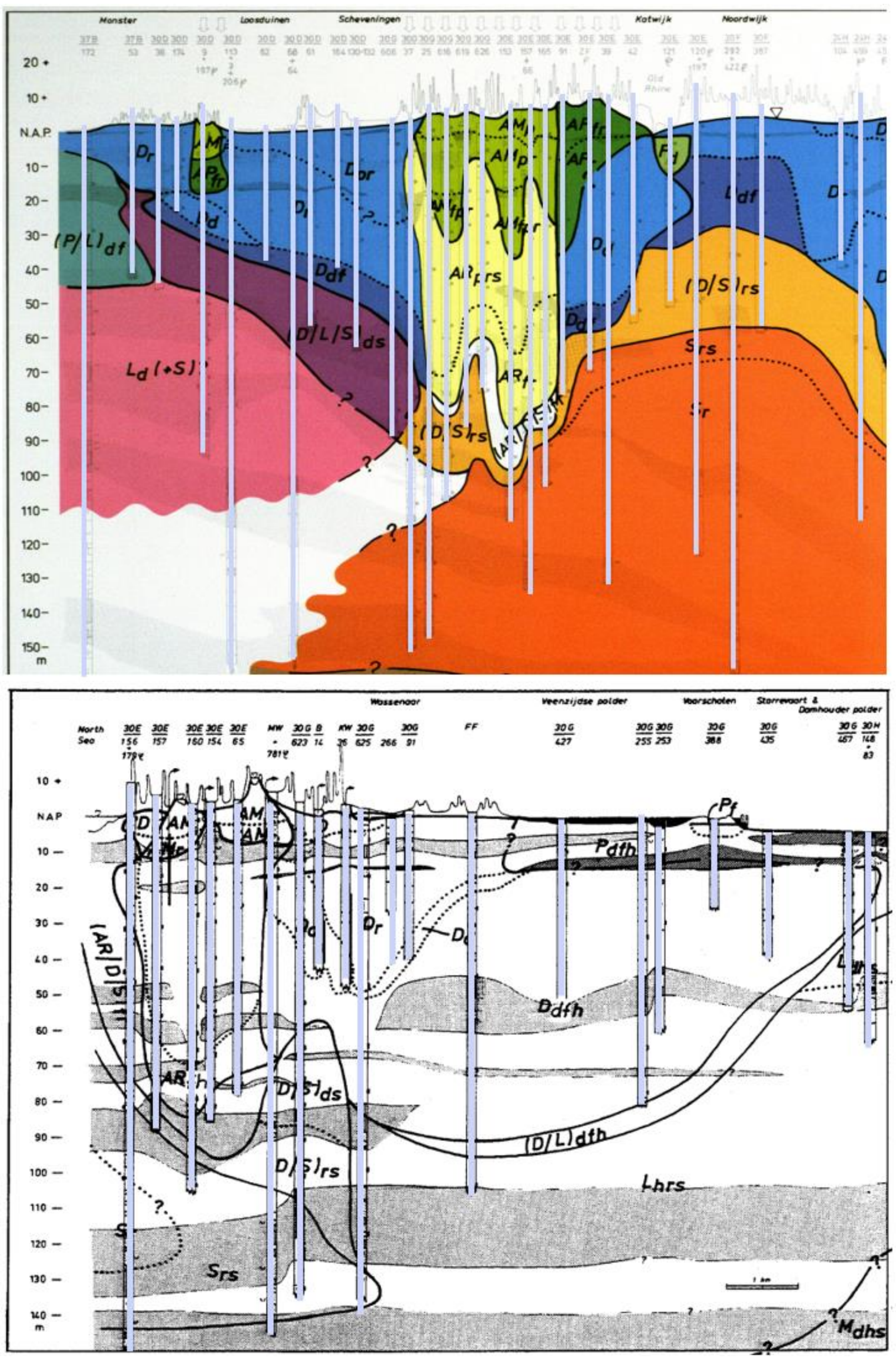


Figure 25: Sampling interval of hydrochemical cross sections. top: along coastline (Stuyfzand, 1993a); bottom: perpendicular to coastline (Stuyfzand, 1993b). The sampled boreholes are highlighted in grey. In the study area (between Scheveningen and Katwijk), the sampling interval is 1km horizontally for the cross section along the coastline and 500m perpendicular to the coastline. For the complete cross sections, see Appendix A and B. For the interpreted cross sections showing groundwater origins, see Figures 26 and 27.

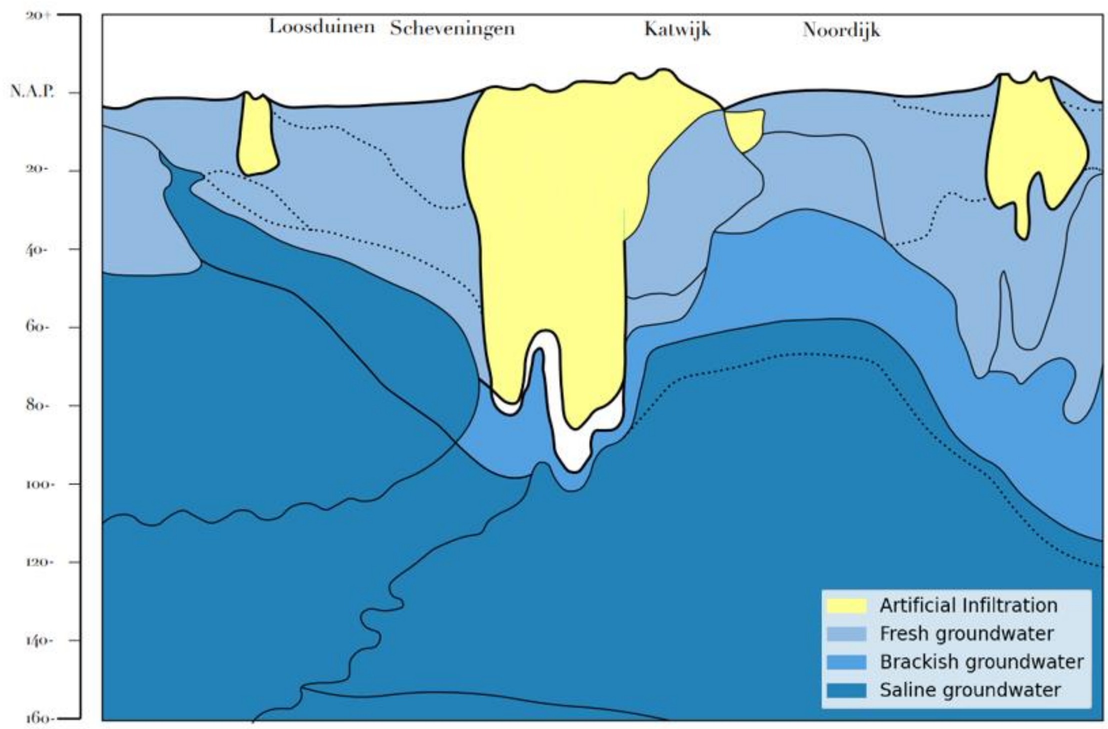


Figure 26: Cross section along coastline (see Figure 24), through the study area showing groundwater origins. Interpreted from the Hydrochemical Facies Analysis (Stuyfzand, 1993a). See Appendix A for the interpretation.

An interpreted version of the HyFA cross section along the coastline (Stuyfzand, 1993a) is given in Figure 26 and in Figure 27 the interpreted version of the cross-section perpendicular to the coastline (Stuyfzand, 1993b) is shown.

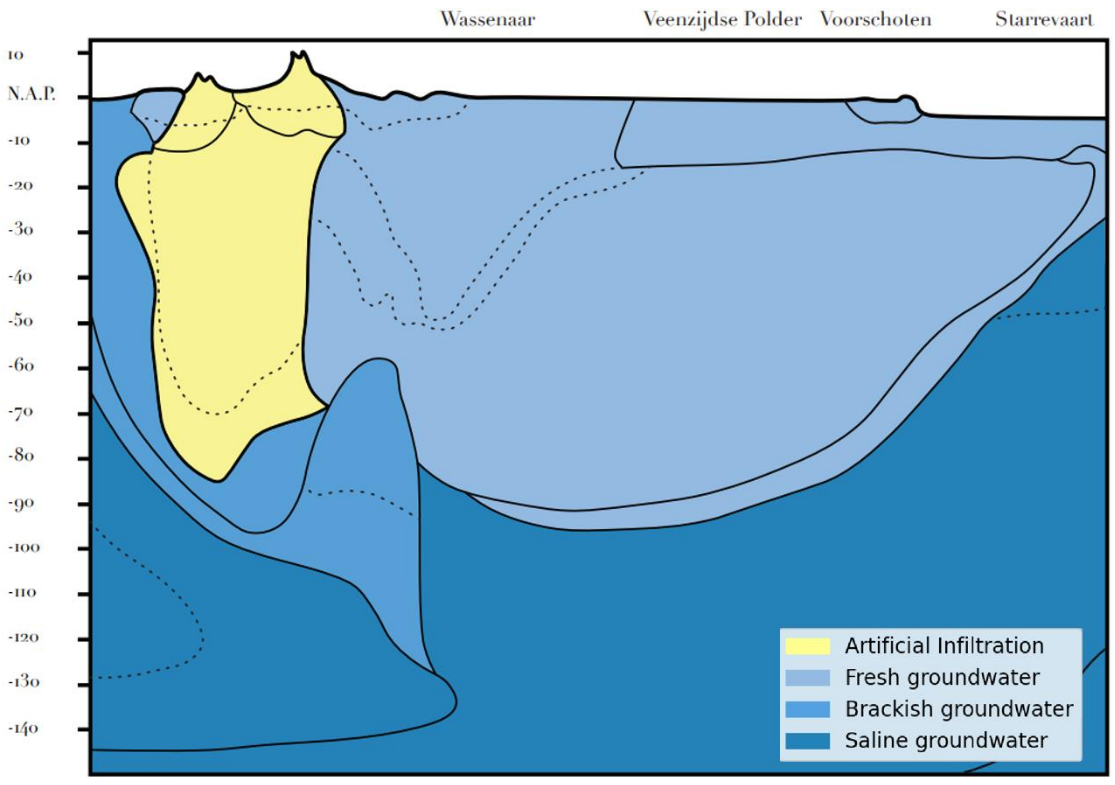


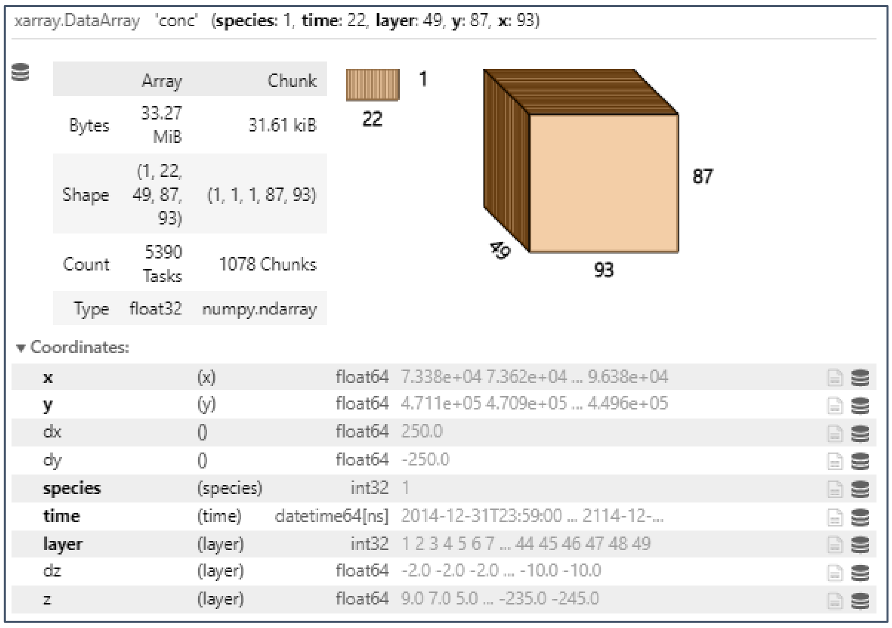
Figure 27: Cross section perpendicular to coastline (see Figure 24), through the study area showing groundwater origins. Interpreted from the Hydrochemical Facies Analysis (Stuyfzand, 1993b). See Appendix B for the interpretation.

By virtue of the HyFA, the geohydrology of the study domain is now not only subdivided by fresh, brackish, and saline groundwater, but also fresh, artificially infiltrated groundwater is shown. For clarity's sake, the cross sections by Stuyfzand (1993a, 1993b) were interpreted using different colors and showing relevant features only, see Appendix A and B for the respective original cross sections and their legend.

From a modeling perspective, it is crucial to realize that these cross sections are prone to uncertainty. An interpolation and interpretation is necessary to create a visual description of the geohydrology of Meijendel-Berkheijde from the boreholes with a sampling interval of 500-1000m. This interpolation and interpretation generates uncertainty, this should be kept in mind when using the HyFA for model validation.

Implementation in the metamodel

The HyFA is used for model validation by implementing origin tracers as species in MODFLOW SEAWAT. In the metamodel, datasets are managed in Xarray (2023). To incorporate artificial infiltration as a species, the species dimension is added to the GHB boundary condition (or Robin boundary condition - see Section 2.1.5) in the study area, representing the infiltration ponds in the metamodel. Figure 28 shows the "conc" data array, that every active groundwater cell contains, in which not only groundwater salinity is defined, but also a dimension "species" for tracing artificial infiltration is added. This approach mimics hydrochemical origin tracing as performed in the HyFA.


```
xarray.DataArray  'conc'  (species: 1, time: 22, layer: 49, y: 87, x: 93)

              Array          Chunk
Bytes         33.27 MiB      31.61 kiB
Shape         (1, 22, 49, 87, 93)   (1, 1, 1, 87, 93)
Count         5390 Tasks     1078 Chunks
Type          float32        numpy.ndarray

▾ Coordinates:
   x          (x)          float64          7.338e+04 7.362e+04 ... 9.638e+04
   y          (y)          float64          4.711e+05 4.709e+05 ... 4.496e+05
   dx         ()           float64          250.0
   dy         ()           float64          -250.0
   species    (species)    int32            1
   time       (time)       datetime64[ns]   2014-12-31T23:59:00 ... 2114-12-...
   layer      (layer)      int32            1 2 3 4 5 6 7 ... 44 45 46 47 48 49
   dz         (layer)      float64          -2.0 -2.0 -2.0 ... -10.0 -10.0
   z          (layer)      float64          9.0 7.0 5.0 ... -235.0 -245.0
```

Figure 28: The Xarray data array "conc", containing the concentrations of Cl^- to represent groundwater salinity. The "species" dimension is defined, by which a flag can be added to the artificially infiltrated water through the infiltration ponds (the GHB boundary condition, see Section 2.1.5).

The model simulation time is set to 40 years, which is by approximation the relative time between commencing artificial infiltration in the 1950s (Stuyfzand, 1993a) to the time when Stuyfzand (1993a) commenced data acquisition through boreholes in the early 1990s. In addition to the origin and path of groundwater, the representations by Stuyfzand (1993a, 1993b) also provide information on the rate at which artificial infiltration is achieved.

3.1.2. Fresh-saline interface

The fresh saline interface implemented in groundwater modeling by Oude Essink (2001) serves as a conceptual tool to investigate the shape and extent of the freshwater floating on the saline groundwater, acknowledging that there is a transition zone (or "brackish water zone") between the fresh groundwater and the heavier saline groundwater (Stuyfzand, 1993a). Following the knowledge of Seibert and McDonnell (2002), it may be seen as "soft data". In this section, this concept will first be used to construct an argument that questions the validity of results of variable density groundwater models that are uncalibrated on groundwater salinity data. Furthermore, analyzing the original model's and metamodel's groundwater salinity outputs may give new insights into the discrepancy between the two models. Finally, the fresh-saline interface may give insight in the development of the recharge of the fresh drinking water reservoir in the study area.

In a dialog between experimentalists and modelers in hydrology, using only hard data in a single-criterion model comes with a risk of achieving "right answers for the wrong reasons" (Seibert and McDonnell, 2002). As discussed in section 1.2, the original model used by Dunea for assessing the feasibility of brackish groundwater as a groundwater resource, has only been validated on hydraulic head data and extraction water balance. The original model has not been calibrated on the long-term development of the groundwater salinity, though the model does incorporate variable density flow equations.

For the sake of argumentation, validation solely on hydraulic head data can be conceptually represented in Figure 29b, where it can be seen that there is only information on the phreatic groundwater surface level. Calibration of a fresh-saline groundwater flow model only on hydraulic head data may be right for the wrong reasons, as it doesn't address fresh-saline groundwater regions like the conceptual fresh-saline interface shown in Figure 29a.

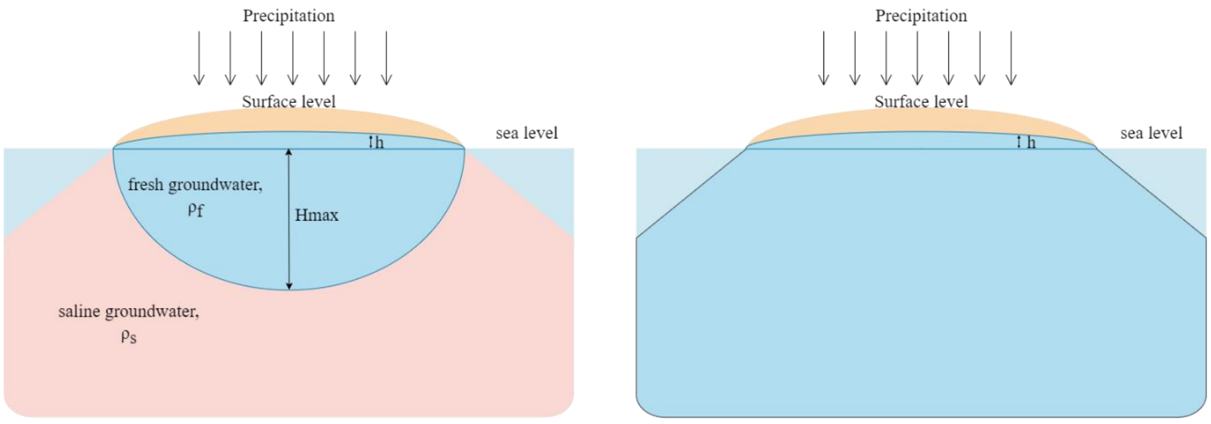


Figure 29a,b: conceptual representation showing: (a, left) the fresh-saline interface of an island surrounded by sea water, with precipitation on the surface; (b, right) The same island without representation of groundwater salinity, only the phreatic surface. h = hydraulic head with respect to sea level, Hmax = maximum depth of freshwater lens with respect to mean sea level. ρ_f = fresh groundwater density, ρ_s = saline groundwater density.

Groundwater salinity is represented in the model with chloride, as it is the dominant anion in Dutch coastal groundwater and density is linearly related to it within naturally occurring concentrations (Delsman 2015). The fresh-saline interface is interpreted as the interface between fresh and brackish groundwater, where groundwater is subdivided in terms of salinity corresponding to Stuyfzand (1993a, 1993b):

$$\begin{array}{lll} \textit{Fresh groundwater} & \text{for} & [\mathrm{Cl}^-] \leq 300 \ \frac{mg}{L} \\ \textit{Brackish groundwater} & \text{for} & 300 \ \frac{mg}{L} \leq [\mathrm{Cl}^-] \leq 10 \ \frac{g}{L} \\ \textit{Saline groundwater} & \text{for} & [\mathrm{Cl}^-] \geq 10 \ \frac{g}{L} \end{array}$$

Since groundwater salinity is provided in the original model output, these data can be processed to visualize the fresh-saline interface. Therefore, it can be used to compare the original and the metamodel outputs. In a technical sense, the fresh-saline interface is defined as the minimum depth at which groundwater becomes brackish. Because the model domain is structured as a 3D grid, calculating the fresh-saline interface over depth can be seen as a reduction into a 2D fresh-saline interface dataset.

However, it should be stressed that the original model has not been calibrated on groundwater salinity. From the perspective of this section, the discrepancy will therefore not be expressed as an “error”, rather serve to shed a new light on the discrepancies and test its consistency. This discussion will be continued in the overarching discussion in Chapter 4.

To summarize: the fresh-saline interface is used as a tool to: (I) study if model validation on hydraulic head data only, is sufficient for a variable density groundwater flow model; (II) it can be used to shed a new light on the discrepancies between the original model and the metamodel; (III) it may give insight in the rate by which the drinking water reserve in the study area recharges.

3.2. Results & Discussion

The potential of the HyFA and the fresh-saline interface for model validation will be further investigated in this section. Discrepancies in fresh-saline interface behaviour between the original model and the metamodel are discussed in this section and will be further addressed an overarching discussion in Chapter 4. The interpreted HyFA is compared to the model outputs in Sections 3.2.1 and 3.2.2. For the HyFA, the two cross sections which are considered from Stuyfzand (1993a) and (1993b) are shown in Figure 30 below.

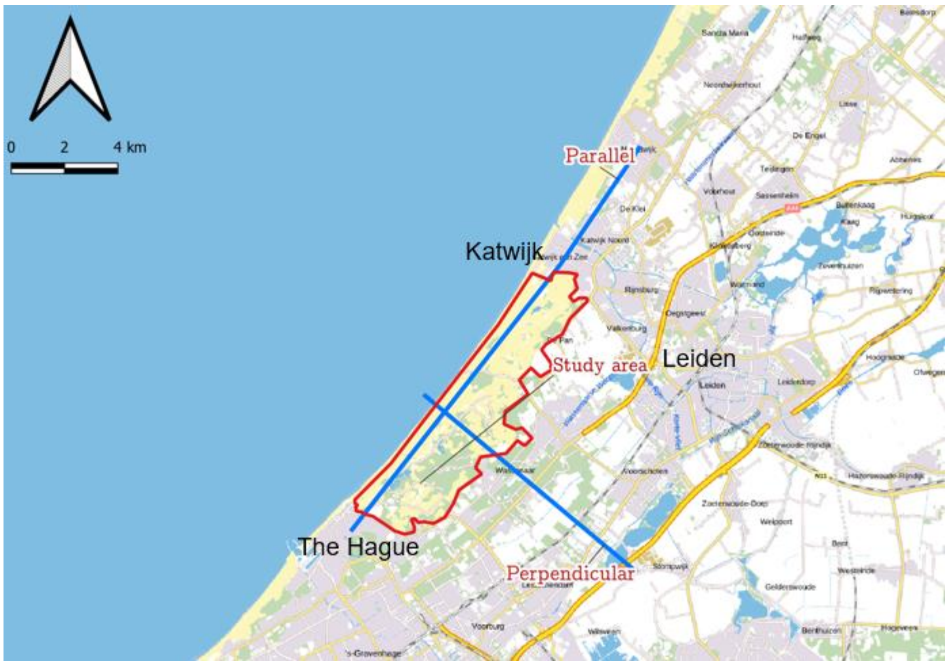


Figure 30: overview of cross sections used to compare the HyFA with model origin tracers (species). In red, the study area can be seen (copy of Figure 24, shown here for quick reference).

3.2.1. HyFA: Cross section along coastline

Figure 31 shows how the origin tracers in the metamodel compare to the HyFA in the cross section along the coastline. The depths to which the artificially enhanced infiltration reaches is about 80m. The brackish water zone is about 20m depth in both. The shape of the artificially infiltrated groundwater by the metamodel differs from that of (Stuyfzand, 1993a), as there is a region of “native” fresh water in between two artificially infiltrated water bodies. The horizontal extent corresponds globally, although the overall shape varies between the two.

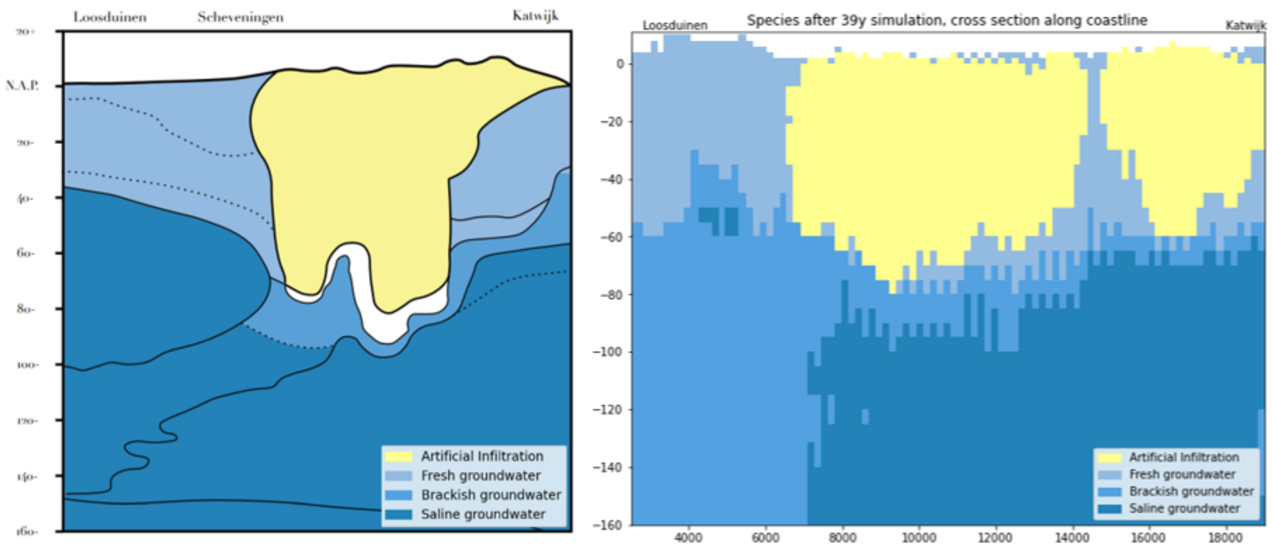


Figure 31: Left: Cross section with origin tracers along coastline interpreted version of cross section by Stuyfzand (1993a), see Appendix A for original) ; Right: Modeled origin tracers by metamodel (Right after a 40y simulation)

3.2.2. HyFA: Cross section perpendicular to coastline

Figure 32 shows the groundwater origins on cross sections perpendicular to the coastline, with the HyFA. The depth to which artificial infiltration reaches in 40 years is slightly lower in the metamodel output (-60m) , compared to the HyFA (-80m). The horizontal extent and global shape of the artificially infiltrated groundwater are represented by the metamodel. The shape of the fresh-brackish and the brackish-saline interfaces in the metamodel coincide globally with those in the HyFA. In the HyFA, saline groundwater is situated below the fresh groundwater along the entire cross section, land inwards. In the modeled origin tracers, this is not represented.

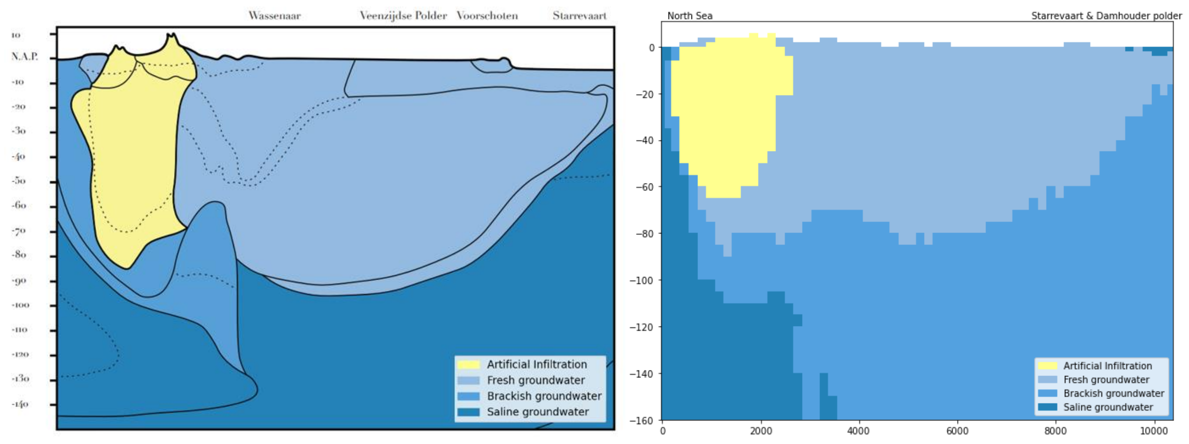


Figure 32: Left: Cross section perpendicular to coastline as interpreted from Stuyfzand (1993b), see appendix B for original)- Right: Modeled groundwater origin tracers using species. (see Figure 30 for overview of cross sections for model species)

3.2.3. Fresh-saline interface

In Figure 33, the depth of the fresh-saline interface is shown (see section 3.1.2 for fresh-saline interface). The metamodel fresh-saline interface has a comparable overall shape, relatively deep near Scheveningen and shallower further inland at the edges of the model domain. Along the shoreline, the metamodel has a deeper fresh-saline interface than the original model. The largest errors can be seen near Voorschoten (60m higher in the metamodel than in the original model output) and near the southern edge of the domain, which South of The Hague. It should be noted that these errors lie outside the study area (along the coastline between Scheveningen and Katwijk).

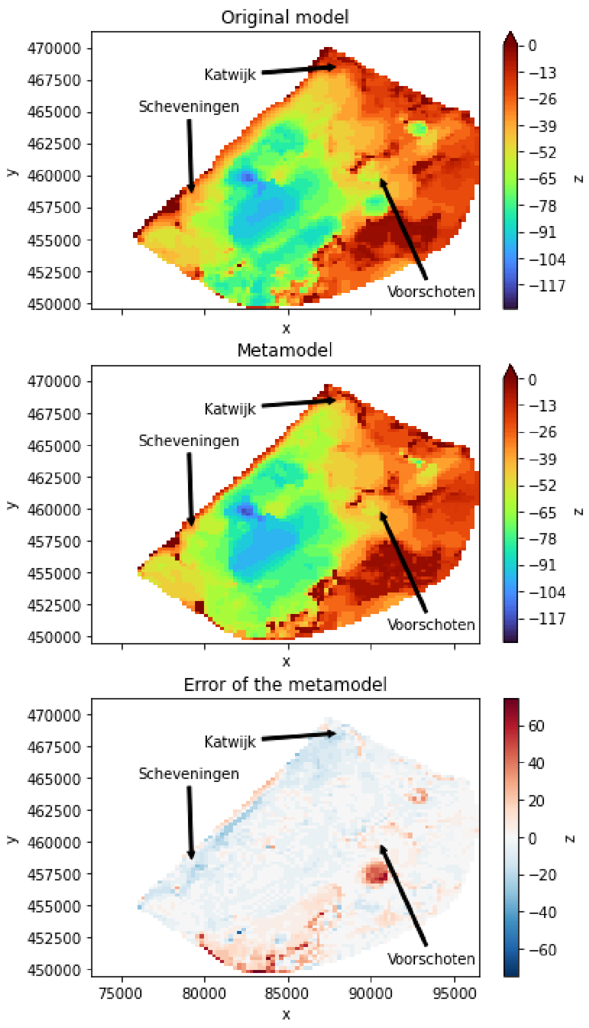


Figure 33: Fresh saline interface after parallel 40y simulations, to investigate the discrepancy between the original model (top, a), metamodel (middle, b), and the error of the metamodel with the original model (bottom, c). The Study area is located between Scheveningen (South) and Katwijk (North)

For the study area (see Figure 2), the original model and metamodel's fresh-saline interface development over time can be seen in Figure 34. As defined in Section 3.1.2, the fresh saline interface is the minimum depth at which groundwater becomes brackish, reducing the model salinity data to a 2D dataset. Taking the average over the study area and plotting vs time results in the representation shown in Figure 34.

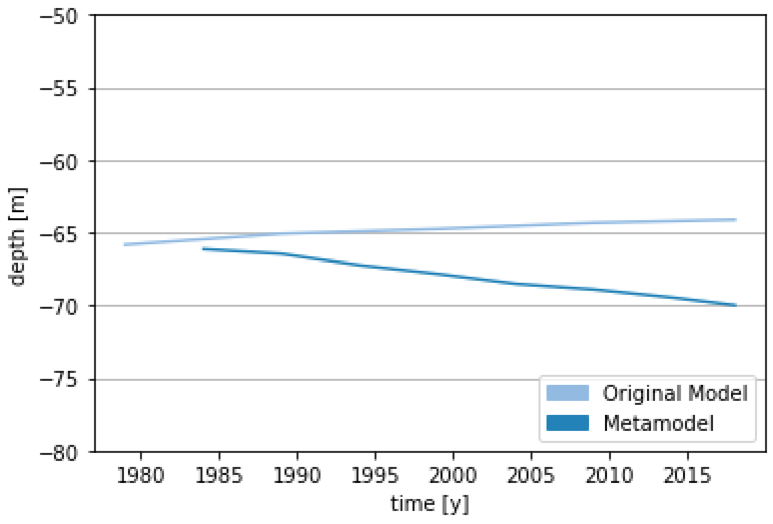


Figure 34: Development of fresh-saline interface in the study area over two parallel 40y simulations, to investigate the discrepancy between the original model and metamodel, when both starting with the same initial groundwater salinity.

The original model fresh-saline interface becomes less shallow in 40 years and moves to an equilibrium at around -63m, whereas the metamodel shows an increase in depth and continues to increase more in depth at -70m after 40 years, when subjected to the same fluxes.

Extending the metamodel's simulation time to 200y resulted in the development of the f/s interface shown in Figure 35. As can be seen, the f/s interface has not reached an equilibrium depth after a 200y simulation.

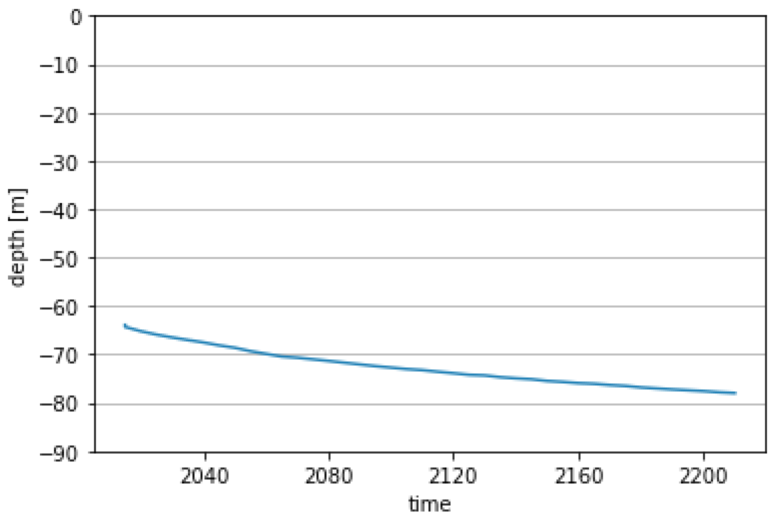


Figure 35: Development of the metamodel's calculated fresh-saline interface over 200y simulation, starting with the same groundwater salinity as all other simulations.

4. Discussion & Conclusion

In the preceding chapters of this thesis, modeling of coastal groundwater reserves has been approached from several perspectives. In Chapter 1, the motivation for this study and the global scientific relevance which it aims to achieve are discussed. Subsequently, two major concepts area appointed and addressed in Chapters 2 and 3, both focusing on one aspect of the problem described in Chapter 1. Chapter 2 concentrates on contribute to efficient modeling practices through metamodeling, where Chapter 3 focuses on the potential of scarcely used validation techniques for groundwater models. This chapter serves as an overarching discussion and conclusion, where the perspectives of Chapters 2 and 3 are reunited to contribute to address the research objectives defined in Section 1.3.

First, a more thorough analysis of the discrepancies between the original model and the metamodel is required, after introducing the metamodel to both the fresh-saline interface and the HyFA. This is presented in Section 4.1. Subsequently, the two major concepts of this study: efficient modeling practices and model validation techniques (Section 4.2) are discussed, on which the conclusions (Section 4.3) are founded. Finally, recommendations for further research are given in Section 4.4.

4.1. Analysis of discrepancies

4.1.1. Flow lower face (FLF)

The metamodel's higher downward flow in the study area (Figure 23) and its fresh-saline interface increasing in depth, compared to the original model hint at a consistency between these two results. To investigate this, the steady state FLF budget errors and the hydraulic heads errors are more closely analyzed for the study area.

The FLF budget error shown in Figure 23 (i.e., vertical flow) is calculated as the mean discrepancy between the original model and metamodel, over depth and the study area. However, the discrepancy between the respective vertical flows has not yet been regarded over depth. Figure 36 shows the mean error in vertical flow over the study area (horizontally) vs depth, as FLF [L^3T^{-1}]. Above mean sea level (0m), the vertical flow downward calculated by the metamodel is much lower than that in the original model. Below 0m, the metamodel calculated vertical flow downward is higher than that of the original model, until the error approaches zero at a depth of -125m. At this depth, both the metamodel and original model vertical flows approach zero.

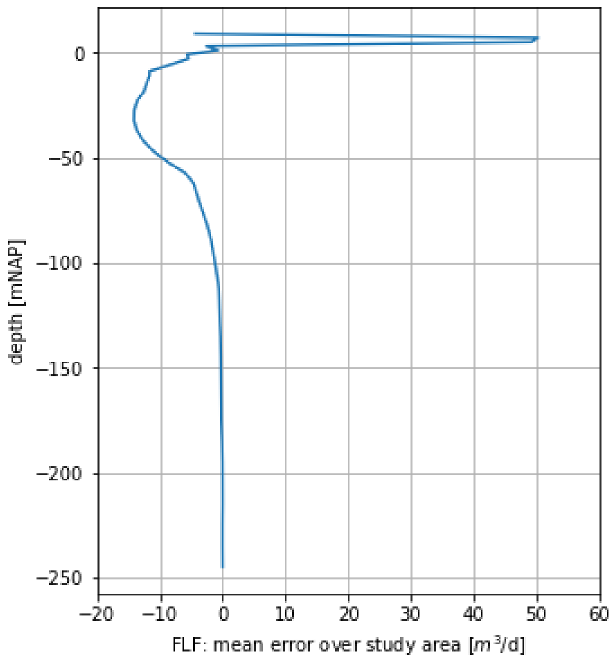


Figure 36: The discrepancy between the original model and the metamodel's vertical flow (FLF, see Figure 10 for convention) as a mean error over study area vs depth. (Steady-State)

A closer look at the discrepancy over depth may also be necessary for the hydraulic heads, as this has not been regarded yet. As presented in Section 2.2.2, the original model calculates higher steady state heads in the study area than the metamodel, calculated as a mean over depth, although the respective discrepancy in hydraulic heads over the study area vs depth, has not been considered yet.

Provided in Figure 37 is the discrepancy in hydraulic heads between the original and metamodel. The metamodel calculates lower hydraulic heads than the original model in the study area, up to a depth of -77.5m, below which the metamodel calculates higher hydraulic heads. The maximum discrepancy is seen above 0m, where the metamodel calculates hydraulic heads in the study area that are 1.3m lower than that of the original model.

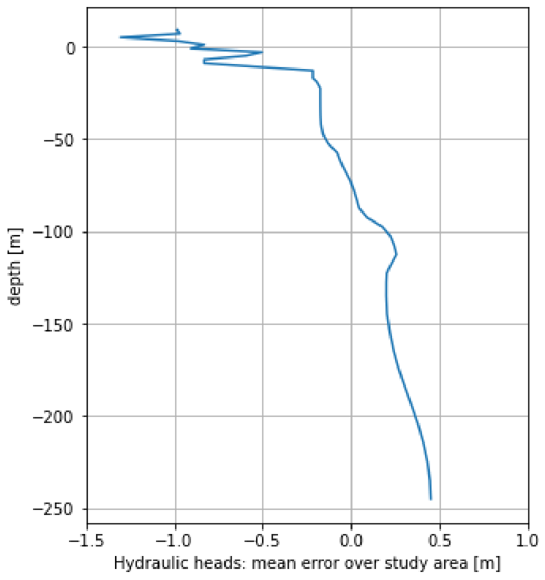


Figure 37: discrepancy between original and metamodel's calculated steady-state hydraulic heads over study area vs depth.

Both the vertical flow from FLF budgets and the hydraulic heads show a maximum discrepancy above 0m in the study area. The observed lower hydraulic heads and lower vertical flow at this location calculated by the metamodel hint at accordance with one another. The maximum discrepancy at this location may be due to the calibrated conductance of the infiltration ponds (situated above 0m) at a factor of 0.625 with the original model's conductance to match the water balances of the boundary conditions (Section 2.2.2).

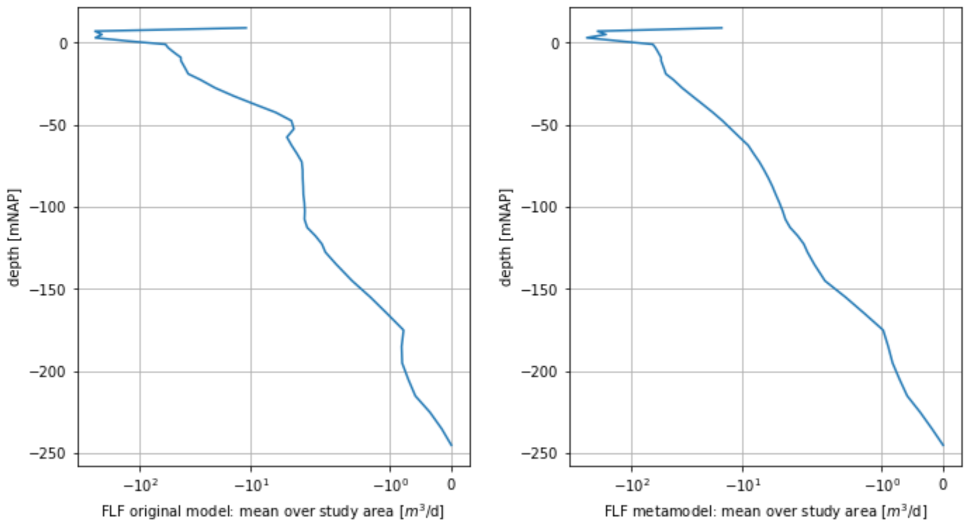


Figure 38: Comparison of steady state vertical flows in the study area (See Figure 2 for study area, see Figure 10 for vertical flow, FLF convention) for the original model and metamodel. The vertical flows area calculated as mean over the study area, expressed vs depth. The discrepancy between these two is shown in Figure 37.

When observing the vertical flow itself as calculated by the two models (Figure 38), it can be seen that a high vertical flow is observed in both, above 0m (above mean sea level, in the dunes). Below 0m, the vertical flows in the original model decrease rapidly up to a depth of -50m, below which the vertical flows slowly approach zero at -150m. The metamodel's vertical flows decrease over a larger depth, eventually approaching zero at -150m. The difference between the two model's calculated vertical flows over depth is shown in Figure 36.

The observed higher vertical flow downward by the metamodel at a depth of -30m shows no direct visual correlation with the discrepancy in hydraulic heads. This could be an effect of regridding. As discussed in Section 2.1.3., regridding of vertical hydraulic conductivities is done in this study by applying the harmonic mean, a method described as producing satisfactory representative groundwater flows (Bierkens, 1994). However, it has not been explicitly investigated if the correlation distance among original cells is smaller than the regridded cell size. In other words, it has not been investigated if no cells, whose parameters are uncorrelated, are regridded into one larger cell (with a single vertical hydraulic conductivity). This is a condition that should be met when applying the harmonic mean for vertical hydraulic conductivity (Bierkens, 1994).

4.1.2. Fresh-saline interface and HyFA

The fresh-saline interface and the Hydrochemical Facies Analysis (Stuyfzand, 1993a) can be used to further investigate the discrepancies between the original model and the metamodel. Two issues are addressed in this section: (I) The discrepancy in downward flow in the study area (between the original and metamodel), as indicated by the FLF values; (II) A saline groundwater intrusion found in the original model (Figure 19), and a closer look at the implementation of the fresh-saline interface in this study.

Before going into this discussion, however, it should be stressed again that the original model has not been calibrated on long term groundwater salinity, and although the metamodel (which has been calibrated to the original model in water balance, hydraulic head distribution and groundwater salinity) does show semblance with the HyFA, (see Sections 3.2.1 and 3.2.2), this does not necessarily mean that the original model's f/s interface is in fact, closest to the actual f/s interface.

Downward flow in the study area

The discrepancy in downward flow and hydraulic heads shows that the metamodel has a lower downward flow than the original model above 0m, in the dune area. This discrepancy is also appearing in the hydraulic head discrepancy in the study area over depth (Figure 37). In depths between 0 and -125m, the metamodel calculates higher vertical flows downward than the metamodel.

The rate at which the f/s interface increases in depth with respect to the original model (Figure 34), show that the f/s interface increases in depth in the metamodel, where the original model's fresh-saline interface slightly decreases in the study area. As can be seen in Figures 36 and 39, the metamodel's vertical downward flow approaches zero at a lower rate than that of the original model. This discrepancy accounts for a difference in infiltration rate, although the order of magnitude of several m^3/d may be too small to account for tens of meters of infiltration depth difference.

Shallow saline groundwater intrusion and the fresh-saline interface

In this study, the fresh-saline interface is calculated as the minimum depth at which the groundwater salinity becomes brackish (see Section 3.1.2). However, as seen from the CS3 and CS4 from the groundwater salinity cross sections perpendicular to the coastline, presented in Figure 39, in the original model, the saline groundwater intrusion occurs at a shallow depth, below which there is more fresh groundwater. Therefore, calculating the fresh-saline interface here as the minimum depth at which groundwater becomes brackish or saline, leads to an underestimation of the actual fresh-saline interface depth that lies below this intrusion zone.

This intrusion may “disturb” the presentation of fresh-saline interface development of the original model in Section 3.2.3, as there is another fresh-saline interface below the shallow intrusion that may still increase in depth. This could mean that in the original model, the “bottom” fresh saline interface could still increase in depth, instead of decreasing, as observed in Figure 34.

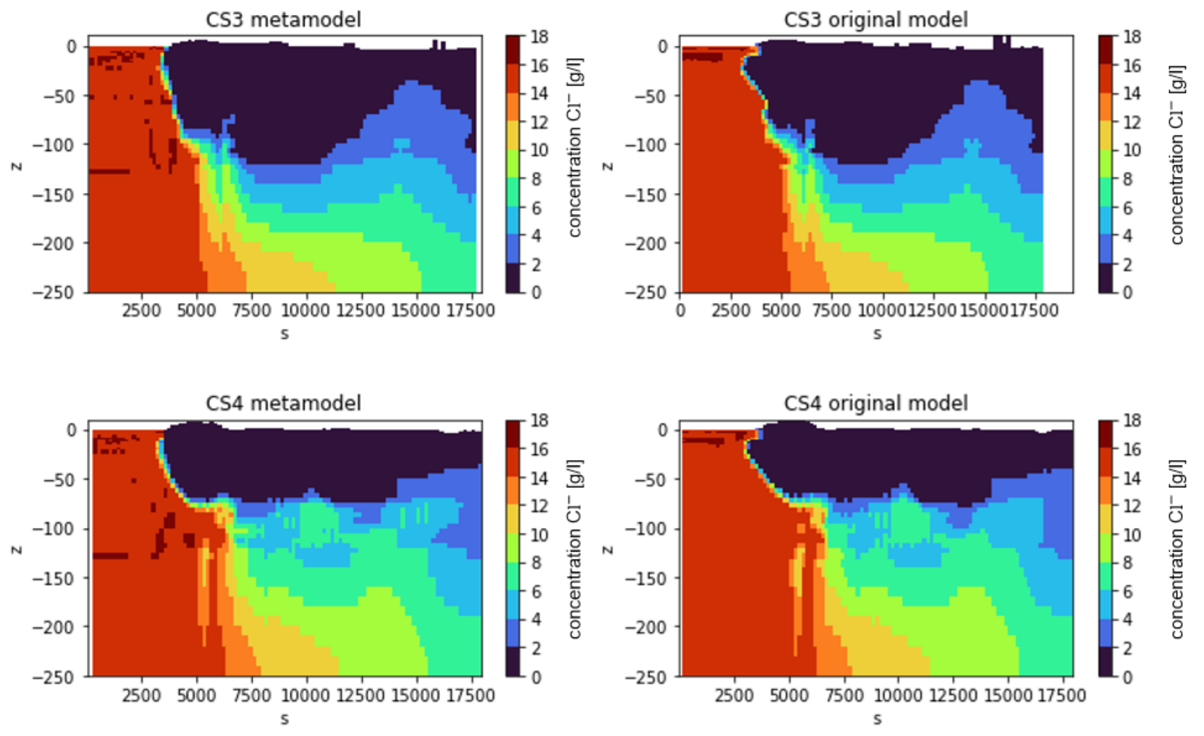


Figure 39: CS3 and CS4 perpendicular to coastline (See Figure 9 for all cross section) for the metamodel (Left) and the original model (right). The results shown are calculated by both models when starting with the same groundwater salinity and a simulation time of 40 years to investigate the discrepancy between the two.

An analysis of variable density flow may provide further insight in the discrepancy between the original model's shallow intrusion and its absence in the metamodel. The general form of Darcy's law for variable density conditions may be rewritten as: (Bear, 1979; Senger and Fogg, 1990; Langevin and Guo, 2006):

$$\mathbf{q} = -\mathbf{K}_\mathrm{f}\frac{\mu_\mathrm{f}}{\mu}\left(\nabla h_f + \frac{\rho-\rho_\mathrm{f}}{\rho_\mathrm{f}}\nabla z\right) \qquad \text{[Equation 2]}$$

Where $\mathbf{K} = \frac{\mathbf{k}\rho_f g}{\mu_\mathrm{f}}$ [LT^{-1}], k is the permeability tensor [L^2], μ is the dynamic viscosity [$ML^{-1}T^{-1}$], h_f is the freshwater head [L], p is pressure [$ML^{-1}T^{-2}$], g is the gravitational acceleration [LT^{-2}], and z is the upward coordinate direction aligned with gravity. This relation shows that horizontal flow components can be directly evaluated from the freshwater head, and vertical flow components can be calculated from the second term inside the parentheses (buoyancy term, Holzbecher 1998, Oude Essink, 1998).

The vertical buoyancy term (second term within the parentheses) can be of similar magnitude as the horizontal component (first term within the parentheses), and the effects of relative viscosity are neglected in SEAWAT (Langevin and Guo, 2006). Although uncalibrated on long term groundwater salinity, the original model can represent this shallow intrusion correctly since the horizontal flow component of variable density flow can directly be calculated from the hydraulic head gradient. Based on Equation 2 (Bear, 1979; Senger and Fogg, 1990; Langevin and Guo, 2006), it could therefore be stated that the saline groundwater intrusion represented by the original model can be correct, despite being uncalibrated on long term groundwater salinity data.

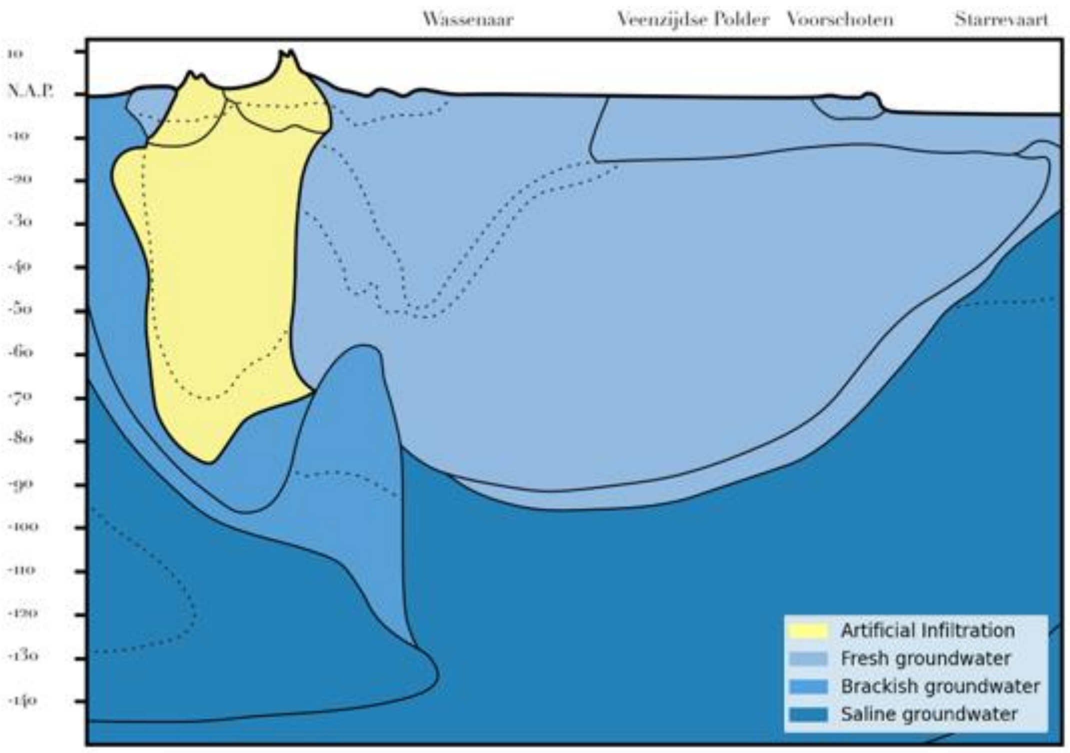


Figure 40: Copy of Figure 31 provided here for discussion's sake. Cross section perpendicular to the coastline, interpreted from the HyFA by Stuyfzand (1993b). For interpretation and original, see Appendix B.

Moreover, in the cross-section perpendicular to the coastline in the HyFA of Stuyfzand (1993b) (Figure 40, Appendix B), a shallow groundwater intrusion with the same horizontal extent (100s of m) and vertical extent (10s of m). Indicating that the original model may be representing this intrusion well.

Furthermore, by increasing the model's cell size, its capability to predict small scale groundwater flows may decrease. The shallow groundwater intrusion that is unrepresented in the metamodel may be a large-scale effect resulting from accumulation of small-scale groundwater flows that the coarse metamodel fails to capture because it is too coarsely gridded.

However, the vertical component of variable density flow is dependent on groundwater density (thus groundwater salinity) in the buoyancy term. Since the original model has not been calibrated on groundwater salinity data, vertical variable density flows may still not be well represented by the original model.

4.2. Efficient modeling practices

The future of groundwater modeling will partially depend on how well modeling serves the overall project (Langevin and Panday, 2012). Increasing model efficiency, not only by decreasing calculation time, but also by decreasing the effort in creating and validating the model, may benefit the project more than a highly accurate yet impractical model. The question is then: *how accurate and efficient should a model be?* This section approaches this question from various perspectives. First, a conceptual discussion on the boundaries of model accuracy is held in Section 4.2.1, After which conventional and unconventional model validation techniques are assessed in 4.2.2, with the knowledge gained from this study.

4.2.1. Model uncertainty, accuracy, and efficiency

A model is a simplified representation of a complex system (Clarke, 1973). A metamodel could therefore be seen as a superlatively simplified representation of a physical system, as it is constructed from an original model. The discussion in Section 4.1 has contributed to grasping the magnitude of these discrepancies and where they come from, but uncertainty remains as to where these discrepancies come from, and if these are small enough to argue that the proposed model adjustments are sufficient to retain sufficient model accuracy.

This problem can be approached from a theoretical point of view, by discussing hydrological models' predictive capacity for a physical process. In their study on model types for hydrology purposes, Clarke (1973) makes the distinction between *conceptual* models and *empirical* models, according to whether (conceptual) or not (empirical) they incorporate the physical process acting upon parameters to produce the output variables. Clarke (1973) argue that many models that are conceived as conceptual models are based on empirically derived parameters. Conversely, many empirical models have components that are conceptual. This distinction is made to address the feature that all models have in common: their predictions differ from the actual physical processes that occur (Clarke, 1973).

Moreover, distinction between empirical and conceptual modeling features may help in checking consistencies in assumptions on which the model relies, and to account for the discrepancies a model produces compared to observations (Clarke, 1973). Regridding the discretization of the original model to a coarser grid size, and calibrating to the cumulative effect on water balances or hydraulic heads is an empirical practice since it loses the conceptual value of small-scale variations in groundwater flow or subsurface characterizations. Subsequently, the original model is in itself empirical since it compromises the geohydrology to a 25x25m grid size.

The question may be posed at this point, if it is at all possible to represent all the small-scale variations in groundwater flow that the hydrological system contains, i.e., if it is possible to create a perfect model. In a discussion on "uniqueness of place" of hydrological systems (i.e., its variety of parameters influencing groundwater flow) and its limitations on groundwater modeling, Beven (2000) states that: Even a perfect model, which has parameter values specified for each location, will be subject to uncertainty of parameter values arising from the uniqueness of place, scale effects and measurements. Uncertainty appears to be inherently related to hydrological modeling. However, an approach to represent the hydrological system in a model may deal with uncertainty in an efficient manner and may still be valuable when applied to other hydrological systems.

4.2.2. Model Validation techniques

Quasi-paleo modeling

Proposed as a way out of the density feedback of solute concentration on groundwater flow by Delsman (2015), the use of paleo-modeling can be applied to mitigate the difficulties in choosing an initial concentration. By placing the starting point in time sufficiently far away, its influence on the current state may become negligible. Moreover, Delsman (2015) argues that neither the salinity distribution nor the hydraulic head distribution was, in their paleomodel, at any point in steady, state and that the assumption of steady state on present-day conditions (found in literature is) not warranted for coastal aquifers as the one considered in his research. This is underlined by the results of this study: the ever-increasing depth of fresh-saline interface (Figure 35) after a 200y simulation. This section is dedicated to adding to the discussion on what modeling approach to apply when considering coastal aquifers like the one studied in both Delsman's study (2015) and this study, with the purpose of contributing to sustainable groundwater management efforts globally.

As mentioned in Section: 1.1 in areas where extensive human intervention (like extraction of groundwater) takes place, the anthropogenic influence on fresh-saline distributions may become primary, on a much smaller timescale (Oude Essink, 2001). The metamodel results show that a similar fresh-saline interface with an artificially infiltrated volume of groundwater with an order of magnitude similar to that obtained in the HyFA could be obtained using a modeling approach which required a much smaller timescale than those found in paleo-reconstructions in literature (1-10 ka by Delsman (2015), 1-32ka by van Engelen (2020)). At first this may seem surprising, but it should be borne in mind that in this case, the water balance is dominated by the infiltration and extraction system of the reservoir, and not by natural recharge processes.

Moreover, as acknowledged by Delsman (2015) in their paleo-reconstruction: calibration and a rigorous sensitivity analysis could not be performed, due to the lack of data on such a long timescale and therefore calibration could only be done on the most recent periods. The methods used and shared in this study leave room for more rigorous calibration and sensitivity analyses. The remaining problem with choosing an initial groundwater salinity distribution could be mitigated when choosing a starting point in time that is sufficiently far away from the present, although this timescale may induce estimating climatological forcings like fluctuations in sea level and precipitation patterns.

This study has shown that validation on a shorter timescale is possible, for the purpose of investigating the artificial infiltration in the dune reservoir, occurring on unnaturally short timescales, adding to the validity of the hypothesis of Oude Essink (2001). By using the resulting salinity of a 100y simulation of the original model as starting salinity, the method used could be regarded as "quasi-paleo-modeling". This method acknowledges the fact that the coastal aquifer considered is at no point in steady state, but makes use of a shorter simulation time to increase calibration possibilities. As for steady state boundary conditions, this discussion remains restricted. Further research may be necessary to investigate the possibilities of quasi-paleo-modeling.

Hydrochemical Facies Analyses for model optimization

As argued by Langevin and Panday (2012), we can improve our groundwater models with more advanced modeling programs, faster computers, and better calibration strategies, but without better quality data and more of it, improvements in our models and their predictive capabilities will be modest. A feasible investment for site specific models for groundwater management support may therefore be the acquisition of quality data, like the HyFA. A balance between efficiency and accuracy may be easier achieved when introducing data to the model like the HyFA.

The Hydrochemical Facies Analysis shows similarity with the modeled groundwater distribution using origin species. The model results show two unconnected infiltrated groundwater regions, whereas the HyFA of Stuyfzand (1993a, 1993b) shows these artificially infiltrated groundwater regions are connected. As mentioned in Section 3.1.1, the horizontal sampling interval of 1km for the longitudinal cross section induces interpretation. The interpolation necessary to create a description of the subsurface as presented by (Stuyfzand, 1993) from data acquired with a sampling interval of 1km, could mean that this feature is represented well in the modeled groundwater distribution of this study, and unrepresented in the HyFA distribution.

Nevertheless, the value of the HyFA as a tool to study the "path" of groundwater has proven to be fruitful for model validation by the results of this study. By using species, the metamodel was able to reproduce a volume of artificially infiltrated fresh groundwater with the same order of magnitude as the HyFA. This confirms the hypothesis that HyFA is valuable for model validation, and over a relatively short time span. It also hints at the applicability of the original model: although uncalibrated on long term salinity, it may show similarity to the HyFA, as indicated by the metamodel.

Coming back to the statement by Langevin and Panday (2012) on the future of groundwater modeling: "[In the future] we will use models more effectively to not only seek solutions but also analyze data significance and guide further collection efforts toward minimizing uncertainty in predictions", the significance of groundwater salinity data and HyFA have proven to be insightful for investigating a coastal fresh groundwater reservoir, from a modeling perspective. When the efforts of Stuyfzand (1993a, 1993b) are incorporated in an early phase of a project, and groundwater samples are acquired simultaneously with collection of hydraulic head data and groundwater salinity data from boreholes, the use of HyFA combined with a modeling practice described in this study may be feasible to incorporate in groundwater modeling practice. For the aim of the original model (investigating the feasibility of brackish groundwater as a source of drinking water), early incorporation of the HyFA data to the model may have inferred the use of an even larger horizontal cell size.

Fresh-saline interface vs HyFA for model optimization

The fresh-saline interface can be used as a tool to study discrepancies between model outputs and be used as a validation tool as it incorporates variable density groundwater flow. It is relatively easy to incorporate from the modeled groundwater salinity output in SEAWAT, although consideration should be taken whether it is calculated as the maximum depth at which groundwater becomes brackish, or the minimum depth, as indicated by the discussion in Section 4.1.2.

For model validation, it requires data acquisition of groundwater salinity, from which a conceptual fresh saline interface can be developed. If data is collected over time, it can be a useful tool to study the depth at which the freshwater lens changes in size over time, whether it is a groundwater reservoir that is recharged, or drained. The model output can easily then be adjusted to study discrepancies with the collected data, at times.

For model validation, the fresh-saline interface may easier implement than the HyFA in a project. It requires less data interpretation and sampling than the HyFA and is therefore less labor intensive. From a modeling perspective, these methods are both relatively easily implemented. However, the HyFA does show insight in the path that groundwater travels and the geohydrological history, even when this analysis is only conducted once.

4.3. Conclusions

The Hydrochemical Facies Analysis (HyFA) as conducted by Stuyfzand (1993a, 1993b) has multiple benefits for modeling coastal groundwater resources, based on the findings of this study. Firstly, it allows for validation of modeled groundwater flow paths, which is beneficial to validation on snapshot data. Secondly, the HyFA is a valuable tool for variable density groundwater model validation not only for 2D models on a paleo-timescale, but also for 3D models and on a shorter timescale, particularly in the context of areas with extensive human intervention. Finally, it is relatively easy to implement in the SEAWAT modeling structure using “species”.

Although the total effort required for model validation to the fresh-saline interface may be less than validation to the HyFA, it does not give insight in the path and history of groundwater like the HyFA does. The HyFA and the fresh-saline interface are both easily implemented in SEAWAT, although data acquisition and analysis may be more labor intensive for the HyFA.

The methods implemented in the metamodel to increase model efficiency led to a decrease in calculation time. Using the input and output data from the original model as a starting point, and initializing an upscaled metamodel involves rescaling parameters and setting up boundary conditions to control the model’s variable density groundwater flow and to ensure sufficient semblance to the original model.

The original model has not been validated on long-term groundwater salinity. However, the results of the metamodel (which has been calibrated to the original model) show significant semblance to the HyFA. Therefore, the original model has not been invalidated based on this study. However, the HyFA in itself is also an interpretation which has uncertainties, although incorporating it in the metamodel still gives insight in the original model’s calibration: in the northern part of the study area, the metamodel calculates an artificial infiltration rate that is too large. Potentially, this could point at a hydraulic conductance term that is too large for the infiltration ponds. Further research would be necessary to explicitly investigate the original model’s hydraulic conductance in this area, as this could lead to an overestimation of the fresh groundwater reservoir.

The conclusions of this study contribute to the project owners’ previous efforts (Dunea, Deltares, Arcadis). Moreover, the findings of this research are applicable to drinking water resource management in coastal areas globally.

4.4. Recommendations

Based on this study's findings, the adaptation of the Hydrochemical Facies Analysis (HyFA) in variable density groundwater model validation is recommended to be continued in further research. Another feasible option to be used in future studies for gaining insight in the modeled variable density groundwater flows, is incorporating the fresh-saline interface, although this does not reveal as much insight in the "path" of groundwater as the HyFA.

This study has primarily shown the potential of the HyFA from a modeler's perspective. Establishing an efficient approach to a geochemical analysis like the HyFA requires expertise from other fields. Therefore, further research is necessary to study the feasibility of HyFA from all perspectives. Moreover, it should be acknowledged that creating a hydrochemical cross section for validation requires interpretation, and consequently uncertainty. This is a crucial insight when validating a groundwater model to any data, or interpretation that can help identify the origins of uncertainties in the model outputs.

Further research is also necessary to investigate the excessive artificial infiltration in the northern part of the study area revealed by the metamodel and the HyFA. Primarily, this could be investigated by incorporating the species dimension in the original model and validating with the HyFA. Secondly, this could involve conducting additional field studies and gathering more data to decrease uncertainty on the model outputs in this specific region.

For efficient modeling, the tradeoff between accuracy and efficiency should be recognized from the start. Methods in this study have been successful for increasing efficiency, while retaining accuracy but this is highly dependent on the objective that the model serves. Careful consideration must be taken when choosing a grid size to represent a complex hydrological system. Through transparency, this study aims at contributing to efficient modeling practices. This study could be consulted when developing groundwater models for coastal drinking water resources in other areas.

A model like the metamodel could be fit to further investigate timescales on which the fresh-saline interfaces can develop with artificial infiltration. As the climate is changing, drinking water resource in coastal areas may rely more on groundwater recharge through artificial infiltration. A model like the metamodel may be valuable for investigating the recharge rate of a drinking water reservoir, after years of drought, overexploitation or when assessing the feasibility of a site as a drinking water resource.

5. Bibliography

Arcadis, Deltares, KWR. (2019). Vergroten overbruggingsvermogen: naar een robuuste strategische zoetwatervoorraad. Amersfoort: Allied Waters. (Dutch). English translation: Increasing bridging capacity: towards a robust and strategic fresh water supply.

Akrofi, T. F.; Farmer, T.; Nganyi, J. (2016). Human Settlements on the Coast. *The ever mode popular coasts.* United Nations Atlas of the Oceans. Retrieved from: http://www.oceansatlas.org/subtopic/en/c/114/

Barton, R. R. (1998), Simulation metamodels, in Proceedings of the 30th Conference on Winter Simulation, edited by D. J. Medeiros and E. F. Watson, pp. 167– 176, IEEE Press, Washington, D. C.

Basco Carrera, L., van Beek, E. van Deursen, W., Choudhury, G.A., Haasnoot, M. (2018). Fast Integrated Systems Modeling: The method and its application in Bangladesh and the Philippines. 9th International Congress on Environmental Modelling and Software, BYU ScholarsArchive

Bear, J. 1979. Hydraulics of Groundwater. New York: McGrawHill.

Beven, K.J. (2000). Uniqueness of place and process representations in hydrological modelling. Hydrology and Earth System Sciences, 4(2), 203-213.

Bierkens, M.F.P. (1994). Block conductivities; upscaling in geohydrology. (In Dutch: Blokdoorlatendheden; opschaling in de geohydrologie. H_20 (27) nr. 23. Retrieved from: https://edepot.wur.nl/371116

Bootsma, H., Boonekamp, T., Huizer, S. (2021). Coastar case A2: Brakwaterwinning kust voorstudie en aanzet tot pilot voor toepassing brakwater als derde drinkwaterbron TKI-WT Coastar. Klaasjan Raat KWR Water Research Institute.

Chen, J. H., (1973). Numerical Boundary Conditions and Computational Modes. Geophysical Fluid Dynamics Laboratory/NOAA, Princeton University, Princeton, New Jersey 08540

Clarke, R.T., 1973. A review of some mathematical models used in hydrology, with observations on their calibration and use. J. Hydrol., 19: 1-20.

Crout, N.; Kokkonen, T.; Jakeman, A.J.; Norton, J.P.; Newham, L.T.H.; Anderson, R.; Assaf, H.; Croke, B.F.W.; Gaber, N.; Gibbons, J.; Holzworth, D.; Mysiak, J.; Reichl, J.; Seppelt, R.; Wagener, T.; Whitfield, P. (2008). Good Modeling Practice. Environmental Modelling, Software and Decision Support.

Delsman, J. R. (2015). Saline groundwater - surface water interaction in coastal lowlands. IOS Press BV. https://doi.org/10.3233/978-1-61499-518-0-i

Deltares. iMOD Python: imod.prepare.regridder. API Reference – Prepare model input. https://deltares.gitlab.io/imod/imod-python/api/generated/prepare/imod.prepare.Regridder.html#imod.prepare.Regridder

Fraser, C. E., McIntyre, N., Jackson, B. M., Wheater, H. S. (2013). Upscaling hydrological processes and land management change impacts using a metamodeling procedure

Gary Wang, G., Shan, S. (2006). Review of Metamodeling Techniques in Support of Engineering Design Optimization. Journal of Mechanical Design Volume 129, Issue 4.

Harbaugh, A.W., 2005, MODFLOW-2005, the U.S. Geological Survey modular ground-water model—the Ground-Water Flow Process: U.S. Geological Survey Techniques and Methods, book 6, chap. A16, variously paged, accessed June 27, 2017, at https://pubs.usgs.gov/tm/2005/tm6A16/.

Hill, M.C. (1990) preconditioned Conjugate Gradient 2. A computer program for solving ground-water flow equations. U.S.G.S. Water-Resources Investigations Report 90-4048.

Holzbecher, E. 1998. Modeling Density-Driven Flow in Porous Media: Principles, Numerics, Software. Berlin Heidelberg, Germany: Springer-Verlag.

Hoogland, T., van den Akker, J.J.H., Brus, D.J., (2012) Modeling the subsidence of peat soils in the Dutch coastal area. doi https://doi.org/10.1016/j.geoderma.2011.02.013

Hunt, R.J., Feinstein, D.T., Pint, C.D., Anderson, M.P. (2005). The importance of diverse data types to calibrate a watershed model of the Trout Lake Basin, Northern Wisconsin, USA. U.S.G.S., Journal of Hydrology 321 (2006) 286-296

Hunt, R.J. 2002. Evaluating the importance of future data collection sites using parameter estimation and analytic element groundwater flow models. Developments in Water Science, volume 47 pages 775-761. Doi: https://doi.org/10.1016/S0167-5648(02)80136-8

Jakeman, A.J., Chen, C.H., Rizzoli, A.E. and Voinoc, A.A. (2008) Modelling and Software as Instruments for Advancing Sustainability. Environmental Modelling, Software and Decision Support. https://reader.elsevier.com/reader/sd/pii/S1574101X08006017?token=CA22B3771323F33509F5782EC8FABD58AC6E77A2701E3C70C8B32AFC43DBD638A9F47446619E502B76094EF8698F7708&originRegion=eu-west-1&originCreation=20230413134332

Knoben, W. J. M., Clark, M. P., Bales, J., Bennett, A., Gharari, S., Marsh, C. B., et al. (2022). Community Workflows to Advance Reproducibility in Hydrologic Modeling: Separating model-agnostic and model-specific configuration steps in applications of large-domain hydrologic models. Water Resources Research, 58, e2021WR031753. https://doi. org/10.1029/2021WR031753

Langevin, C. D., and W. Guo (2006), MODFLOW/MT3DMS-based simulation of variable-density ground water flow and transport., Ground Water, 44(3), 339–51, doi:10.1111/j.1745-6584.2005.00156.x.

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, U.S.G.S., Chapter 55 of Section A, Groundwater Book 6, Modeling Techniques.

Langevin C. D., Panday, S. (2012) Future of Groundwater Modeling , USGS., Vol. 50, No. 3–GROUND WATER–May-June 2012. doi: 10.1111/j.1745-6584.2012.00937.x

Langevin, C.D. and Guo, W. (2002). User's Guide to SEAWAT: A Computer Program for Simulation of Three-Dimensional Variable-Density Ground-Water Flow (U.S. Geological Survey Techniques of Water-Resources Investigations ; 6 A7) Includes bibliographical references. ISBN 0-607-99257-3

Leake, S.A., and Prudic, D.E. (1991). Documentation of a computer program to simulate aquifer-system compaction using the modular finite-difference ground-water flow model: U.S. Geological Survey Techniques of Water-Resources Investigations, book 6, chap. A2, 68 p., accessed June 27, 2017, at https://pubs.er.usgs.gov/publication/twri06A2.

Lusczynski, N.J. (1961). Head and flow of groundwater of variable density.*J. Geophys. Res.*, 66 (1961), pp. 4247-4256

McDonald, M.G., and Harbaugh, A.W., 1988, A modular three-dimensional finite-difference ground-water flow model: U.S. Geological Survey Techniques of Water-Resources Investigations, book 6, chap. A1, 586 p., accessed June 27, 2017, at https://pubs.er.usgs.gov/publication/twri06A1.

McDonald, M.G., Harbaugh, A.W., Orr, B.R., and Ackerman, D.J., 1992, A method of converting no-flow cells to variable-head cells for the U.S. Geological Survey modular finite-difference ground-water flow model: U.S. Geological Survey Open-File Report 91–536, 99 p., accessed June 27, 2017, at https://pubs.er.usgs.gov/publication/ofr91536.

Mehl, S. W., and M. C. Hill (2001), MODFLOW-2000, the U.S. Geological Survey modular ground-water model. User guide to the Link-AMG (LMG) package for solving matrix equations using an algebraic multigrid solver, U.S. Geol. Surv. Open File Rep., 01-177, 33 pp.

Mehl, S. W., and M. C. Hil (2003), Locally refined block-centered finitedifference groundwater models, in Evaluation of Parameter Sensitivity and the Consequences for Inverse Modelling and Predictions, IAHS Publ., 277, 227 – 232

Oude Essink, Gualbert H.P (2001). "Improving fresh groundwater supply—problems and solutions." In: Ocean & Coastal Management 44.5, pp. 429–449. doi: 10 . 1016 / S0964-5691(01)00057-6.

Oude Essink, G.H.P. 1998. MOC3D adapted to simulate 3D density-dependent groundwater flow. In Proceedings of MODFLOW '98 Conference at the International Ground Water Modeling Center, vol. 1, 291–300. Golden, Colorado: Colorado School of Mines.

Post, V., Kooi, H., Simmons, C. (2007). Using hydraulic head measurements in variable-density ground water flow analyses. National Groundwater Association. Groundwater / Volume 45, Issue 6, p. 664-671. Doi: 10.1111/j.1745-6584.2007.00339.x

Post, V. E. A., and H. Kooi (2003), Rates of salinization by free convection in high-permeability sediments: insights from numerical modeling and application to the Dutch coastal area, Hydrogeol. J., 11(5), 549–559, doi:10.1007/ s10040-003-0271-7.

Royal HaskoningDHV. (2018). Dunea - Overbruggingsvermogen duin. Amsterdam: HaskoningDHV Nederland B.V. (In Dutch)

Seibert, J., McDonnell, J.J. 2002. On the dialog between experimentalist and modeler in catchment hydrology: Use of soft data for multicriteria model calibration. Water Resources Research, volume 38 Issue 11 pages 23-1 / 23-14. doi: https://doi.org/10.1029/2001WR000978

Senger, R.K., and G.E. Fogg. 1990. Stream functions and equivalent freshwater heads for modeling regional flow of variable-density groundwater. 1. Review of theory and verification. Water Resources Research 26, no. 9: 2089–2096.

Stagge, J. H., Rosenberg, D. E., Abdallah, A. M., Akbar, H., Attallah, N. A., & James, R. (2019). Assessing data availability and research reproducibility in hydrology and water resources. Scientific Data, 6(1), 190030. https://doi.org/10.1038/sdata.2019.30

Stuyfzand, P. J. (1993a). Hydrochemistry and hydrology of the coastal dune area of the Western Netherlands. KIWA N.V.

Stuyfzand, P.J., Luers, F., de Jonge, H.G. (1993b) SWE 93.001 Hydrochemistry and hydrology of the dunes and adjacent polders between Katwijk and Kijkduin. KIWA N.V. Personally provided by P.J. Stuyfzand upon request via e-mail.

TNO, Netherlands Ministry of Internal Affairs. GeoTOP (GTM). BRO – Basisregistratie Ondergrond. Retrieved from: https://basisregistratieondergrond.nl/inhoud-bro/registratieobjecten/modellen/geotop-gtm/

TNO, Netherlands Ministry of Internal Affairs. REGIS, hydrological model (HGM). Retrieved from: https://basisregistratieondergrond.nl/inhoud-bro/registratieobjecten/modellen/regis-ii-hydrogeologisch-model-hgm/

TNO, Netherlands Ministry of Internal Affairs. National Key Registry of the Subsurface. BRO: Transparent. Clear. Accessible. Retrieved from: https://basisregistratieondergrond.nl/english/about-key-registry/

U.S.G.S. MODFLOW time dialog boxes. Model Muse. Retrieved from: https://water.usgs.gov/nrp/gwsoftware/ModelMuse/Help/modflow_time_dialog_box.html

United Nations Statistics Division (2021). Demographic Yearbook 2021. Retrieved from https://en.wikipedia.org/wiki/List_of_countries_and_dependencies_by_population_density#Most_populous_countries_by_density van de Ven, G.P. (1993) Man-Made Lowlands: History of Water Management and Land Reclamation in the Netherlands. Uitgeverij Matrijs, Utrecht.

van Engelen, J. (2020). Fresh Groundwater Reserves in Major Deltas: Evolution and state of Deltaic Groundwater Resources. Utrecht: Faculty of Geosciences, University Utrecht.

Vermeulen, P. T. M., te Stroet, C. B. M.; and Heemink, A. W. (2006), Model inversion of transient nonlinear groundwater flow models using model reduction, Water Resour. Res., 42, W09417, doi:10.1029/2005WR004536.

Werner, A.D.; Bakker, M.; Post, V.E.A.; Vandenbohede, A.; Lu, C.;Ataie-Ashtiani, B.; Simmons, C.T.; Barry, D.A. (2012). Seawater intrusion processes, investigation and management: Recent advances and future challenges. *Advances in Water Resources* 51 3-26

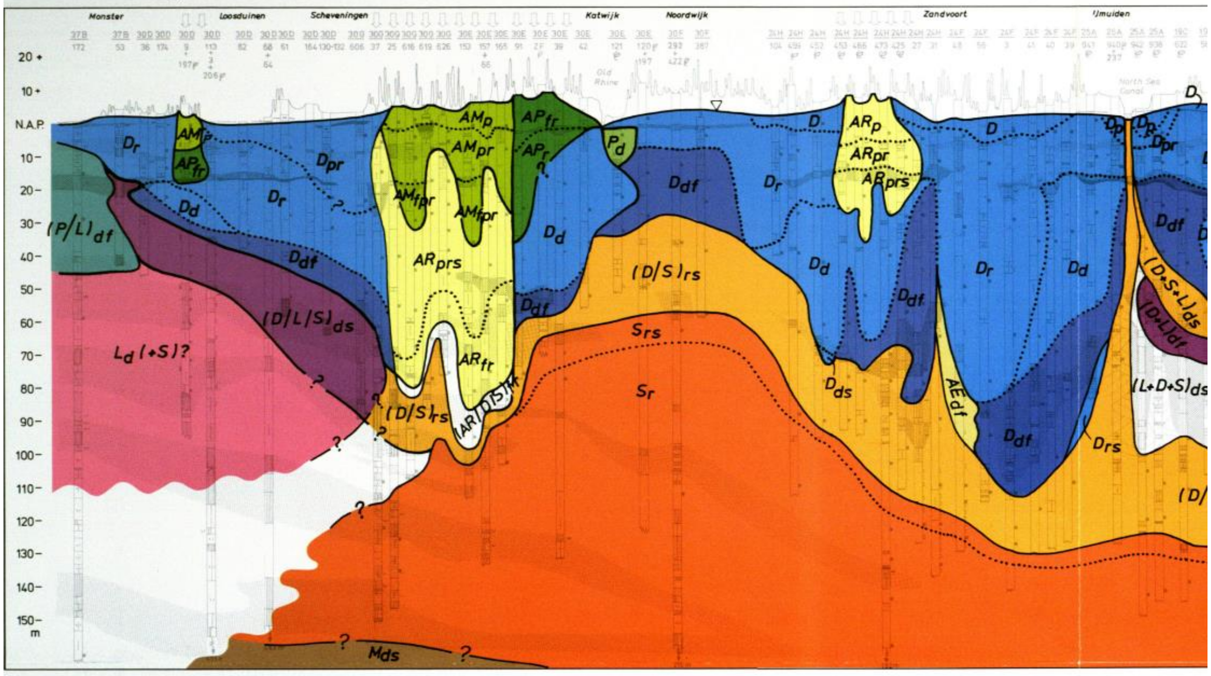
Xarray Documentation. (2023). Xarray developers. NumFOCUS. Accessible via: *https://docs.xarray.dev/en/stable/*

Yu, X., & Michael, H. A. (2022). Impacts of the scale of representation of heterogeneity on simulated salinity and saltwater circulation in coastal aquifers. Water Resources Research, 58, e2020WR029523

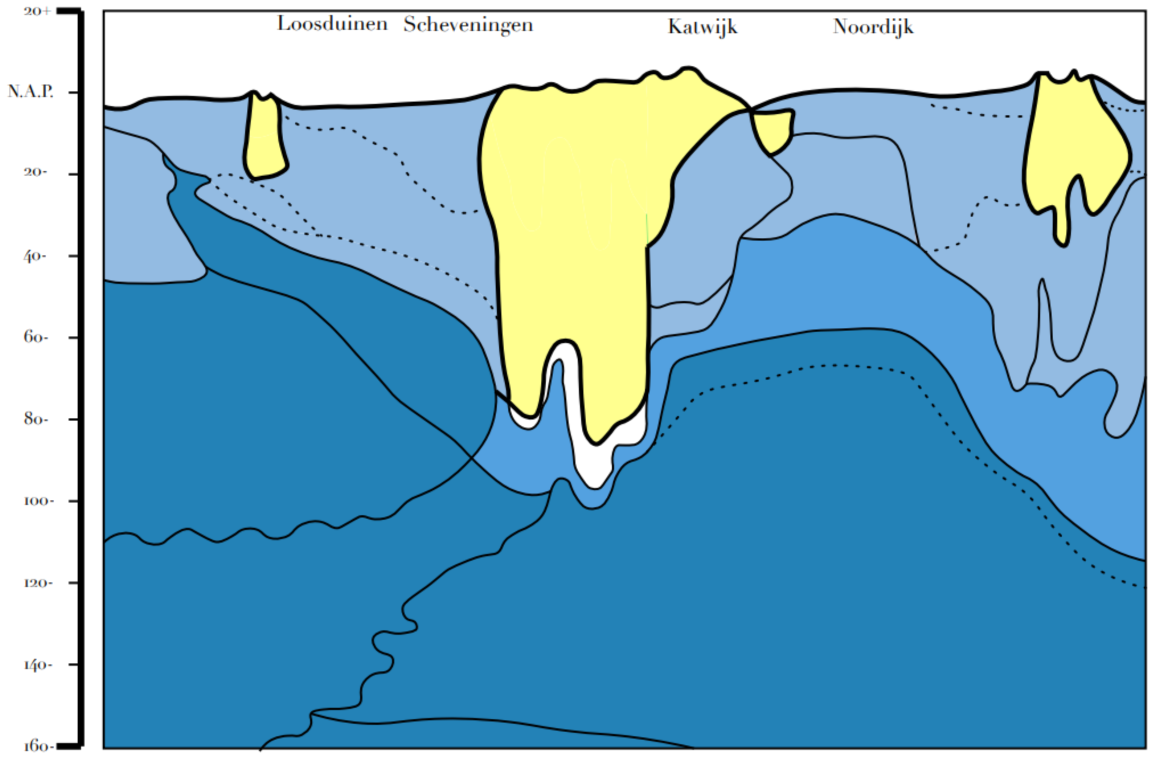
Zheng, Chunmiao, and Wang, P.P., 1999, MT3DMS—A modular three-dimensional multi-species transport model for simulation of advection, dispersion and chemical reactions of contaminants in groundwater systems; Documentation and user's guide: Contract report

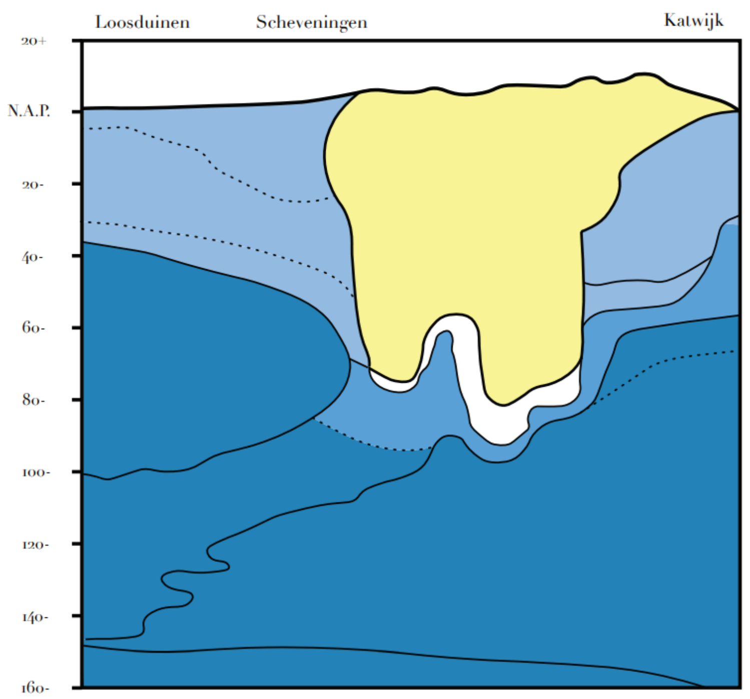
6. Appendix A: Interpretation of hydrochemical longitudinal cross section

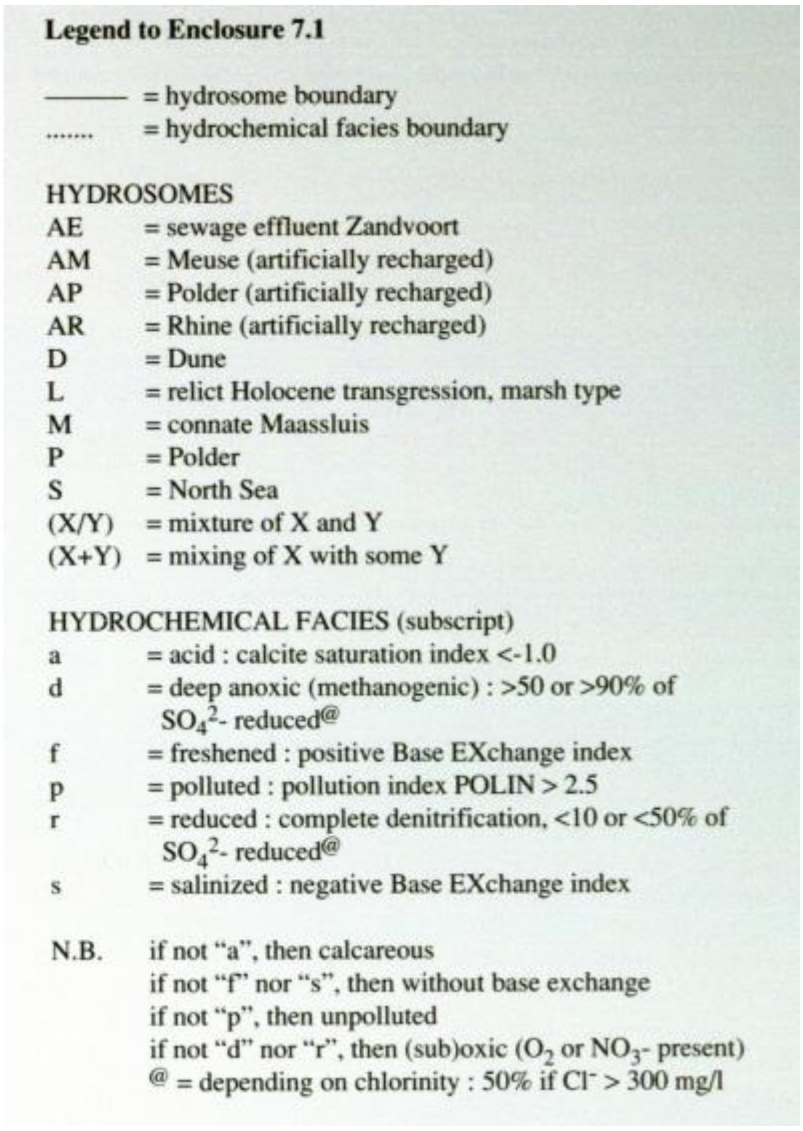
HYDROCHEMICAL, LONGITUDINAL SECTION OVER THE MONSTER-CAMPERDUIN AREA : HYDROSOMES, THEIR FACIES AND CHEMICAL WATERTYPES



ENCLOSURE 7.1 Longitudinal section A (for location see Fig.3.1), over the younger dune area of the Western Netherlands, with the spatial distribution of hydrosomes and their hydrochemical facies.







Legend to Enclosure 7.1

———	= hydrosome boundary
…….	= hydrochemical facies boundary

HYDROSOMES

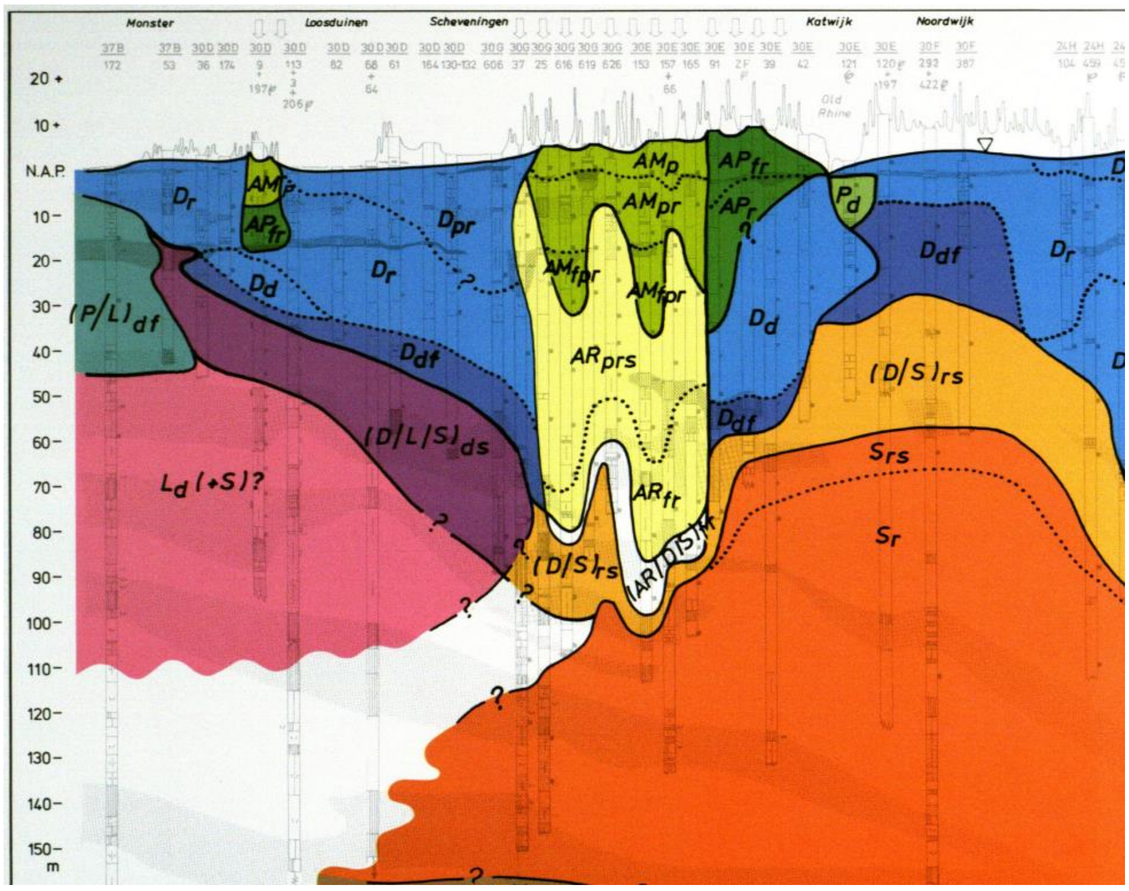
AE	= sewage effluent Zandvoort
AM	= Meuse (artificially recharged)
AP	= Polder (artificially recharged)
AR	= Rhine (artificially recharged)
D	= Dune
L	= relict Holocene transgression, marsh type
M	= connate Maassluis
P	= Polder
S	= North Sea
(X/Y)	= mixture of X and Y
(X+Y)	= mixing of X with some Y

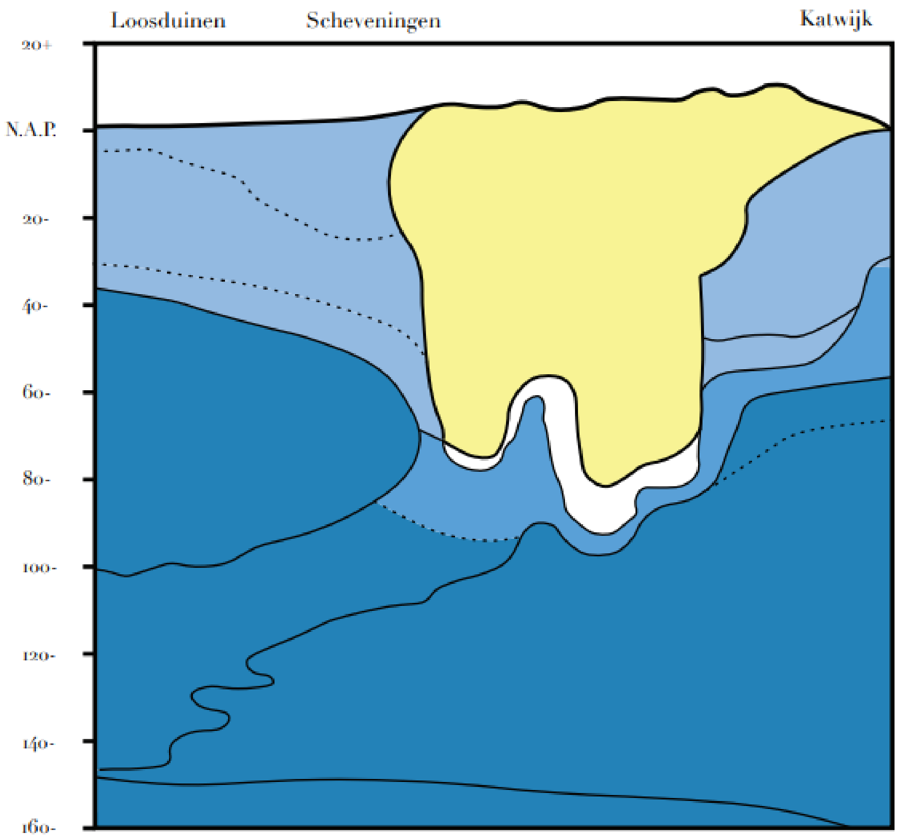
HYDROCHEMICAL FACIES (subscript)

a	= acid : calcite saturation index <-1.0
d	= deep anoxic (methanogenic) : >50 or >90% of SO_4^{2-} reduced@
f	= freshened : positive Base EXchange index
p	= polluted : pollution index POLIN > 2.5
r	= reduced : complete denitrification, <10 or <50% of SO_4^{2-} reduced@
s	= salinized : negative Base EXchange index

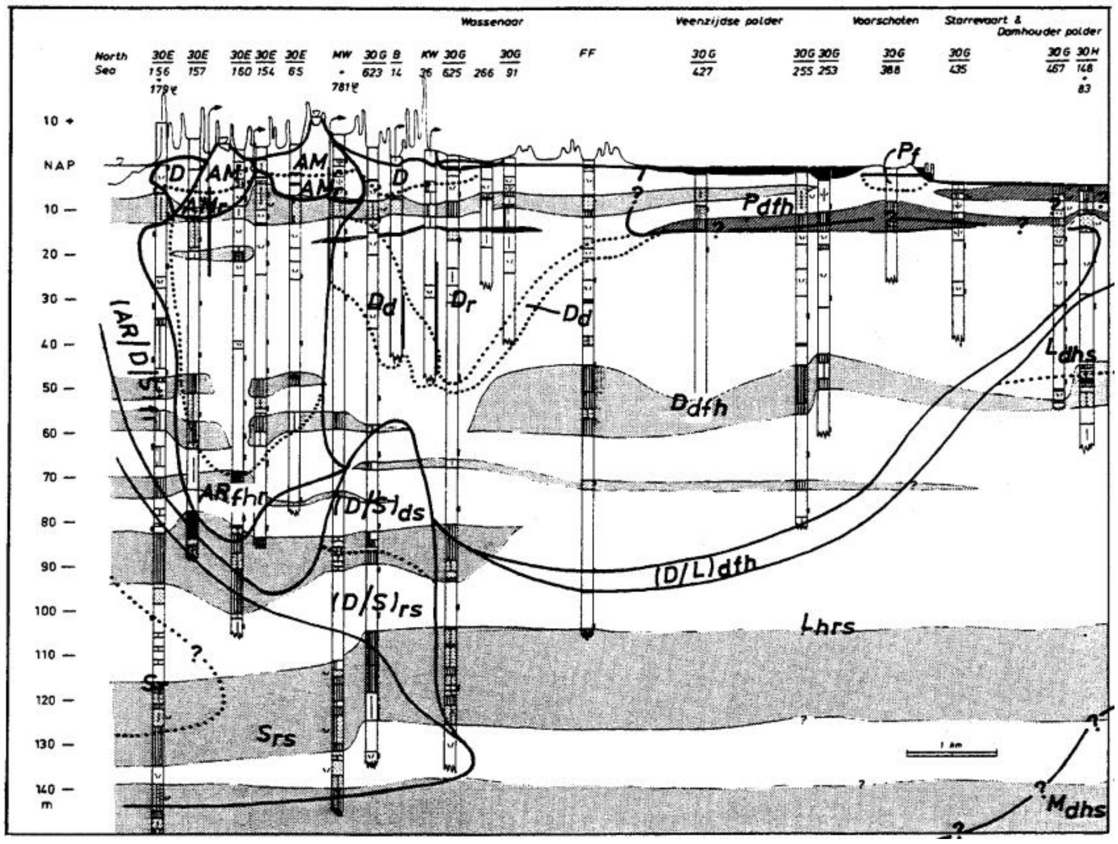
N.B. if not "a", then calcareous
if not "f" nor "s", then without base exchange
if not "p", then unpolluted
if not "d" nor "r", then (sub)oxic (O_2 or NO_3^- present)
@ = depending on chlorinity : 50% if $Cl^- > 300$ mg/l

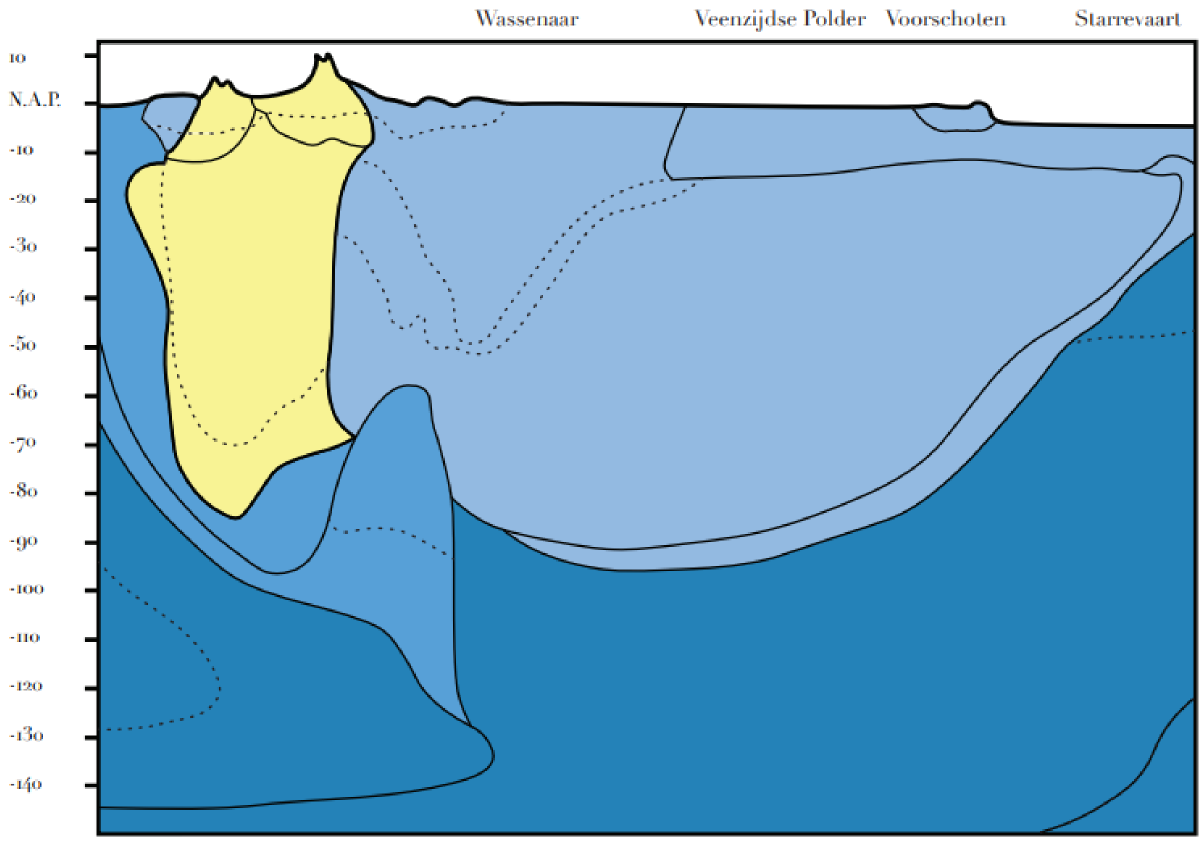
Legend to cross section parallel to coastline, provided on previous page.

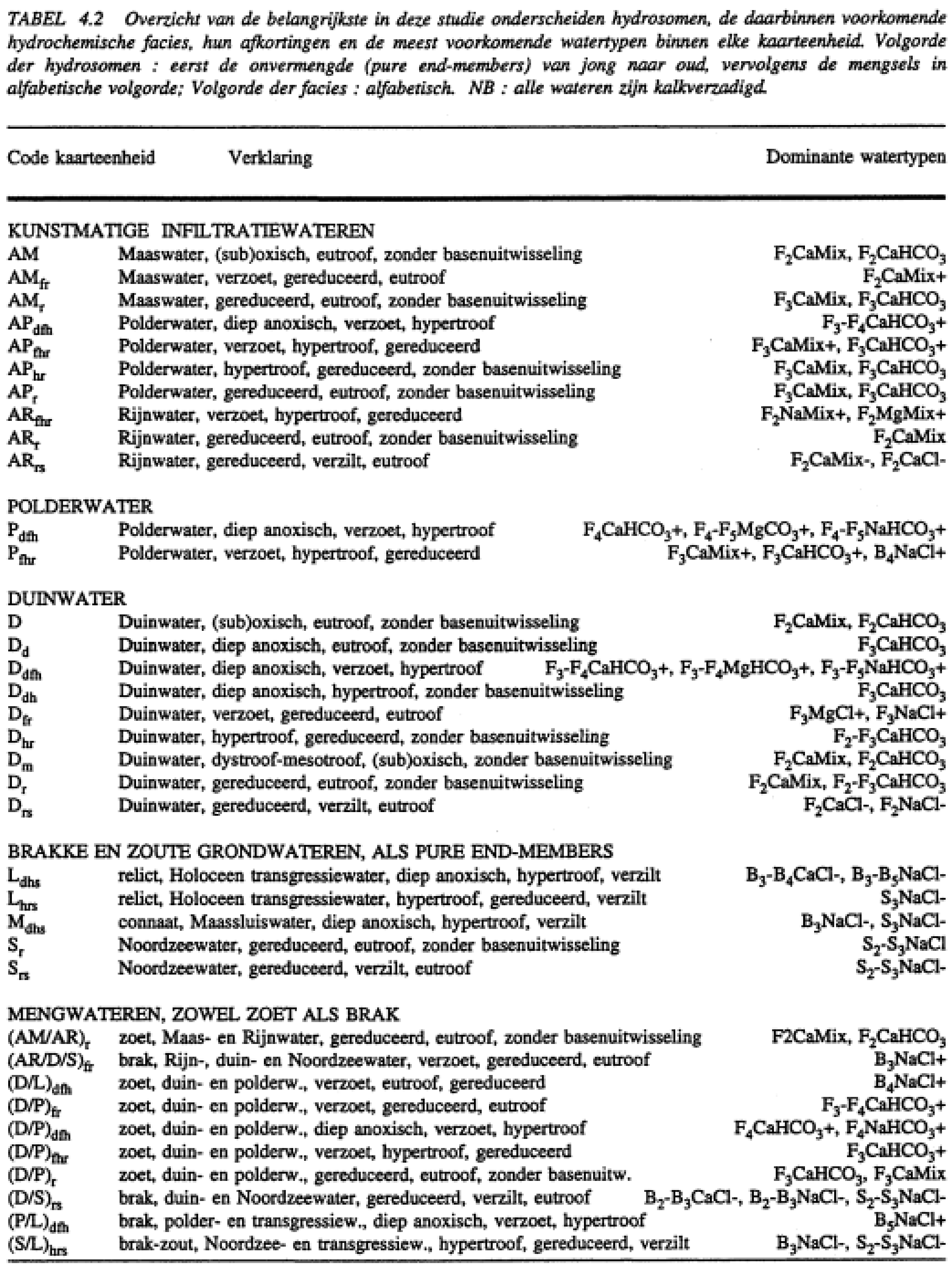




7. Appendix B: Interpretation of hydrochemical perpendicular cross section to coastline


Wassenaar
Veenzijdse polder
Voorschoten
Starrevaart & Damhouder polder
North Sea
NAP
10 m
1 km
(D/AM)
AM
D
Pdfh
Pf
Dd
Dr
Ddfh
Ldhs
(AR/D/S)
ARfh
(D/S)ds
(D/L)dfh
(D/S)rs
Lhrs
Srs
Mdhs


Wassenaar
Veenzijdse Polder
Voorschoten
Starrevaart
10
N.A.P.
-10
-20
-30
-40
-50
-60
-70
-80
-90
-100
-110
-120
-130
-140



TABEL 4.2 Overzicht van de belangrijkste in deze studie onderscheiden hydrosomen, de daarbinnen voorkomende hydrochemische facies, hun afkortingen en de meest voorkomende watertypen binnen elke kaarteenheid. Volgorde der hydrosomen : eerst de onvermengde (pure end-members) van jong naar oud, vervolgens de mengsels in alfabetische volgorde; Volgorde der facies : alfabetisch. NB : alle wateren zijn kalkverzadigd.

Code kaarteenheid	Verklaring	Dominante watertypen
KUNSTMATIGE INFILTRATIEWATEREN		
AM	Maaswater, (sub)oxisch, eutroof, zonder basenuitwisseling	F_2CaMix, F_2CaHCO_3
AM_{fr}	Maaswater, verzoet, gereduceerd, eutroof	$F_2CaMix+$
AM_r	Maaswater, gereduceerd, eutroof, zonder basenuitwisseling	F_3CaMix, F_3CaHCO_3
AP_{dfh}	Polderwater, diep anoxisch, verzoet, hypertroof	F_3-F_4CaHCO_3+
AP_{fhr}	Polderwater, verzoet, hypertroof, gereduceerd	$F_3CaMix+$, F_3CaHCO_3+
AP_{hr}	Polderwater, hypertroof, gereduceerd, zonder basenuitwisseling	F_3CaMix, F_3CaHCO_3
AP_r	Polderwater, gereduceerd, eutroof, zonder basenuitwisseling	F_3CaMix, F_3CaHCO_3
AR_{fhr}	Rijnwater, verzoet, hypertroof, gereduceerd	$F_2NaMix+$, $F_2MgMix+$
AR_r	Rijnwater, gereduceerd, eutroof, zonder basenuitwisseling	F_2CaMix
AR_{rs}	Rijnwater, gereduceerd, verzilt, eutroof	$F_2CaMix-$, F_2CaCl-
POLDERWATER		
P_{dfh}	Polderwater, diep anoxisch, verzoet, hypertroof	F_4CaHCO_3+, F_4-F_5MgCO_3+, F_4-F_5NaHCO_3+
P_{fhr}	Polderwater, verzoet, hypertroof, gereduceerd	$F_3CaMix+$, F_3CaHCO_3+, B_4NaCl+
DUINWATER		
D	Duinwater, (sub)oxisch, eutroof, zonder basenuitwisseling	F_2CaMix, F_2CaHCO_3
D_d	Duinwater, diep anoxisch, eutroof, zonder basenuitwisseling	F_3CaHCO_3
D_{dfh}	Duinwater, diep anoxisch, verzoet, hypertroof	F_3-F_4CaHCO_3+, F_3-F_4MgHCO_3+, F_3-F_5NaHCO_3+
D_{dh}	Duinwater, diep anoxisch, hypertroof, zonder basenuitwisseling	F_3CaHCO_3
D_{fr}	Duinwater, verzoet, gereduceerd, eutroof	F_3MgCl+, F_3NaCl+
D_{hr}	Duinwater, hypertroof, gereduceerd, zonder basenuitwisseling	F_2-F_3CaHCO_3
D_m	Duinwater, dystroof-mesotroof, (sub)oxisch, zonder basenuitwisseling	F_2CaMix, F_2CaHCO_3
D_r	Duinwater, gereduceerd, eutroof, zonder basenuitwisseling	F_2CaMix, F_2-F_3CaHCO_3
D_{rs}	Duinwater, gereduceerd, verzilt, eutroof	F_2CaCl-, F_2NaCl-
BRAKKE EN ZOUTE GRONDWATEREN, ALS PURE END-MEMBERS		
L_{dhs}	relict, Holoceen transgressiewater, diep anoxisch, hypertroof, verzilt	B_3-B_4CaCl-, B_3-B_5NaCl-
L_{hrs}	relict, Holoceen transgressiewater, hypertroof, gereduceerd, verzilt	S_3NaCl-
M_{dhs}	connaat, Maassluiswater, diep anoxisch, hypertroof, verzilt	B_3NaCl-, S_3NaCl-
S_r	Noordzeewater, gereduceerd, eutroof, zonder basenuitwisseling	S_2-S_3NaCl
S_{rs}	Noordzeewater, gereduceerd, verzilt, eutroof	S_2-S_3NaCl-
MENGWATEREN, ZOWEL ZOET ALS BRAK		
$(AM/AR)_r$	zoet, Maas- en Rijnwater, gereduceerd, eutroof, zonder basenuitwisseling	F2CaMix, F_2CaHCO_3
$(AR/D/S)_{fr}$	brak, Rijn-, duin- en Noordzeewater, verzoet, gereduceerd, eutroof	B_3NaCl+
$(D/L)_{dfh}$	zoet, duin- en polderw., verzoet, eutroof, gereduceerd	B_4NaCl+
$(D/P)_{fr}$	zoet, duin- en polderw., verzoet, gereduceerd, eutroof	F_3-F_4CaHCO_3+
$(D/P)_{dfh}$	zoet, duin- en polderw., diep anoxisch, verzoet, hypertroof	F_4CaHCO_3+, F_4NaHCO_3+
$(D/P)_{fhr}$	zoet, duin- en polderw., verzoet, hypertroof, gereduceerd	F_3CaHCO_3+
$(D/P)_r$	zoet, duin- en polderw., gereduceerd, eutroof, zonder basenuitw.	F_3CaHCO_3, F_3CaMix
$(D/S)_{rs}$	brak, duin- en Noordzeewater, gereduceerd, verzilt, eutroof	B_2-B_3CaCl-, B_2-B_3NaCl-, S_2-S_3NaCl-
$(P/L)_{dfh}$	brak, polder- en transgressiew., diep anoxisch, verzoet, hypertroof	B_5NaCl+
$(S/L)_{hrs}$	brak-zout, Noordzee- en transgressiew., hypertroof, gereduceerd, verzilt	B_3NaCl-, S_2-S_3NaCl-

Legend by (Stuyfzand, 1993). In Dutch

8. Appendix C: Intermediate calibration results of infiltration pond conductance

Calibrate conductance

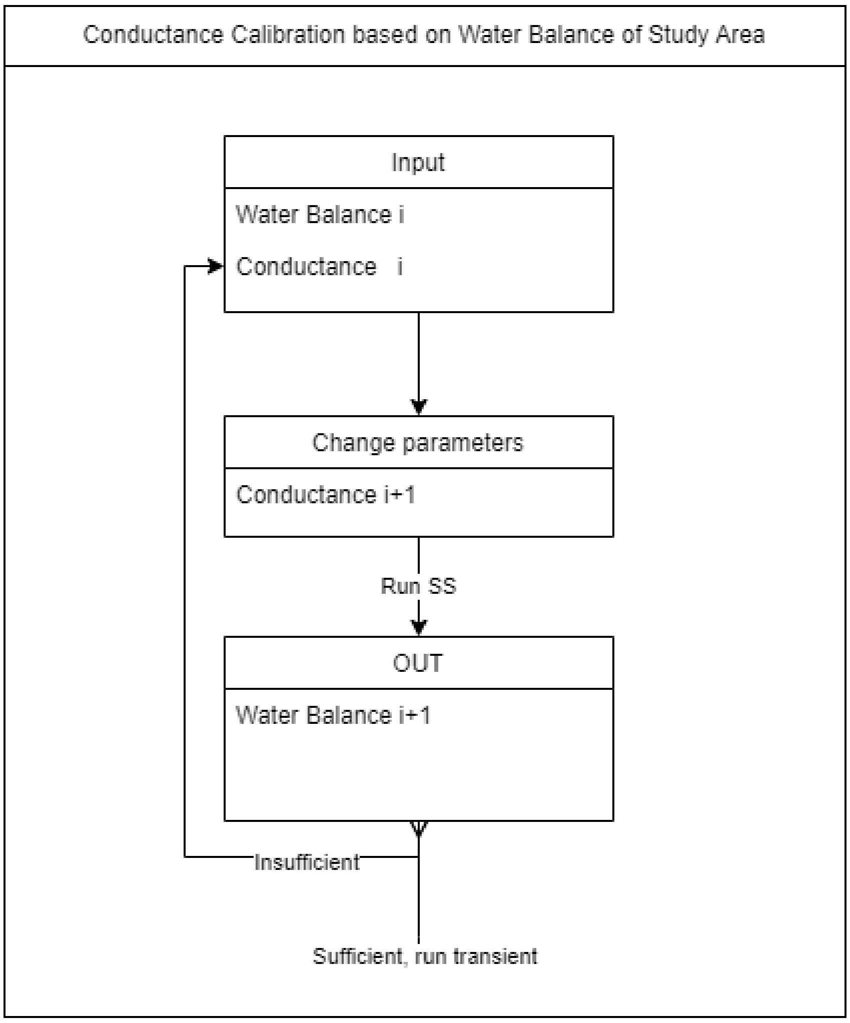
AUTHOR

Justus Krantz

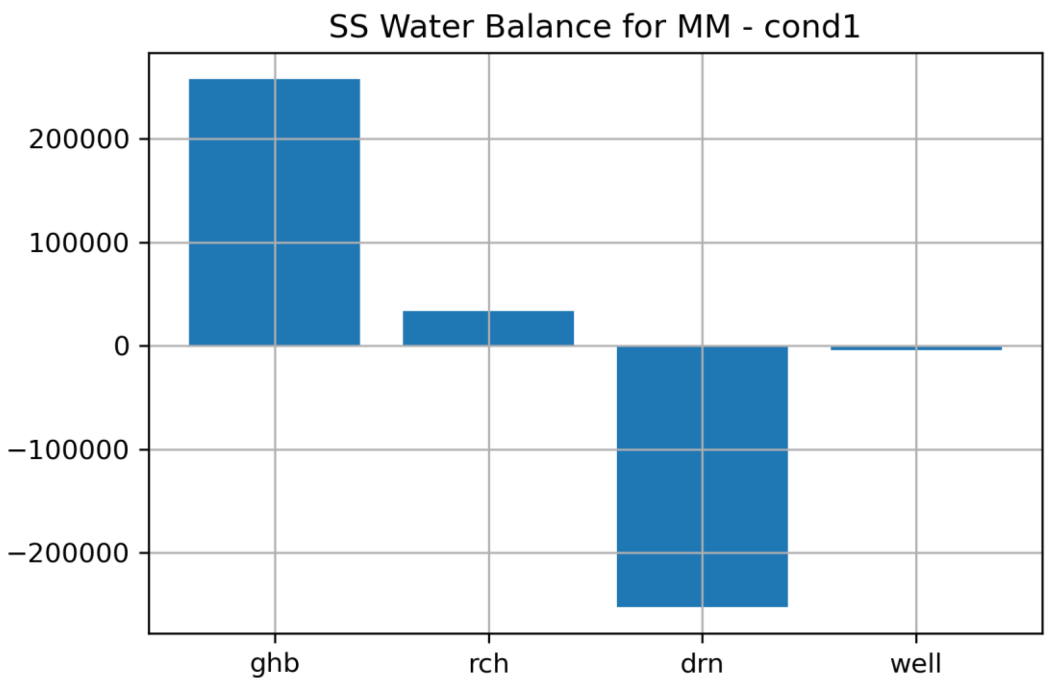
Introduction

Before calibration on water balance in the study area, the starting conductance is calculated by the water balance for the infiltration ponds themselves: By taking heads and budget changes (flows) from OM and dividing the two, giving cell conductances. This conductance dataset will be used as input for the calibration. Conductance is calculated by taking heads and budget changes (flows) from OM. dividing the two gives cell conductances.

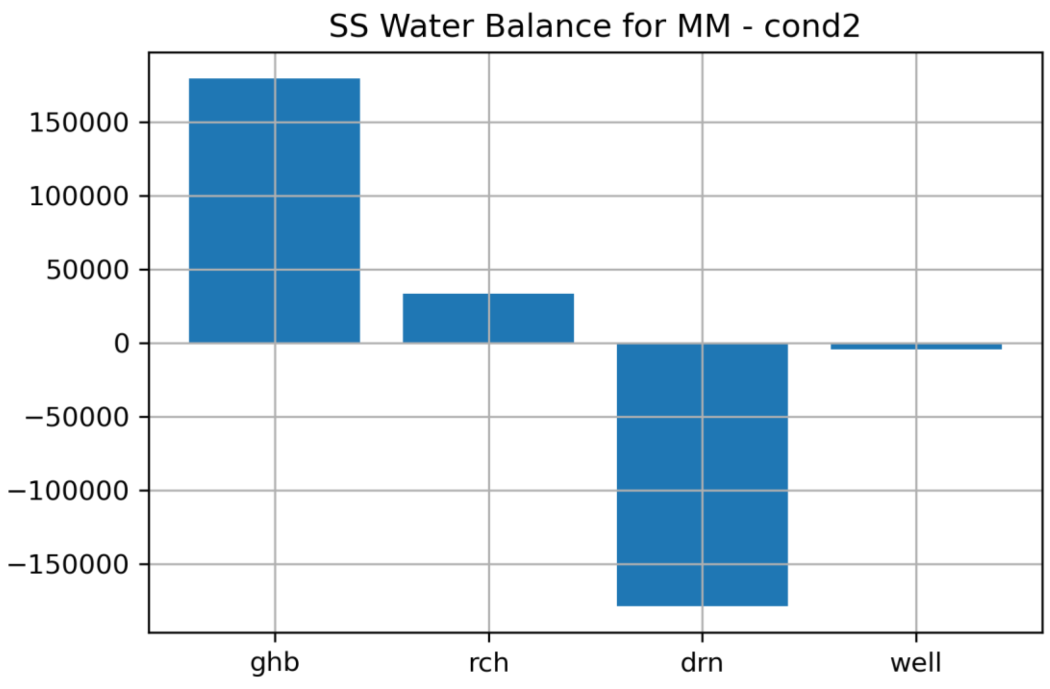
The following diagram shows the calibration procedure:



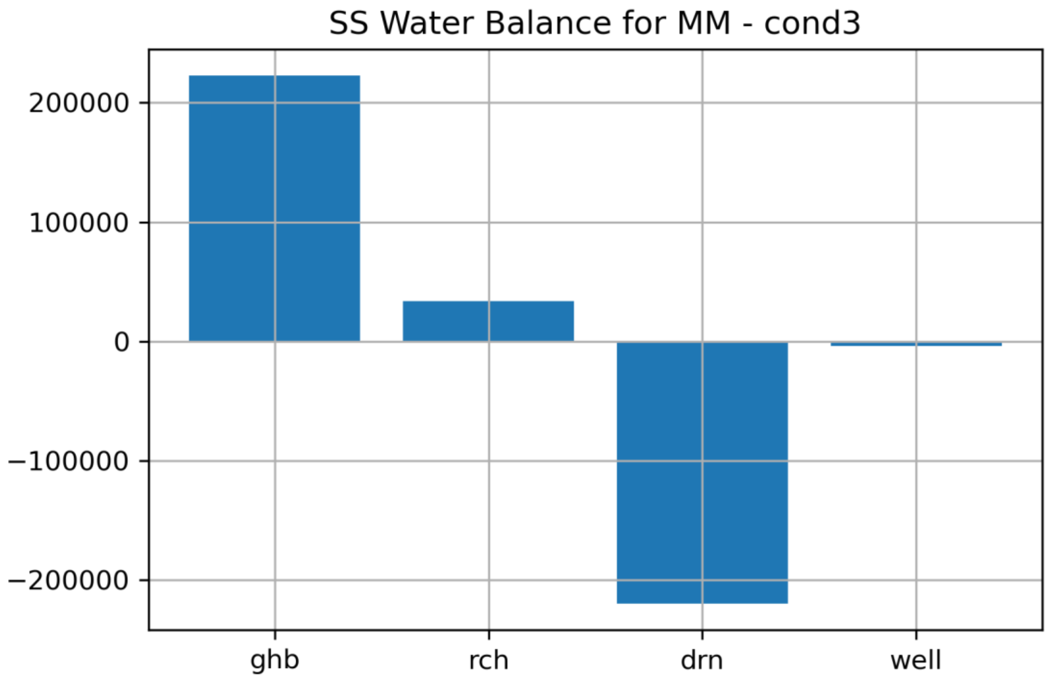
Output - Water Balances



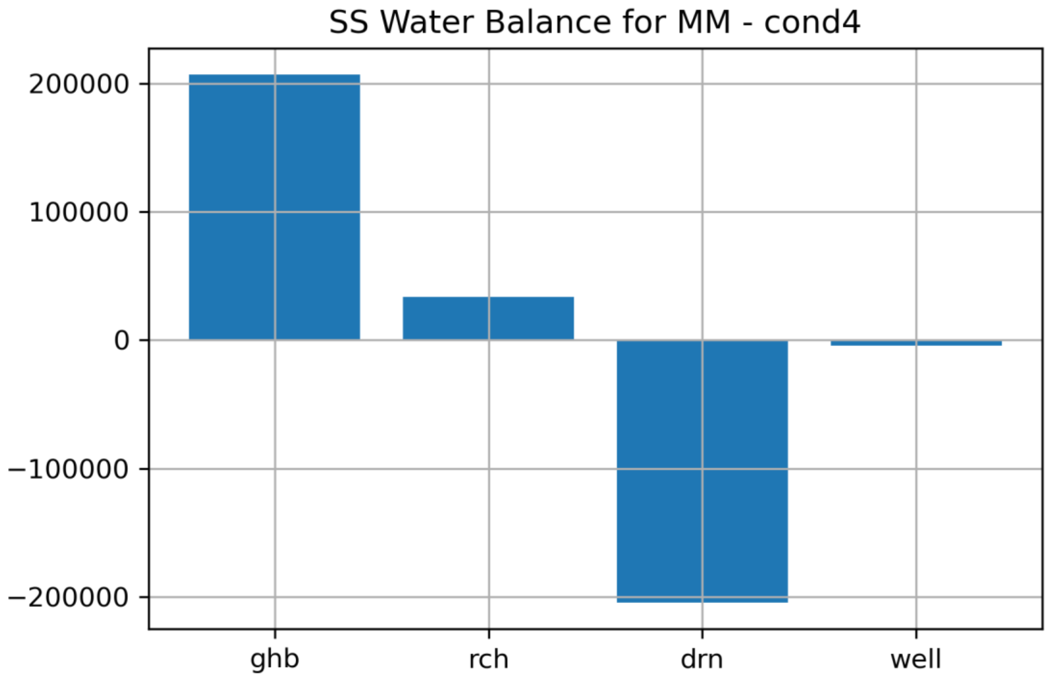
MM Water balance over study area, cond1



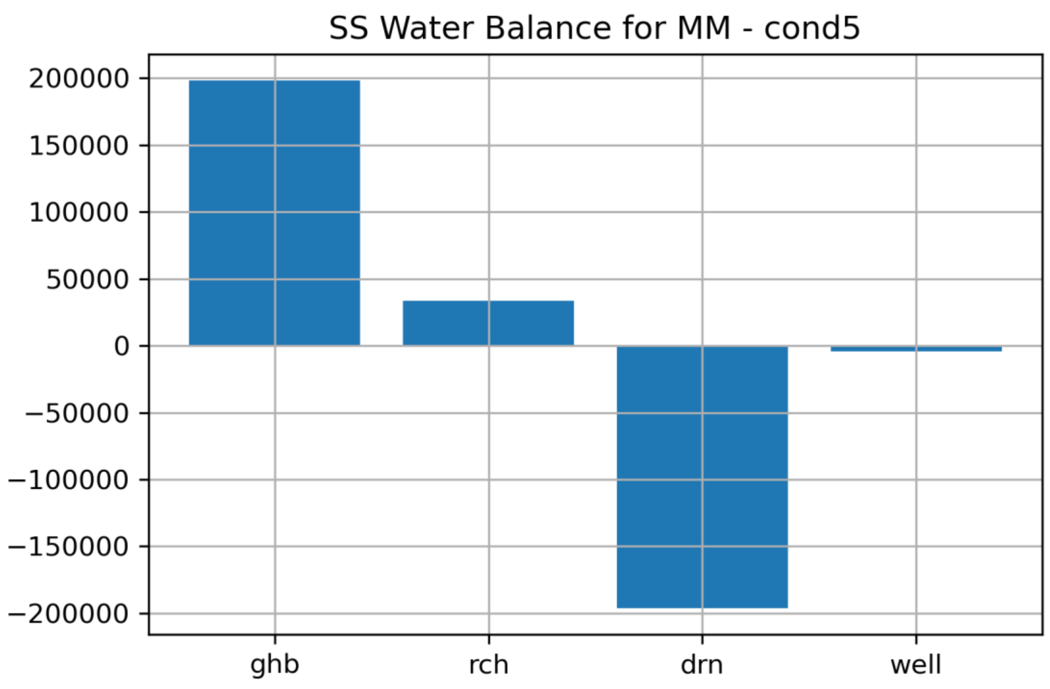
OM Water balance over study area, cond2



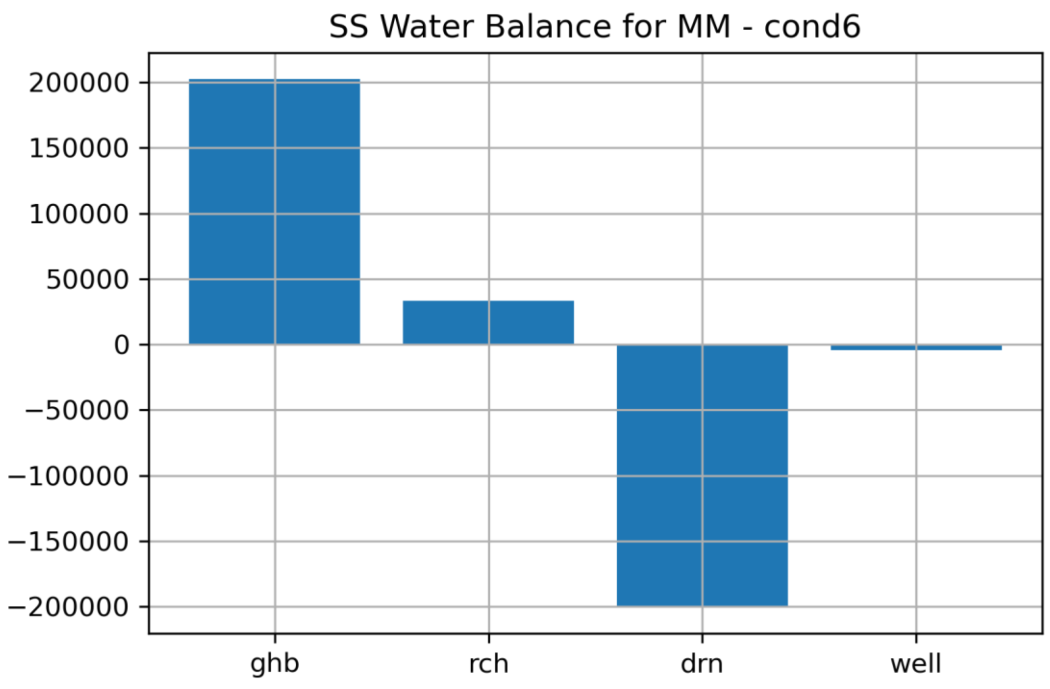
OM Water balance over study area, cond3



OM Water balance over study area, cond4



OM Water balance over study area, cond5



OM Water balance over study area, cond5

Output - Observations

OM:

- IP: 202230.78 m^3/d
- drn: -193343 m^3/d

MM(cond1)

- IP: 28.000 m^3/d
- drn: -26000 m^3/d

MM(cond2)

- cond2 = 0.5 * cond1
- IP: 18.000 m^3/d
- drn: -17000 m^3/d

MM(cond3)

- cond3 = 0.75 * cond1
- IP: 222650.48 m^3/d
- drn: -219553.22 m^3/d

MM(cond4)

- cond4 = 0.65 * cond1
- IP: 206675.44 m^3/d
- drn: -204291.1 m^3/d

MM(cond5)

- cond5 = 0.60 * cond1
- IP: 198116.83 m^3/d
- drn: -196130.66 m^3/d

MM(cond5)

- cond5 = 0.625 * cond1
- IP: 202450.4 m^3/d
- drn: -200261.34 m^3/d

Calibrated

To calibrate on the water balance, the conductances of the infiltration ponds in the study area need to be scaled by a factor of 0.625.

9. Appendix D: Model scripts

The following scripts follow the structure discussed in Section 2.1.6

src\1-prepare\1.1-create-like.py

```python
# %%
"""
Create empty LIKE grid of 250m, which is the grid to which all data will be regridded.

"""
#%%
import imod
import numpy as np
import os
import xarray as xr
#%%
os.chdir("c:/projects/msc-thesis")
# %%

def round_extent(extent, cellsize):
    """Increases the extent until all sides lie on a coordinate
    divisible by cellsize."""
    xmin, ymin, xmax, ymax = extent
    xmin = np.floor(xmin / cellsize) * cellsize
    ymin = np.floor(ymin / cellsize) * cellsize
    xmax = np.ceil(xmax / cellsize) * cellsize
    ymax = np.ceil(ymax / cellsize) * cellsize
    return xmin, ymin, xmax, ymax

# %%
template = xr.open_dataarray("data/1-external/template.nc")

x_old = template.coords["x"]
y_old = template.coords["y"]

dx_1 = 250
xmin_1 = x_old[0]
xmax_1 = x_old[-1]


dy_1 = -250
ymin_1 = y_old[-1]
ymax_1 = y_old[0]

xmin_1, ymin_1, xmax_1, ymax_1 = round_extent((xmin_1, ymin_1, xmax_1, ymax_1), dx_1)

like = imod.util.empty_2d(
    dx_1,
    xmin_1,
    xmax_1,
    dy_1,
    ymin_1,
    ymax_1,
)
like_ds = like.to_dataset(name=("like"))

like_ds.to_netcdf("data/2-interim/like.nc")

# %%
```

src\1-prepare\1.2-create_lpf_bas.py

```python
"""
Create the content of the LPF and BAS packages.

Steps in this script:

* Read 25 m data
* Regrid IBOUND data to 250 m: Only make 250 m cells active if more than half of
  volume is occupied by 25 m active cells
* Regrid conductivity parameters...
* Set up BasicFlow model
* Set up LayerPropertyFlow model

For the starting heads
* Heads calculated by the 25m model are used (SS run)
* Heads from output of previous run can be used
"""

#%%
import scipy.ndimage
import imod
import numpy as np
import os
import pandas as pd
import xarray as xr
#%%
os.chdir("c:/projects/msc-thesis")

# %%
# Open data
template      = xr.open_dataarray("data/1-external/template_2d.nc")
starting_head = xr.open_zarr(r"data\1-external\data-25-run-1\head_ss_t0.zarr")
starting_head_ar = starting_head["head"].astype(np.float64)
#starting_head_2 = starting_head_ar.to_netcdf()
starting_head_3 = starting_head_ar.swap_dims({"layer":"z"}).drop("time")
starting_head_nc = xr.open_dataarray("data/1-external/starting-head.nc") # For calibration purposes, the model should be run using the same starting heads, but regridded.
conductivity  = xr.open_dataset("data/1-external/conductivity.nc")
like = xr.open_dataarray("data/2-interim/like.nc")

kh = conductivity["kh"]
kv = conductivity["kv"]

#%%
# preparing regridders
sum_regridder      = imod.prepare.Regridder(method="sum", use_relative_weights=True)
mean_regridder     = imod.prepare.Regridder(method="mean")
harmonic_regridder = imod.prepare.Regridder(method="harmonic_mean")
#%%
# Robust method for linking z, dz and layer
def link_z_layer(ds, ibound):
    z = ibound["z"].values
    dz =ibound["dz"].values
    layer = ibound["layer"].values
    lookup1 = {key: value for key, value in zip(z, layer)}
    lookup2 = {key: value for key, value in zip(z, dz)}
    layer_numbers   = [lookup1[z] for z in ds["z"].values]
    layer_thickness = [lookup2[dz] for dz in ds["z"].values]
    return ds.assign_coords(layer=("z", layer_numbers),dz = ("z", layer_thickness))

#%%

# ibound
domain2d = starting_head_nc.isel(z=-1, drop =True).notnull()
ibound = kh.notnull() & domain2d

ibound_re = sum_regridder.regrid(ibound,like)
ibound_coarse = ibound_re > 50
ibound_coarse = (ibound_coarse
    .assign_coords(layer =("z", np.arange(1,50)))
    .assign_coords(dz=kh["dz"]).drop(["dx", "dy"])
    .assign_coords(dx=like["dx"], dy=like["dy"])
)

ibound_coarse_ds = ibound_coarse.to_dataset(name="ibound_coarse")
ibound_coarse_ds.to_netcdf("data/2-interim/ibound_coarse.nc")
#%%
# Conductivity
kh_coarse = mean_regridder.regrid(kh,like)

kh_coarse = (kh_coarse
    .where(ibound_coarse)
    .assign_coords(layer=("z", np.arange(1, 50)))
    .assign_coords(dz=kh["dz"]).drop(["dx", "dy"])
)
kv_coarse = harmonic_regridder.regrid(kv,like)

kv_coarse = (kv_coarse
    .where(ibound_coarse)
    .assign_coords(layer=("z", np.arange(1, 50)))
    .assign_coords(dz=kh["dz"]).drop(["dx", "dy"])
)

#%%
# Starting head
SH_re = mean_regridder.regrid(starting_head_nc, like)
SH_re_2 = mean_regridder.regrid(starting_head_3, like)
SH_re = SH_re.where(ibound_coarse)
#SH_re_2 = link_z_layer(SH_re,ibound_coarse)
SH_re.to_netcdf(r"data/2-interim/starting-head-coarse.nc")

#%%
# Basic Flow
bas = imod.wq.BasicFlow(
    ibound = ibound_coarse,    #must be dataarray
    top    = ibound.coords["ztop"][0],
    bottom = ibound.coords["zbot"].zbot,
    starting_head = SH_re,
    inactive_head = -9999.0
)
bas.dataset.to_netcdf("data/3-input/bas.nc")
#%%
# Layer Property Flow
lpf = imod.wq.LayerPropertyFlow(k_horizontal = kh_coarse, k_vertical =
kv_coarse,save_budget = True)
lpf.dataset.to_netcdf("data/3-input/lpf.nc")

```

```
# %%

```

src\1-prepare\1.3-create_recharge.py

```python
#%%
"""
Create recharge package after regridding. expand concentration dimension with species
"""
#%%
import scipy.ndimage
import imod
import numpy as np
import os
import pandas as pd
import xarray as xr
#%%
os.chdir("c:/projects/msc-thesis")

# %%
# Open data
recharge      = xr.open_dataarray("data/1-external/recharge.nc")
like          = xr.open_dataarray("data/2-interim/like.nc")
ibound_coarse = xr.open_dataarray("data/2-interim/ibound_coarse.nc")

#%%
# prepare regridders
mean_regridder = imod.prepare.Regridder(method = "mean")

#%%
# Regrid recharge
recharge_coarse  = mean_regridder.regrid(recharge,like)
recharge_mean    = recharge_coarse.mean("layer").mean("time") # recharge on inactieve cells
maakt niet uit
#%%
# expand concentration dimension into species dimension
y = ibound_coarse["y"]
x = ibound_coarse["x"]
cond = recharge_mean

conc = xr.full_like(cond, 0.0).where(cond.notnull()) # [Cl] = 0?
species_nd = xr.concat([
    conc.assign_coords(species=1),  # cl
    xr.full_like(conc, 0.0).where(conc.notnull()).assign_coords(species=2),  # AM
    xr.full_like(conc, 0.0).where(conc.notnull()).assign_coords(species=3)], # polders
    dim="species")

 #%%
rch =  imod.wq.RechargeHighestActive(rate=recharge_mean, concentration=species_nd,
save_budget=True)
rch.dataset.to_netcdf("data/3-input/rch.nc")

# %%

```

src\1-prepare\1.3-create_riv.py

```python
"""
Create river and infiltration ponds onjects
* Data imported
* Moving average for removing unwanted "boezem" effect
* Species added, to be used for replicating Stuyfzand's geochemical analyses
    conc      (chloride conc)       = Cl
    river     (Artificial Polder)   = AP
    inf_ponds (Artificial Meuse)    = AM

Notes
- River is also saved as a river package in this script, but currently unused
- It's saved as a netcdf in this script, which is used in 1.5-create-ghb

"""
# %%
import os
import imod
import numpy as np
import scipy.ndimage
import xarray as xr
import numpy as np
# %%
os.chdir("c:/projects/msc-thesis")
# %%
# Read input data.
like          = xr.open_dataarray("data/2-interim/like.nc")
ibound_coarse = xr.open_dataset("data/2-interim/ibound_coarse.nc")
river         = xr.open_dataset("data/1-external/river.nc")
conc          = xr.open_dataarray("data/2-interim/chloride_coarse.nc")
# %% Functions
# moving average to remove boezems
def moving_average(da, windowsize: int):
    weights = np.ones((windowsize, windowsize))
    weights = weights / weights.sum()  # sums to 1
    out = da.copy()
    scipy.ndimage.convolve(da.values, weights, out.values)
    return out
# Robust method for linking z, dz and layer
def link_z_layer(ds, ibound):
    z = ibound["z"].values
    dz =ibound["dz"].values
    layer = ibound["layer"].values
    lookup1 = {key: value for key, value in zip(z, layer)}
    lookup2 = {key: value for key, value in zip(z, dz)}
    layer_numbers   = [lookup1[z] for z in ds["z"].values]
    layer_thickness = [lookup2[dz] for dz in ds["z"].values]
    return ds.assign_coords(layer=("z", layer_numbers),dz = ("z", layer_thickness))

# %%
# process data: find mean and filter "boezems" out
river_mean     = river.mean("time", skipna=True)
river_stage    = river_mean["stage"].max("z")
river_full     = imod.prepare.fill(river_stage)
moving_avg_riv = moving_average(river_full,11)

keep = (river_full - moving_avg_riv) < 1.0
keep = keep | (keep["y"] > 462_500.0)
filtered_riv_ds = river_mean.where(keep)
#%%
# Regridders
mean_regridder = imod.prepare.Regridder(method="mean")
cond_regridder = imod.prepare.Regridder(method="conductance")

river_regridded = xr.Dataset()
for var in ("stage", "cond", "bot", "density"):
    river_regridded[var] = mean_regridder.regrid(filtered_riv_ds[var], like=like)
# ink z and layers
river_linked  = link_z_layer(river_regridded, ibound_coarse)
# %%
# Add concentration[species] dimension to river
y = ibound_coarse["y"]
x = ibound_coarse["x"]

cond = river_linked["cond"]

conc_polders = xr.full_like(cond, 0.0).where(cond.notnull()) # [Cl] = 0?
species_nd = xr.concat([
    conc_polders.assign_coords(species=1), #cl
    xr.full_like(conc_polders, 0.0).where(conc_polders.notnull()).assign_coords(species=2), # AM
    xr.full_like(conc_polders, 1.0).where(conc_polders.notnull()).assign_coords(species=3)], # polders
    dim="species")

river_linked["conc"] = species_nd
river_linked.to_netcdf("data/2-interim/river.nc")
#%% UNUSED
#riv = imod.wq.River(stage                   = river_linked["stage"],
#                     conductance           = river_linked["cond"],
#                     bottom_elevation      = river_linked["bot"],
#                     density               = river_linked["density"],
#                     concentration         = river_linked["conc"],
#                     save_budget           = True
#)

# %%

```

src\1-prepare\1.4.1-find-cond.py

```python
# %%
"""
- Find the conductances for polders and infiltration ponds of MM, using data from riv OM
    - Also calculate infiltration ponds conductance here using 25m data
    - Infiltration ponds conductance is saved also.
        Note that conductance here is inconsistent with other infiltration ponds dataarrays["stage"] etc.
        The inf_ponds[cond] dataarray has a NaN where the other inf_ponds are active. fixed in 1.5
- To be used in 1.5-create-ghb
"""
#%%
import imod
import numpy as np
import os
import xarray as xr
import matplotlib.pyplot as plt
#%%
os.chdir("c:/projects/msc-thesis")
#%%
# Open input data 25m model
riv_25_stage  = imod.idf.open(r"c:\projects\msc-thesis\DUNEA_1\3- input-dunea_transient\riv\stage_19791231235959_l*.idf") # First date is 1s before next, is the mean
riv_25_budget = xr.open_zarr(r"DUNEA_1\4-output\bdgriv_ss_t0.zarr") # budget file is SS
riv_25_bdg_da = riv_25_budget["bdgriv"]
riv_25_cond   = imod.idf.open(r"c:\projects\msc-thesis\DUNEA_1\3- input-dunea_transient\riv\conductance_l*.idf").astype(np.float64)
inf_ponds     = xr.open_dataset(r"data/1-external/infiltration_ponds.nc").mean("time", skipna = True)
ibound_coarse = xr.open_dataarray(r"data/2-interim/ibound_coarse.nc")

SH_25         = imod.idf.open(r"c:\projects\msc-thesis\DUNEA_1\3- input-dunea_transient\bas\starting_head_l*.idf").astype(np.float64)
H_SS_25       = xr.open_zarr(r"c:\projects\msc-thesis\DUNEA_1\4-output\head_ss_t0.zarr")["head"].astype(np.float64)

like = xr.open_dataarray("data/2-interim/like.nc").astype(np.float64)
# %% Regrid stage and budget
sum_regridder  = imod.prepare.Regridder(method="sum", use_relative_weights=True)
mean_regridder = imod.prepare.Regridder(method="mean")

# steady state heads from Dunea
cell_SS_25 =  H_SS_25.swap_dims({"layer":"z"}).drop("time")
H_SS_250       = mean_regridder.regrid(cell_SS_25, like)

# Polders
r_25_s = riv_25_stage.swap_dims({"layer": "z"})
r_25_b = riv_25_bdg_da.swap_dims({"layer": "z"})
cell_SS_25 =  H_SS_25.swap_dims({"layer":"z"}).drop("time")
riv_250_stage  = mean_regridder.regrid(r_25_s, like).mean("time")
riv_250_budget = sum_regridder.regrid(r_25_b,  like)

# Inf_ponds except conductance
inf_ponds_stage_re = mean_regridder.regrid(inf_ponds["stage"], like)

#%% Calculate cond using bdg and stage
# rivers conductance
```

```python
h_cell_250 = H_SS_250.where(riv_25_cond.notnull().any())
cond_250   = riv_250_budget / (riv_250_stage - h_cell_250)

cond_250.to_netcdf("data/2-interim/cond_250_polders.nc")

# infiltration ponds conductance
h_cell_inf_ponds   = H_SS_250.where(inf_ponds["is_pond_2D"].notnull().any())
cond_250_inf_ponds = riv_250_budget / (inf_ponds_stage_re - h_cell_inf_ponds)
cond_250_ip_ib = cond_250_inf_ponds.where(ibound_coarse.notnull())

#cond_250_inf_ponds.to_netcdf("data/2-interim/cond_250_inf_ponds.nc")
cond_250_ip_ib.to_netcdf("data/2-interim/cond_250_inf_ponds.nc")


# %%

```

src\1-prepare\1.4.2-calibrate-cond.py

```python
"""
- In this script, the cond of infiltration ponds will be calibrated.
- The calibration will be done based on budgets (4.2)
- Performing a SS run of 1s to bring the budgets of OM and MM closer together
- After sucessful calibration, this cond can be used to study the effect
  on the fresh-saline interface depth in (4.5)
- Calibration steps:
    - cond1         = cond as calculated by 1.4 - based on OM budgets and heads.
    - cond2         = cond1/2
    - cond3         = cond1*0.75
    - cond4         = cond1*0.55
    - cond5         = cond1*0.65
    - cond6         = cond1*0.625
    - cond_final    = conductance as calibrated.  = cond6
- cond_final is used in (1.5-create-ghb)
"""
#%%
import imod
import numpy as np
import os
import xarray as xr
import matplotlib.pyplot as plt
os.chdir("c:/projects/msc-thesis")
# %% Data
# import
cond1_IP = xr.open_dataarray(r"data/2-interim/cond_250_inf_ponds.nc")
# Process
cond2_IP = 0.5*cond1_IP
cond3_IP = 0.75*cond1_IP
cond4_IP = 0.65*cond1_IP
cond5_IP = 0.60*cond1_IP
cond6_IP = 0.625*cond1_IP

# %% Save data
cond2_IP.to_netcdf("data/2-interim/cond2_IP.nc")
cond3_IP.to_netcdf("data/2-interim/cond3_IP.nc")
cond4_IP.to_netcdf("data/2-interim/cond4_IP.nc")
cond5_IP.to_netcdf("data/2-interim/cond5_IP.nc")
cond6_IP.to_netcdf("data/2-interim/cond6_IP.nc")

# %%

```

src\1-prepare\1.4-create_btn_dsp_adv_vdf.py

```python
#%%
"""
Create:
- Basic Transport
- Dispersion
- AdvectionTVD
- Variable Density Flow

 *  This script uses the same input for starting concentrations as the original model,
    But regridded
"""
#%%
import numpy as np
import imod
import os
import xarray as xr
#%%
os.chdir("c:/projects/msc-thesis")
#%%
# Open data
ibound_coarse = xr.open_dataarray("data/2-interim/ibound_coarse.nc")
chloride  = xr.open_dataarray("data/1-external/chloride.nc")
like      = xr.open_dataarray("data/2-interim/like.nc")
inf_ponds = xr.open_dataset(r"data/1-external/infiltration_ponds.nc")

# Open output data, to be used as input for future runs
c1_2054 = imod.idf.open(r"data\4-output\conc\conc_c1_205412312359_l*.IDF") # Cl
c2_2054 = imod.idf.open(r"data\4-output\conc\conc_c2_205412312359_l*.IDF") # AM
c3_2054 = imod.idf.open(r"data\4-output\conc\conc_c3_205412312359_l*.IDF") # Polders

# Use the same concentrations that are used by Dunea
conc_25m  = xr.open_zarr(r"data\1-external\data-25-run-1\conc-selection.zarr")
["conc"].astype(np.float64)
c1_t0_25m = conc_25m.isel(time=0, drop=True)
c1_t0_25m = c1_t0_25m.swap_dims({"layer":"z"})
#%%
mean_regridder = imod.prepare.Regridder(method="mean")

#%%
chloride_coarse = mean_regridder.regrid(c1_t0_25m, like)
chloride_new    = chloride_coarse.where(ibound_coarse)
chloride_new    = chloride_new.assign_coords(layer=("z", np.arange(1,50)))

chloride_fresh  = chloride_new.notnull() * chloride_new.min()
chloride_saline = chloride_new.notnull() * chloride_new.max()

chloride = chloride_new

species_nd = xr.concat([
    chloride.assign_coords(species=1), #cl
    xr.full_like(chloride, 0.0).where(chloride.notnull()).assign_coords(species=2),  # AM
    xr.full_like(chloride, 0.0).where(chloride.notnull()).assign_coords(species=3)], #
polders
    dim="species")
#%%

```

```python
species_out = xr.concat([
    c1_2054.squeeze("time"),
    c2_2054.squeeze("time"),
    c3_2054.squeeze("time")],
    dim="species")
species_out = species_out.swap_dims({"layer":"z"}).drop("time")
# output contains a negative value, erase it?
#species_positive = species_out["c1"] > 0
#ds_negative = species_out["c1"] < 0
#species_out["c1"] = species_out["c1"].where(~ds_negative, other=0.0)


#%%
btn = imod.wq.BasicTransport(icbund=ibound_coarse, starting_concentration=species_nd,
n_species=3)
dsp = imod.wq.Dispersion(longitudinal=0.001)
adv = imod.wq.AdvectionTVD(courant=1.0)
vdf = imod.wq.VariableDensityFlow(density_concentration_slope=0.71)

#%%
# Save datasets
chloride_coarse.to_netcdf("data/2-interim/chloride_coarse.nc")
btn.dataset.to_netcdf("data/3-input/btn.nc")
dsp.dataset.to_netcdf("data/3-input/dsp.nc")
adv.dataset.to_netcdf("data/3-input/adv.nc")
vdf.dataset.to_netcdf("data/3-input/vdf.nc")
# %%

```

src\1-prepare\1.5.1-create-chd.py

```python
"""
- Create the constand head boudnary in a separate script
- Assign species to this boundary condition
"""
#%%

import imod
import os
import xarray as xr
import numpy as np
import scipy.ndimage
import matplotlib.pyplot as plt
#%%
os.chdir("c:/projects/msc-thesis")
#%%
# Open data
inf_ponds     = xr.open_dataset(r"data/1-external/infiltration_ponds.nc").drop("is_pond_2D")
ibound_coarse = xr.open_dataarray(r"data/2-interim/ibound_coarse.nc")
shead_coarse  = xr.open_dataarray(r"data/2-interim/starting-head-coarse.nc")
sconc_c1  = imod.idf.open(r"c:\projects\msc-thesis\data\3-input\SS_1\btn\starting_concentration_c1*.idf")
sconc_c2  = imod.idf.open(r"c:\projects\msc-thesis\data\3-input\SS_1\btn\starting_concentration_c2*.idf")
sconc_c3  = imod.idf.open(r"c:\projects\msc-thesis\data\3-input\SS_1\btn\starting_concentration_c3*.idf")

#%%
# Functions
# Find the boundary of the lowest layer
def find_boundary(ibound):
    deepest_ibound = ibound.isel(z=-1, drop=True)
    eroded = deepest_ibound.copy(data=scipy.ndimage.binary_erosion(deepest_ibound.values))
    is_boundary = (deepest_ibound == 1) & (eroded == 0)
    return is_boundary

# Robust method for linking z, dz and layer
def link_z_layer(ds, ibound):
    z = ibound["z"].values
    dz =ibound["dz"].values
    layer = ibound["layer"].values
    lookup1 = {key: value for key, value in zip(z, layer)}
    lookup2 = {key: value for key, value in zip(z, dz)}
    layer_numbers   = [lookup1[z] for z in ds["z"].values]
    layer_thickness = [lookup2[dz] for dz in ds["z"].values]
    return ds.assign_coords(layer=("z", layer_numbers),dz = ("z", layer_thickness))
#%%
# Boundary
boundary = find_boundary(ibound_coarse)
shead_bound = shead_coarse
#chloride = xr.full_like(shead_bound, 16.048)
chloride_2 = sconc_c1
species_nd = xr.concat([
    chloride_2.assign_coords(species=1), #cl
    xr.full_like(chloride_2, 0.0).assign_coords(species=2).where(chloride_2.notnull()),  # AM
    xr.full_like(chloride_2, 0.0).assign_coords(species=3).where(chloride_2.notnull())], #
polders
    dim="species")

ds_bound = xr.Dataset()
ds_bound["stage"]    = shead_bound
ds_bound["conc"]     = species_nd
ds_bound["cond"]     = xr.full_like(shead_bound, 5000.0)     # High value
ds_bound = ds_bound.where(boundary)                           # may be needed to add again if
error!

ds_linked = link_z_layer(ds_bound, ibound_coarse)
#%%
chd_out = imod.wq.ConstantHead(
    ds_linked["stage"],
    ds_linked["stage"],
    ds_linked["conc"],
    )
chd_out.dataset.to_netcdf("data/3-input/chd.nc")
# %%

```

src\1-prepare\1.5.2-create-ghb.py

```python
 1  #%%
 2  """
 3  Create General Head Boundary package GHB as a combination of:
 4  - infiltration ponds (from 1.3)
 5  - river (from 1.3)
 6  - 1.4-find-conductances for polder conductance and infiltration ponds
 7      - as stated in the intro of 1.3, the conductance as calculated by the budgets has inconsistend NaN with other dataarrays in inf_ponds dataset.
 8      - This inconsistency is resolved by a .where() command
 9      - see marked line 67
10  - 1.4.2-calibrate-cond NOTE
11      - see marked line 67
12  - sea
13  """
14  #%%
15  import imod
16  import os
17  import pandas as pd
18  import xarray as xr
19  import numpy as np
20  import scipy.ndimage
21  import matplotlib.pyplot as plt
22  #%%
23  os.chdir("c:/projects/msc-thesis")
24  # %%
25  # Open data
26  shead          = xr.open_dataarray(r"data/1-external/starting-head.nc")
27  like           = xr.open_dataarray(r"data/2-interim/like.nc")
28  ibound_coarse = xr.open_dataarray(r"data/2-interim/ibound_coarse.nc")
29  inf_ponds      = xr.open_dataset(r"data/1-external/infiltration_ponds.nc").drop("is_pond_2D")
30  shead_coarse   = xr.open_dataarray(r"data/2-interim/starting-head-coarse.nc")
31  rivers         = xr.open_dataset(r"data/2-interim/river.nc")
32  sea            = xr.open_dataset(r"data/1-external/sea.nc")
33  cond_polders   = xr.open_dataarray(r"data/2-interim/cond_250_polders.nc")    # see 1.4
34  cond_ponds     = xr.open_dataarray(r"data/2-interim/cond_250_inf_ponds.nc")  # see 1.4
35  cond2_IP       = xr.open_dataarray(r"data/2-interim/cond2_IP.nc")
36  cond3_IP       = xr.open_dataarray(r"data/2-interim/cond3_IP.nc")
37  cond4_IP       = xr.open_dataarray(r"data/2-interim/cond4_IP.nc")
38  cond5_IP       = xr.open_dataarray(r"data/2-interim/cond5_IP.nc")
39  cond6_IP       = xr.open_dataarray(r"data/2-interim/cond6_IP.nc")
40
41  # Robust method for linking z, dz and layer
42  def link_z_layer(ds, ibound):
43      z = ibound["z"].values
44      dz =ibound["dz"].values
45      layer = ibound["layer"].values
46      lookup1 = {key: value for key, value in zip(z, layer)}
47      lookup2 = {key: value for key, value in zip(z, dz)}
48      layer_numbers   = [lookup1[z] for z in ds["z"].values]
49      layer_thickness = [lookup2[dz] for dz in ds["z"].values]
50      return ds.assign_coords(layer=("z", layer_numbers),dz = ("z", layer_thickness))
51
52  # %%
53  # Sea
54  cond_regridder = imod.prepare.Regridder(method= "conductance")
```

```python
mean_regridder = imod.prepare.Regridder(method= "mean")

ds = xr.Dataset()
for var in ("stage", "conc", "density"):
    ds[var] = mean_regridder.regrid(sea[var], like=like)
ds["cond"] = cond_regridder.regrid(sea["cond"], like=like)

ds_clipped = xr.Dataset()
for var in ("stage", "conc", "density", "cond"):
    ds_clipped[var] = ds[var].where(ibound_coarse)

#%%
# Infiltration ponds - Conductance already added and combined with rivers in 1.4
inf_ponds_mean      = inf_ponds.mean("time", skipna = True)
inf_ponds_regridded = xr.Dataset()
for var in ("stage" , "bot", "density"):
    inf_ponds_regridded[var] = mean_regridder.regrid(inf_ponds_mean[var],
like=like).where(cond_ponds.notnull())
inf_ponds_regridded["cond"] = cond6_IP
#%%
# Add concentration[species] dimension to
# infiltration ponds
y = ibound_coarse["y"]
x = ibound_coarse["x"]
cond = inf_ponds_regridded["cond"]

conc = xr.full_like(cond, 0.0).where(cond.notnull()) # [Cl] = 0?
species_nd = xr.concat([
    conc,  #Cl
    xr.full_like(conc, 1.0).where(conc.notnull()),  # AM
    xr.full_like(conc, 0.0).where(conc.notnull())], # polders
    dim="species")
inf_ponds_regridded["conc"]  = species_nd

# Sea
conc_sea = ds_clipped["conc"]
species_sea = xr.concat([
    conc_sea.assign_coords(species=1),  # Cl
    xr.full_like(conc_sea, 0.0).where(conc_sea.notnull()).assign_coords(species=2),  # AM
    xr.full_like(conc_sea, 0.0).where(conc_sea.notnull()).assign_coords(species=3)], #
polders
    dim="species")
ds_sea_final = ds_clipped
ds_sea_final["conc"] = species_sea
ds_sea_final.to_netcdf("data/2-interim/sea.nc")
#%%
# Combine inf ponds and sea to a new dataset
ds_combined = xr.Dataset()
for var in ("stage", "cond", "conc", "density"):
    ds_combined[var] = inf_ponds_regridded[var].combine_first(ds_sea_final[var])

# combine first with rivers
ds_comb_3v2 = xr.Dataset()
for var in ("stage", "cond", "conc", "density"):
    ds_comb_3v2[var] = rivers[var].combine_first(ds_combined[var])

# combine with polder conductances from 1.4-find-conductances
ds_comb_4v2 = xr.Dataset()
for var in ("stage", "conc", "density"):
    ds_comb_4v2[var] = ds_comb_3v2[var]
```

```python
ds_comb_4v2["cond"] = ds_comb_3v2["cond"].combine_first(cond_polders)

# %%
ds_5 = link_z_layer(ds_comb_4v2, ibound_coarse)
# Erase the one cell with negative conductivity for now
ds_positive = ds_5["cond"] > 0
ds_negative = ds_5["cond"] < 0
ds_5["cond"] = ds_5["cond"].where(~ds_negative, other=0.0)
ds_5["cond"] = ds_5["cond"].where(ds_5["density"].notnull())

#ds_final = ds_positive * ds_5
# %% The following results in: conductance in <imod.wq.ghb.GeneralHeadBoundary object at 0x000001CE9FEB4E20> is not consistent with all variables in: conductance, density, concentration, head. nan values do not line up.
ghb_out = imod.wq.GeneralHeadBoundary(
    head          = ds_5["stage"],
    conductance   = ds_5["cond"],
    concentration = ds_5["conc"],
    density       = ds_5["density"],
    save_budget   = True
)
ghb_out.dataset.to_netcdf("data/3-input/ghb.nc")

# %%
inf_ponds_regridded.to_netcdf("data/2-interim/inf_ponds_re.nc")
# %%

```

src\1-prepare\1.6.1-create_drn_wel.py

```python
#%%
"""
Create Drainage and Well package
    Drainage:
    - Surface runoff defined by top of active cells in ibound, excluding the sea area.
    Drainage 2:
    - the phreatic drainage (horizontal drains) are specified in a second drainage object
    Well:
    - the idf's used for the 25m model are used
"""
#%%
import scipy.ndimage
import imod
import numpy as np
import os
import pandas as pd
import xarray as xr
#%%
os.chdir("c:/projects/msc-thesis")
# %%
# Functions
def moving_average(da, windowsize: int):
    weights = np.ones((windowsize, windowsize))
    weights = weights / weights.sum()  # sums to 1
    out = da.copy()
    scipy.ndimage.convolve(da.values, weights, out.values)
    return out


# %%
# Open data
ibound_coarse = xr.open_dataarray("data/2-interim/ibound_coarse.nc") # opening as data
array to store
ghb           = xr.open_dataset("data/3-input/ghb.nc")
river_dataset = xr.open_dataset("data/1-external/river.nc")
like          = xr.open_dataarray("data/2-interim/like.nc")
sea           = xr.open_dataset(r"data/2-interim/sea_clipped.nc")

# Second drainage and wells from output of 25m model
drn_2_cond  = imod.idf.open(r"data\1-external\data-25-run-1\drn\conductance_l*.idf")
drn_2_elev  = imod.idf.open(r"data\1-external\data-25-run-1\drn\elevation_l*.idf")
wells       = imod.ipf.read(r"data\1-external\data-25-run-
1\wel\wel_19791231235959_l*.ipf")

# Phreatic drainage from the correct package holymoly!
drn_3_cond  = imod.idf.open(r"data\1-external\data-25-run-1\phreatic-
extraction\conductance_l*.idf")
drn_3_elev  = imod.idf.open(r"data\1-external\data-25-run-1\phreatic-
extraction\elevation_l*.idf")


#%%
# Add rivers from river dataset and remove "boezems"
riv_mean = river_dataset.mean("time")

river_stage = riv_mean["stage"].max("z")
full_river  = imod.prepare.fill(river_stage)
```

```python
moving_average_river = moving_average(full_river, 11)

# We use the moving average to detect the high elevation canals. However, this
# results in a a false positive near the dunes. This looks like the only false
# positive, so we force all locations above the line y=462_500 to be kept.

keep = (full_river - moving_average_river) < 1.0
keep = keep | (keep["y"] > 462_500.0)
filtered_river_dataset = riv_mean.where(keep)

#%%
# Prepare regridders
mean_regridder = imod.prepare.Regridder(method="mean")
cond_regridder = imod.prepare.Regridder(method="conductance")

river_regridded = xr.Dataset()
for var in ("stage", "bot", "density"):
    river_regridded[var] = mean_regridder.regrid(filtered_river_dataset[var], like=like)
river_regridded["cond"] = cond_regridder.regrid(filtered_river_dataset["cond"], like=like)

drn_2_re = xr.Dataset()
drn_2_re["elevation"]   = mean_regridder.regrid(drn_2_elev, like=like)
drn_2_re["conductance"] = cond_regridder.regrid(drn_2_cond, like=like)

drn_3_re = xr.Dataset()
drn_3_re["elevation"]   = mean_regridder.regrid(drn_3_elev, like=like)
drn_3_re["conductance"] = cond_regridder.regrid(drn_3_cond, like=like)

#%%
# Set up DRN: Elevation of surface runoff
top      = ibound_coarse.coords["ztop"]
top_layers = top.where(ibound_coarse != 0).min("z")

# exclude sea
sea_2d  = sea["stage"].isel(z=0, drop = True)

top3d   = top.where(ibound_coarse != 0 )
top3d_2 = top3d.where(sea_2d.isnull())

surface_level = top3d.max("z")
is_top = top3d == surface_level

surface_level_2 = top3d_2.max("z")
is_top_2 = top3d_2 == surface_level_2

drain_elevation_2 = top3d_2.where(is_top_2)

# Set up DRN: add river stage and conductance
# Without sea
drn_el_combined_2 = drain_elevation_2.combine_first(river_regridded["stage"])
surface_level_2.to_netcdf("data/2-interim/surface_level_without_sea.nc")
#%%
# Conductance of drain

is_cond = is_top * ghb.conductance.max()
is_cond_2 = is_cond.where(is_top)

is_cond_lower = is_top * 250
is_cond_2_lower = is_cond_lower.where(is_top)

```

```python
# Conductance river drains
is_cond_combined = is_cond_2_lower.combine_first(river_regridded["cond"])
is_cond_no_sea   = is_cond_combined.where(sea_2d.isnull())

#%%
drn = imod.wq.Drainage(drn_el_combined_2, is_cond_no_sea, save_budget=True)
drn.dataset.to_netcdf("data/3-input/drn.nc")

drn_2 = imod.wq.Drainage(drn_2_re["elevation"], drn_2_re["conductance"], save_budget =
True)
drn_2.dataset.to_netcdf("data/3-input/drn_2.nc")

drn_3 = imod.wq.Drainage(drn_3_re["elevation"], drn_3_re["conductance"], save_budget =
True)
drn_3.dataset.to_netcdf("data/3-input/drn_3.nc")
#%%
# Wells
wel = imod.wq.Well(
    id_name=wells["id_name"],
    x=wells["x"],
    y=wells["y"],
    rate=wells["rate"],
    layer=wells["layer"],
    save_budget=True,
)
wel.dataset.to_netcdf("data/3-input/wel.nc")

# %%

```

src\1-prepare\1.6.2-create-oc-gcg-pcg.py

```python
#%%
import numpy as np
import imod
import os
import xarray as xr
#%%
os.chdir("c:/projects/msc-thesis")

#%%

oc= imod.wq.OutputControl(
    save_head_idf=True, save_concentration_idf=True, save_budget_idf=True
)

pcg = imod.wq.PreconditionedConjugateGradientSolver(
    max_iter=50,  # max number of outer iterations
    inner_iter=150,  # number of inner iterations
    hclose=0.0001,  # head change criterion for convergence
    rclose=5000.0,  # residual convergence criterion
    relax=0.98,  # relaxation parameter
    damp=1.0,  # damping factor, equal to 1 means no damping
)
gcg = imod.wq.GeneralizedConjugateGradientSolver(
    max_iter=50,  #
    inner_iter=100,  #
    cclose=1.0e-6,  # convergence criterion in terms of relative concentration
    preconditioner="mic",  #
    lump_dispersion=True,  #
)
2
# Save
oc.dataset.to_netcdf("data/3-input/oc.nc")
pcg.dataset.to_netcdf("data/3-input/pcg.nc")
gcg.dataset.to_netcdf("data/3-input/gcg.nc")

# %%

```

src\3-model\3-prepare-model.py

```python
""" Here the model is prepared. Time discretization can be adjusted to 1s to retrieve SS results
    NOTE the output folder needs to be changed accoding to the corresponding simulation
"""
#%%
import imod
import xarray as xr
import os
import pathlib
#%%
os.chdir("c:/projects/msc-thesis/")
#%%
# Read input for model

bas = imod.wq.BasicFlow.from_file("data/3-input/bas.nc")
lpf = imod.wq.LayerPropertyFlow.from_file("data/3-input/lpf.nc")
btn = imod.wq.BasicTransport.from_file("data/3-input/btn.nc")
dsp = imod.wq.Dispersion.from_file("data/3-input/dsp.nc")
adv = imod.wq.AdvectionTVD.from_file("data/3-input/adv.nc")
vdf = imod.wq.VariableDensityFlow.from_file("data/3-input/vdf.nc")
ghb = imod.wq.GeneralHeadBoundary.from_file("data/3-input/ghb.nc")
rch = imod.wq.RechargeHighestActive.from_file("data/3-input/rch.nc")
wel = imod.wq.Well.from_file("data/3-input/wel.nc")
drn = imod.wq.Drainage.from_file("data/3-input/drn.nc")
drn2= imod.wq.Drainage.from_file("data/3-input/drn_2.nc")
drn3= imod.wq.Drainage.from_file("data/3-input/drn_3.nc")
chd = imod.wq.ConstantHead.from_file("data/3-input/chd.nc")
oc  = imod.wq.OutputControl.from_file("data/3-input/oc.nc" )
pcg = imod.wq.PreconditionedConjugateGradientSolver.from_file("data/3-input/pcg.nc")
gcg = imod.wq.GeneralizedConjugateGradientSolver.from_file("data/3-input/gcg.nc")

#%%
# Initialize model

m_ss = imod.wq.SeawatModel("SS_1")

m_ss["bas"] = bas
m_ss["lpf"] = lpf
m_ss["btn"] = btn
m_ss["dsp"] = dsp
m_ss["adv"] = adv
m_ss["vdf"] = vdf
m_ss["ghb"] = ghb
m_ss["rch"] = rch
m_ss["oc" ] = oc
m_ss["pcg"] = pcg
m_ss["gcg"] = gcg
m_ss["wel"] = wel
m_ss["drn"]  = drn
m_ss["drn2"] = drn2
m_ss["drn3"] = drn3
#m_ss["chd"]  = chd
#m_ss["riv"] = riv
#%%
m_ss.create_time_discretization(additional_times=[
    "2014-12-31T23:59:59.000000000", "2015-01-01T00:00:00.000000000","2020-01-
01T00:00:00.000000000","2025-01-01T00:00:00.000000000",
    "2030-01-01T00:00:00.000000000", "2035-01-01T00:00:00.000000000","2040-01-01T00:00:00.000000000","2045-01-01T00:00:00.000000000",
    "2050-01-01T00:00:00.000000000", "2055-01-01T00:00:00.000000000","2060-01-01T00:00:00.000000000","2065-01-01T00:00:00.000000000","2070-01-01T00:00:00.000000000",
    "2075-01-01T00:00:00.000000000", "2080-01-01T00:00:00.000000000","2085-01-01T00:00:00.000000000","2090-01-01T00:00:00.000000000","2095-01-01T00:00:00.000000000",
    "2100-01-01T00:00:00.000000000", "2105-01-01T00:00:00.000000000","2110-01-01T00:00:00.000000000","2115-01-01T00:00:00.000000000",
    "2120-01-01T00:00:00.000000000", "2125-01-01T00:00:00.000000000","2130-01-01T00:00:00.000000000","2135-01-01T00:00:00.000000000",
    "2140-01-01T00:00:00.000000000", "2145-01-01T00:00:00.000000000","2150-01-01T00:00:00.000000000","2155-01-01T00:00:00.000000000",
    "2160-01-01T00:00:00.000000000", "2165-01-01T00:00:00.000000000","2170-01-01T00:00:00.000000000","2175-01-01T00:00:00.000000000",
    "2180-01-01T00:00:00.000000000", "2185-01-01T00:00:00.000000000","2190-01-01T00:00:00.000000000","2195-01-01T00:00:00.000000000",
    "2200-01-01T00:00:00.000000000", "2205-01-01T00:00:00.000000000","2210-01-01T00:00:00.000000000","2215-01-01T00:00:00.000000000"]
)
m_ss["time_discretization"].dataset["transient"] = False
modeldir_ss = pathlib.Path("data/3-input/SS_1")
m_ss.write(modeldir_ss, result_dir = "data/4-output/4-scenario-200y-fixedrand")

# additional_times = pd.date_range("2000-01-01", "3000-01-01", freq="100Y")
# %%

```

src\4-analyze\4.1-heads-errors-hist.py

```python
#%%
"""
Analyze the SS heads of metamodel, compare with 25m heads:
- Histogram
- TOP VIEW OF ERROR HEADS
"""
#%%
import numpy as np
import os
import imod
import xarray as xr
import matplotlib.pyplot as plt
import pandas as pd
import geopandas
import pathlib
from matplotlib import ticker
#%%
os.chdir("c:/projects/msc-thesis")
#%% Import data
like = xr.open_dataarray(r"c:/projects/msc-thesis/data/2-interim/like.nc")
gdf  = geopandas.read_file(r"c:\projects\msc-thesis\data\1-external\Polygon.shp")
heads_SS_OM_zarr = xr.open_zarr(r"c:\projects\msc-thesis\data\1-external\data-25-run-
1\head_ss_t0.zarr")
heads_SS_MM      = imod.idf.open(r"c:\projects\msc-thesis\data\4-output\4-scenario-100y-
fixedrand\head\head_201412312359_l*.idf")
starting_head_MM = imod.idf.open(r"c:\projects\msc-thesis\data\3-
input\SS_1\bas\starting_head_l*.idf")
#%% Process data
heads_SS_OM = heads_SS_OM_zarr["head"].drop("time").astype(np.float64)
heads_SS_MM_notime = heads_SS_MM.isel(time=0, drop=True)
raster              = imod.prepare.rasterize(gdf, like) # study area
# Import regridder & regrid OM data
mean_regridder = imod.prepare.Regridder(method="mean")
heads_SS_OM_re = mean_regridder.regrid(heads_SS_OM, like=like)
#%% Calculate errors
def er(expected, actual):
    re = actual - expected
    return re
error_SS_global     = er(heads_SS_OM_re, heads_SS_MM_notime)
error_SS_study_area = error_SS_global.where(raster==1)
# %% Statistics: global
mean         = error_SS_global.mean().compute().values
stdev        = error_SS_global.std().values
# Histogram
fig,ax = plt.subplots()
error_SS_global.plot.hist(ax=ax, xlim = [-4,4], bins=200)
ax.set_title("SS heads error global")
plt.ylabel("frequency [N]")
plt.xlabel("error [m]")
formatter = ticker.ScalarFormatter(useMathText=True)    # For scientific notation
formatter.set_scientific(True)
formatter.set_powerlimits((-1,1))
ax.yaxis.set_major_formatter(formatter)
# text box hacky
def as_si(x, ndp):
    s = '{x:0.{ndp:d}e}'.format(x=x, ndp=ndp)
    m, e = s.split('e')
    return r'{m:s}\times 10^{{{e:d}}}'.format(m=m, e=int(e))
plt.text(1, 35000,    r"$\mu = {0:s},$".format(as_si(mean , 2)))
plt.text(1, 31000, r"$\sigma = {0:s} $".format(as_si(stdev, 2)))
path = pathlib.Path(r"C:\projects\msc-thesis\reports\images\4-scenario-100y-
fixedrand/SS_head_global.png")
plt.savefig(path, dpi=300)
#%% Statistics study area
mean  = error_SS_study_area.mean().compute().values
stdev = error_SS_study_area.std().values
# Histogram study area
fig,ax = plt.subplots()
error_SS_study_area.plot.hist(ax=ax, xlim = [-4,4], bins=200)
ax.set_title("SS heads error study area")
plt.ylabel("frequency [N]")
plt.xlabel("error [m]")
formatter = ticker.ScalarFormatter(useMathText=True)    # For scientific notation
formatter.set_scientific(True)
formatter.set_powerlimits((-1,1))
ax.yaxis.set_major_formatter(formatter)
# text box hacky
def as_si(x, ndp):
    s = '{x:0.{ndp:d}e}'.format(x=x, ndp=ndp)
    m, e = s.split('e')
    return r'{m:s}\times 10^{{{e:d}}}'.format(m=m, e=int(e))
plt.text(1, 2000,    r"$\mu = {0:s},$".format(as_si(mean , 2)))
plt.text(1, 1800, r"$\sigma = {0:s} $".format(as_si(stdev, 2)))
path = pathlib.Path(r"C:\projects\msc-thesis\reports\images\4-scenario-100y-
fixedrand/SS_head_SA.png")
plt.savefig(path, dpi=300)

# %% Top view plot error heads
Katwijk_xy       = (88055, 468500)
Scheveningen_xy  = (79240, 458415)
Voorschoten_xy   = (90521, 459909)
# up to 32m depth
fig, (ax1, ax2) = plt.subplots(2,1, sharex=True, sharey=True, figsize = (10,12))
error_SS_global.mean("layer").plot.imshow(ax=ax1, vmin=-1.5, vmax=1.5, cmap ="RdBu")
ax1.set_title("mean hydraulic heads error over z ")
ax1.annotate('Katwijk', xy=Katwijk_xy, xytext=(80000, 467500),
            arrowprops=dict(arrowstyle="simple", facecolor='black'),
            )
ax1.annotate('Scheveningen', xy=Scheveningen_xy, xytext=(76000, 465000),
            arrowprops=dict(arrowstyle="simple", facecolor='black'),
            )
ax1.annotate('Voorschoten', xy=Voorschoten_xy, xytext=(91000, 450500),
            arrowprops=dict(arrowstyle="simple", facecolor='black'),
            )
error_SS_global.isel(layer=
[0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17]).mean("layer").plot.imshow(ax=ax2,
vmin=-1.5, vmax=1.5, cmap="RdBu")
ax2.set_title("mean hydraulic heads error to z = -32.5m")
ax2.annotate('Katwijk', xy=Katwijk_xy, xytext=(80000, 467500),
            arrowprops=dict(arrowstyle="simple", facecolor='black'),
            )
ax2.annotate('Scheveningen', xy=Scheveningen_xy, xytext=(76000, 465000),
            arrowprops=dict(arrowstyle="simple", facecolor='black'),
            )
ax2.annotate('Voorschoten', xy=Voorschoten_xy, xytext=(91000, 450500),
            arrowprops=dict(arrowstyle="simple", facecolor='black'),
            )
```

```python
path = pathlib.Path(r"C:\projects\msc-thesis\reports\images\4-scenario-100y-
fixedrand/SS_errors_topview.png")
plt.savefig(path, dpi=300)
#%% Error over depth
fig,ax = plt.subplots(figsize= (5,6))
error_SS_study_area.mean("x").mean("y").plot(y="z", ax=ax)
plt.xlabel("Hydraulic heads: mean error over study area [m]")
plt.ylabel("depth [m]")
plt.xlim(-1.5, 1.0)
plt.title("")
plt.grid()

# %%

```

src\4-analyze\4.2.1-budgets-topview.py

```python
""""
Analyze the budgets of the metamodel, compare with 25m heads:
- 25m budgets:
    - bdg drn
    - bdg riv
    - bdg wel

"""
#%%
import numpy as np
import os
import imod
import xarray as xr
import matplotlib.pyplot as plt
import scipy.ndimage.morphology
import pandas as pd
import pathlib
import geopandas
#%%
os.chdir("c:/projects/msc-thesis")
#%%
sum_regridder = imod.prepare.Regridder(method="sum")
mean_regridder = imod.prepare.Regridder(method="mean")
# Import data
like         = xr.open_dataarray(r"c:/projects/msc-thesis/data/2-interim/like.nc")
gdf          = geopandas.read_file(r"c:\projects\msc-thesis\data\1-external\Polygon.shp")
inf_ponds    = xr.open_dataset(r"c:\projects\msc-thesis\data\1-
external\infiltration_ponds.nc")
inf_ponds = inf_ponds.isel(time=0, drop=True)

# Output: budgets from metamodel
meta_drn_2053     = imod.idf.open(r"c:\projects\msc-thesis\data\4-
output\bdgdrn\bdgdrn_205312312359_l*.idf").isel(time=0, drop=True)
meta_drn_2024     = imod.idf.open(r"c:\projects\msc-thesis\data\4-
output\bdgdrn\bdgdrn_202412312359_l*.idf").isel(time=0, drop=True)

meta_ghb_2053     = imod.idf.open(r"c:\projects\msc-thesis\data\4-
output\bdgghb\bdgghb_205312312359_l*.idf").isel(time=0, drop=True)
meta_ghb_2024     = imod.idf.open(r"c:\projects\msc-thesis\data\4-
output\bdgghb\bdgghb_202412312359_l*.idf").isel(time=0, drop=True)

meta_rch_2053     = imod.idf.open(r"c:\projects\msc-thesis\data\4-
output\bdgrch\bdgrch_205312312359_l*.idf").isel(time=0, drop=True)
meta_rch_2024     = imod.idf.open(r"c:\projects\msc-thesis\data\4-
output\bdgrch\bdgrch_202412312359_l*.idf").isel(time=0, drop=True)

meta_wel_2053     = imod.idf.open(r"c:\projects\msc-thesis\data\4-
output\bdgwel\bdgwel_205312312359_l*.idf").isel(time=0, drop=True)
meta_wel_2024     = imod.idf.open(r"c:\projects\msc-thesis\data\4-
output\bdgwel\bdgwel_202412312359_l*.idf").isel(time=0, drop=True)

#meta_bnd_2053     = imod.idf.open(r"c:\projects\msc-thesis\data\4-
output\bdgbnd\bdgbnd_205312312359_l*.idf").isel(time=0, drop=True)
#meta_bnd_2024     = imod.idf.open(r"c:\projects\msc-thesis\data\4-
output\bdgbnd\bdgbnd_202412312359_l*.idf").isel(time=0, drop=True)

OM_drn = xr.open_zarr(r"c:\projects\msc-thesis\data\1-external\data-25-run-
1\bdgdrn_ss_t0.zarr")["bdgdrn"]
```

```python
OM_riv = xr.open_zarr(r"c:\projects\msc-thesis\data\1-external\data-25-run-
1\bdgriv_ss_t0.zarr")["bdgriv"]
OM_wel = xr.open_zarr(r"c:\projects\msc-thesis\data\1-external\data-25-run-
1\bdgwel_ss_t0.zarr")["bdgwel"]
# %% error
def er(expected, actual):
    re = actual - expected
    return re
# Relative error
def rel_er(expected, actual):
    re = np.abs((actual - expected) / expected)
    return re
# %% Process data
# OM - regrid
OMdrn_re_mean = mean_regridder.regrid(OM_drn, like)
OMdrn_re_sum  = sum_regridder.regrid(OM_drn, like)

OMriv_re = sum_regridder.regrid(OM_riv, like)
OMwel_re = sum_regridder.regrid(OM_wel, like)

inf_ponds_re = mean_regridder.regrid(inf_ponds["stage"], like)
raster        = imod.prepare.rasterize(gdf, like)

#%% Analyze - errors
error_drn_2053_sum  = er(OMdrn_re_sum, meta_drn_2053)
bdgdrn_error = error_drn_2053_sum.mean().compute()

# polder area and infiltration ponds
error_riv_2053 = er(OMriv_re, meta_ghb_2053*(OMriv_re.notnull()==1))
bdgriv_error = error_riv_2053.mean().compute()
# error well
error_wel = er(OMwel_re,meta_wel_2053 )
bdgwel_error = error_riv_2053.mean().compute()

#%% Plotting
(fig, axs) = plt.subplots(3,1, figsize=(10,15))
error_drn_2053_sum.mean("layer").plot.imshow(ax=axs[0])
axs[0].set_title("Error drn")
error_riv_2053.mean("layer").plot.imshow(ax=axs[1])
axs[1].set_title("Error riv")
error_wel.mean("layer").plot.imshow(ax=axs[2])
axs[2].set_title("Error wel")
path_3 = pathlib.Path(f"reports/images/scenario_FixedHead_Onder/budget_errors.png")
plt.savefig(path_3, dpi=300)
#%% Bar charts: budgets and errors
#%% Bar charts: budgets and errors NOT WORKING YET
df_names_OM = ["River budgets", "Drain budgets", "Well budgets"]
fig,axs = plt.subplots(2,1)

# Study area
df_OM_SA = [OMriv_re.where(raster==1).mean(), OMdrn_re_sum.where(raster==1).mean(),
OM_wel.where(raster==1).mean()]
axs[0] = plt.bar(df_names_OM, df_OM_SA)



# %%

```

src\4-analyze\4.2.2-budgets-waterbalance.py

```
 1  """" Water balance:
 2      IN:
 3      - rch    (MM, OM)
 4      - ghb    (MM,    )   (inf ponds)
 5      - riv    (   , OM)   (inf ponds, polders)
 6  
 7      OUT:
 8      - drn    (MM, OM)   (surface runoff, phreatic extraction)
 9      - well   (MM, OM)   (drainage for drinking water)
10  output:
11  - OM plot defined per calibration step (cond), 1.4.2
12  
13  NOTE The SS budgets from MM calibration are defined in highlighted lines 82 and 84.
14       Change them according to output date of run
15  """
16  #%%
17  import numpy as np
18  import os
19  import imod
20  import xarray as xr
21  import matplotlib.pyplot as plt
22  import pandas as pd
23  import pathlib
24  import geopandas
25  from matplotlib import ticker
26  #%%
27  os.chdir("c:/projects/msc-thesis")
28  #%%
29  sum_regridder = imod.prepare.Regridder(method="sum")
30  mean_regridder = imod.prepare.Regridder(method="mean")
31  # Import data
32  like         = xr.open_dataarray(r"c:/projects/msc-thesis/data/2-interim/like.nc")
33  like_fine    = xr.open_dataarray(r"data/1-external/template_2d.nc")
34  gdf          = geopandas.read_file(r"c:\projects\msc-thesis\data\1-external\Polygon.shp")
35  inf_ponds    = xr.open_dataset(r"c:\projects\msc-thesis\data\1-
    external\infiltration_ponds.nc")
36  inf_ponds = inf_ponds.isel(time=0, drop=True)
37  # Output: budgets from metamodel
38  meta_drn_SS       = imod.idf.open(r"c:\projects\msc-thesis\data\4-output\3-
    scenario_FixedHead_rand\bdgdrn\bdgdrn_201412312359_l*.idf").isel(time=0, drop=True)
39  
40  meta_ghb_SS       = imod.idf.open(r"c:\projects\msc-thesis\data\4-output\3-
    scenario_FixedHead_rand\bdgghb\bdgghb_201412312359_l*.idf").isel(time=0, drop=True)
41  
42  meta_rch_2053     = imod.idf.open(r"c:\projects\msc-thesis\data\4-
    output\bdgrch\bdgrch_205312312359_l*.idf").isel(time=0, drop=True)
43  # meta_drn_SS Unneccesary since rch will be the same
44  
45  meta_wel_2053     = imod.idf.open(r"c:\projects\msc-thesis\data\4-
    output\bdgwel\bdgwel_205312312359_l*.idf").isel(time=0, drop=True)
46  # meta_wel_SS Unneccessary since wel will be the same
47  
48  OM_drn = xr.open_zarr(r"c:\projects\msc-thesis\data\1-external\data-25-run-
    1\bdgdrn_ss_t0.zarr")["bdgdrn"]
49  OM_riv = xr.open_zarr(r"c:\projects\msc-thesis\data\1-external\data-25-run-
    1\bdgriv_ss_t0.zarr")["bdgriv"]
50  OM_wel = xr.open_zarr(r"c:\projects\msc-thesis\data\1-external\data-25-run-
    1\bdgwel_ss_t0.zarr")["bdgwel"]
```

```python
# %% error
def er(expected, actual):
    re = actual - expected
    return re
# Relative error
def rel_er(expected, actual):
    re = np.abs((actual - expected) / expected)
    return re
# %% Process data
# OM - regrid
like_fine["dx"] = xr.DataArray(25.0)
like_fine["dy"] = xr.DataArray(-25.0)
OMdrn_re_mean = mean_regridder.regrid(OM_drn, like)
OMdrn_re_sum  = sum_regridder.regrid(OM_drn, like)

OMriv_re = sum_regridder.regrid(OM_riv, like)
OMwel_re = sum_regridder.regrid(OM_wel, like)

inf_ponds_re  = mean_regridder.regrid(inf_ponds["stage"], like)
raster_MM     = imod.prepare.rasterize(gdf, like)           # study area, MM
raster_OM     = imod.prepare.rasterize(gdf, like_fine)      # study area, OM

# Study area water balance, use sum:
# IN
rch_MM_SA = meta_rch_2053.where(raster_MM==1).sum("layer") # Assumed to be the same for
both? check!
ghb_MM_SA = meta_ghb_SS.where(raster_MM==1).sum("layer") # Infiltration ponds, MM only
riv_OM_SA = OMriv_re.where(raster_MM==1).sum("layer")      # Infiltration ponds, OM only

# OUT
drn_MM_SA = meta_drn_SS.where(raster_MM==1).sum("layer") # Surface runoff and phreatic
drn_OM_SA = OMdrn_re_sum.where(raster_MM==1).sum("layer")  # Surface runoff and phreatic

wel_MM_SA = meta_wel_2053.where(raster_MM==1).sum("layer")
wel_OM_SA = OMwel_re.where(raster_MM==1).sum("layer")
#%% Bar charts OM
fig,ax = plt.subplots()
WB_OM = ["riv", "drn", "well"]
plt.bar(WB_OM, [riv_OM_SA.sum().compute(),   # Infiltration ponds
                drn_OM_SA.sum().compute(),   # surface runoff (+pipe drainage?)
                wel_OM_SA.sum().compute()])  # Wells
plt.title("SS Water Balance for original model - Study Area")
plt.ylabel("$m^{3}$/d")
plt.grid()
#path_3 = pathlib.Path(f"reports/images/3-
scenario_FixedHead_rand/Waterbalance_OM_sum.png")
#plt.savefig(path_3, dpi=300)
#%% Bar Charts MM
fig,axs = plt.subplots()
WB_OM = ["ghb", "drn", "well"]
ax = plt.bar(WB_OM, [ghb_MM_SA.sum().compute(),   # infiltration ponds
                     #rch_MM_SA.sum().compute(),   # precipitation
                     drn_MM_SA.sum().compute(),   # surface runoff (+ pipe drainage?)
                     wel_MM_SA.sum().compute()])  # Wells
plt.title("SS Water Balance for metamodel - Study Area - cond6")
plt.ylabel("$m^{3}$/d")

plt.grid()
#path_3 = pathlib.Path(f"reports/images/3-
scenario_FixedHead_rand/Waterbalance_MM_cond6_sum.png") # According to calibration 1.4.2
#plt.savefig(path_3, dpi=300)
```

```
# %%

```

src\4-analyze\4.2.3-budgets-FLF.py

```python
"""
In this script, the Flux Lower Boundary (FLF) of the OM and MM inside the study area will be investigated.
- FLF could be investigated per depth!
- Histograms may not be needed.
"""
#%%
import numpy as np
import os
import imod
import xarray as xr
import matplotlib.pyplot as plt
from matplotlib import ticker
import pandas as pd
import pathlib
import geopandas
import matplotlib.patches as mpatches
#%%
os.chdir("c:/projects/msc-thesis")
#%% Import Data
flf_MM_SS = imod.idf.open(r"c:\projects\msc-thesis\data\4-output\4-scenario-100y-fixedrand\bdgflf\bdgflf_201412312359_l*.idf")
flf_OM_SS = xr.open_zarr(r"c:\projects\msc-thesis\data\1-external\SS_run2_25m\dunea_transient_budget\bdgflf.zarr")
# Study area
gdf    = geopandas.read_file(r"c:\projects\msc-thesis\data\1-external\Polygon.shp")
like   = xr.open_dataarray(r"c:/projects/msc-thesis/data/2-interim/like.nc")
raster = imod.prepare.rasterize(gdf, like)
#%% Process Data
flf_MM_SS_notime = flf_MM_SS.drop("time")
flf_OM_SS_2 = flf_OM_SS.drop("time").astype(np.float64).to_array()
# Import regridder & regrid original data
sum_regridder  = imod.prepare.Regridder(method="sum")
flf_OM_re      = sum_regridder.regrid(flf_OM_SS_2, like=like)
# Study area:
flf_MM_SA = flf_MM_SS_notime.where(raster==1)
flf_OM_SA = flf_OM_re.where(raster==1)
#%% Errors
def er(expected, actual):
    re = actual - expected
    return re
error_SS_global     = er(flf_OM_re, flf_MM_SS_notime)
error_SS_study_area = error_SS_global.where(raster==1)      # in m3/d
flow_vertical_error = error_SS_study_area/(250*250)         # in m/d
#%%
"""Histograms"""
# Statistics: global
mean        = error_SS_global.mean().compute().values
stdev       = error_SS_global.std().values
# Histogram
fig,ax = plt.subplots()
error_SS_global.plot.hist(ax=ax, xlim = [-100,100], bins=2000)
ax.set_title("SS flf error global")
plt.ylabel("frequency [N]")
plt.xlabel("$m^{3}$/d")
formatter = ticker.ScalarFormatter(useMathText=True)    # For scientific notation
formatter.set_scientific(True)
formatter.set_powerlimits((-1,1))
ax.yaxis.set_major_formatter(formatter)
# text box hacky
def as_si(x, ndp):
    s = '{x:0.{ndp:d}e}'.format(x=x, ndp=ndp)
    m, e = s.split('e')
    return r'{m:s}\times 10^{{{e:d}}}'.format(m=m, e=int(e))
plt.text(30, 2.5e05,    r"$\mu = {0:s},$".format(as_si(mean , 2)))
plt.text(30, 2.3e05, r"$\sigma = {0:s} $".format(as_si(stdev, 2)))
#path = pathlib.Path(f"reports/images/3-scenario_FixedHead_rand/SS_flf_global.png")
#plt.savefig(path, dpi=300)
#%% Statistics: study area
mean        = error_SS_study_area.mean().compute().values
stdev       = error_SS_study_area.std().values
# Histogram
fig,ax = plt.subplots()
error_SS_study_area.plot.hist(ax=ax, xlim = [-100,100], bins=2000)
ax.set_title("SS flf error study area")
plt.ylabel("frequency [N]")
plt.xlabel("$m^{3}$/d")
formatter = ticker.ScalarFormatter(useMathText=True)    # For scientific notation
formatter.set_scientific(True)
formatter.set_powerlimits((-1,1))
ax.yaxis.set_major_formatter(formatter)
# text box hacky
def as_si(x, ndp):
    s = '{x:0.{ndp:d}e}'.format(x=x, ndp=ndp)
    m, e = s.split('e')
    return r'{m:s}\times 10^{{{e:d}}}'.format(m=m, e=int(e))
plt.text(30, 1.1e04,    r"$\mu = {0:s},$".format(as_si(mean , 2)))
plt.text(30, 1.0e04, r"$\sigma = {0:s} $".format(as_si(stdev, 2)))
#path = pathlib.Path(f"reports/images/3-scenario_FixedHead_rand/SS_flf_SA.png")
#plt.savefig(path, dpi=300)


#%% PLOTTING ERROR VS DEPTH
fig,ax = plt.subplots(figsize= (5,6))
error_SS_study_area.mean("x").mean("y").plot(y="z",ax=ax)   # in m3/d or in mm/d
#flow_vertical_error = error_SS_study_area/(250*250)         # in m/d
#flow_vertical_error.mean("x").mean("y").plot(y="z",ax=ax)
plt.xlabel("FLF: mean error over study area [$m^{3}$/d]")
#plt.xlabel("FLF: mean error over study area [mm/d]")

plt.ylabel("depth [mNAP]")
plt.xlim(-20, 60)
plt.title("")
plt.grid()
path = pathlib.Path(f"reports/images/4-scenario-100y-fixedrand/SS_flf_SA.png")
plt.savefig(path, dpi=300)
#%% OM SPECIFICLY
flf_OM_re_with_z = flf_OM_re
flf_OM_re_with_z["z"] = flf_MM_SA["z"]

fig,ax = plt.subplots(figsize= (5,6))
flf_OM_re_with_z.where(raster==1).mean("x").mean("y").plot(y="z",ax=ax)
plt.xlabel("FLF original model: mean over study area [$m^{3}$/d]")
plt.xscale("symlog")
plt.ylabel("depth [mNAP]")
plt.grid()
plt.title("")
```

```python
path = pathlib.Path(f"reports/images/4-scenario-100y-fixedrand/FLF_OM_SA.png")
plt.savefig(path, dpi=300)
#%% MM Specificly
fig,ax = plt.subplots(figsize= (5,6))
flf_MM_SA.mean("x").mean("y").plot(y="z",ax=ax)
plt.xlabel("FLF metamodel: mean over study area [$m^{3}$/d]")
plt.xscale("symlog")
plt.ylabel("depth [mNAP]")
plt.grid()
plt.title("")
path = pathlib.Path(f"reports/images/4-scenario-100y-fixedrand/FLF_MM_SA.png")
plt.savefig(path, dpi=300)
# %%

```

src\4-analyze\4.3-fresh-saline.py

```python
"""
IN this script the depth of the fresh-saline interface will be plotted,
    and its error when comparing to the original model's output is calculated and plotted
Cross sections of the groundwater salinity are also plotted and compared (meta and OM)

"""

#%%
import numpy as np
import os
import imod
import xarray as xr
import matplotlib.pyplot as plt
import scipy.ndimage.morphology
import pandas as pd
import pathlib
import geopandas
#%%
os.chdir("c:/projects/msc-thesis")
#%%
mean_regridder = imod.prepare.Regridder(method="mean")
# Import data
like   = xr.open_dataarray(r"c:/projects/msc-thesis/data/2-interim/like.nc")
ibound = xr.open_dataarray(r"c:/projects/msc-thesis/data/2-interim/ibound_coarse.nc")
gdf    = geopandas.read_file(r"c:\projects\msc-thesis\data\1-external\Polygon.shp")
#surface_level = xr.open_dataarray("data/2-interim/surface_level_without_sea.nc")
# output data
conc_OM     = xr.open_zarr(r"c:\projects\msc-thesis\data\1-external\data-25-run-1\conc-selection.zarr")["conc"].astype(np.float64)
conc_meta   = imod.idf.open(r"c:\projects\msc-thesis\data\4-output\2-scenario_dichte_rand\conc\conc_c*.IDF").isel(species=0,drop=True)
conc_meta = conc_meta.where(conc_meta != 1e30)
conc_meta = conc_meta.where(~(conc_meta < 0.0), other=0.0)
#%% Process data
conc_OM_39y    = mean_regridder.regrid(conc_OM.isel(time=-1), like)    # note that the final date is 39y and 9 months, not 40y
conc_meta_39y = conc_meta.isel(time=-1, drop=True)
conc_OM_39y["z"] = conc_meta_39y["z"] # Add layer coordinates that got lost in regridding
raster         = imod.prepare.rasterize(gdf, like)
#%% Functions
def rel_er(expected, actual):
    act_abs = np.abs(actual)
    exp_abs = np.abs(expected)
    re = ((act_abs - exp_abs)/exp_abs)
    return re
def er(expected, actual):
    er = actual - expected
    return er
#%% Depth of fresh-brackish interface
# Bounds for groundwater types
fresh_upper = 0.300    # g/l
brack_upper = 10.0000   # g/l
#%%
# Depth fresh-saline interface OM
depth_fresh_OM2  = conc_OM["z"].where(conc_OM_39y < fresh_upper).min("layer")
depth_fresh_OM_H = depth_fresh_OM2.combine_first(surface_level)

```

```python
# Depth fresh-saline interface meta
depth_fresh_meta2  = conc_OM["z"].where(conc_meta_39y < fresh_upper).min("layer") # is
this correct?
depth_fresh_meta_H = depth_fresh_meta2.combine_first(surface_level)
#%%
# Errors
error_depth = er(depth_fresh_OM_H, depth_fresh_meta_H)
error_depth_study_area = error_depth.where(raster==1)

error_Cl = er(conc_OM_39y, conc_meta_39y)
error_Cl_SA = error_Cl.where(raster==1)

#%%
# Plotting & saving depth of interface
levels_depth = -1 * np.arange(0,130)
#text = np.array((80000, 462000))
Katwijk_xy       = (88055, 468500)
Scheveningen_xy  = (79240, 458415)
Voorschoten_xy   = (90521, 459909)
fig,(ax1, ax2, ax3) = plt.subplots(3,1, sharex=True, sharey=True, figsize=(6,12))
depth_fresh_OM_H.plot.imshow(ax=ax1, cmap="turbo", levels=levels_depth)
ax1.set_title("Original model")
ax1.annotate('Katwijk', xy=Katwijk_xy, xytext=(80000, 467500),
            arrowprops=dict(arrowstyle="simple", facecolor='black'),
            )
ax1.annotate('Scheveningen', xy=Scheveningen_xy, xytext=(76000, 465000),
            arrowprops=dict(arrowstyle="simple", facecolor='black'),
            )
ax1.annotate('Voorschoten', xy=Voorschoten_xy, xytext=(91000, 450500),
            arrowprops=dict(arrowstyle="simple", facecolor='black'),
            )
depth_fresh_meta_H.plot.imshow(ax=ax2,cmap="turbo", levels=levels_depth)

ax2.set_title("Metamodel")
ax2.annotate('Katwijk', xy=Katwijk_xy, xytext=(80000, 467500),
            arrowprops=dict(arrowstyle="simple", facecolor='black'),
            )
ax2.annotate('Scheveningen', xy=Scheveningen_xy, xytext=(76000, 465000),
            arrowprops=dict(arrowstyle="simple", facecolor='black'),
            )
ax2.annotate('Voorschoten', xy=Voorschoten_xy, xytext=(91000, 450500),
            arrowprops=dict(arrowstyle="simple", facecolor='black'),
            )
#%%
fig,ax = plt.subplots()
error_depth.plot.imshow(ax=ax, vmin=-30, vmax=30, cmap="RdBu")
ax.set_title("Error of the metamodel")
ax.annotate('Katwijk', xy=Katwijk_xy, xytext=(80000, 467500),
            arrowprops=dict(arrowstyle="simple", facecolor='black'),
            )
ax.annotate('Scheveningen', xy=Scheveningen_xy, xytext=(76000, 465000),
            arrowprops=dict(arrowstyle="simple", facecolor='black'),
            )
ax.annotate('Voorschoten', xy=Voorschoten_xy, xytext=(91000, 450500),
            arrowprops=dict(arrowstyle="simple", facecolor='black'),
            )
path = pathlib.Path(f"reports/images/3-
scenario_FixedHead_rand/depth_freshwater_error.png")
path.parent.mkdir(exist_ok=True, parents=True)
fig.savefig(path, dpi=300)
```

```python
#%%
path = pathlib.Path(f"reports/images/3-scenario_FixedHead_rand/depth_freshwater.png")
path.parent.mkdir(exist_ok=True, parents=True)
fig.savefig(path, dpi=300)

#%% Plotting - cross sections salinity
levels_conc  = [0.0, 2.0, 4.0, 6.0, 8.0, 10.0, 12.0, 14.0, 16.0, 18.0]
levels_conc_err = np.arange(0,20)/2-5

# Cross sections
starts = [
    (75000.0, 459948.0), # (x,y)
    (77423.0, 462817.0),
    (79234.0, 464828.4),
    (81880.4, 467911.6),
    (83718.9, 469707.8),
]

ends = [
    (87591.0, 449868.0), # (x,y)
    (92828.2, 450002.0),
    (93914.6, 452120.1),
    (95223.9, 455793.2),
    (96393.9, 460431.4),
]
# %% Creating Cross sections next to eachother
plt.figure(figsize=(21,34))
plt.subplots_adjust(hspace=0.5)
for i, (start, end) in enumerate(zip(starts, ends)):
    ax_OM = plt.subplot(10,2, (i+1)*2)  # Set the position of the subplot
    CS_OM = imod.select.cross_section_line(conc_OM_39y, start=start, end=end)
    CS_OM.plot(ax=ax_OM,y="z", cmap = "turbo", levels = levels_conc)
    plt.title(f"CS{i+1} original model")
    #plt.ylabel("[Cl] [g/l]")
 #   plt.colorbar(ax=ax, label="conc")

# one column set using indexing, rest manual:
ax1 = plt.subplot(10,2,1, sharex=ax_OM) # position 1 (top left)
CS2 = imod.select.cross_section_line(conc_meta_39y, start=starts[0], end=ends[0])
CS2.plot(ax=ax1,y="z", cmap = "turbo", levels = levels_conc)
plt.title(f"CS1 metamodel")
#plt.ylabel("[Cl] [g/l]")

ax3 = plt.subplot(10,2,3, sharex=ax_OM) # position 3 (left)
CS2 = imod.select.cross_section_line(conc_meta_39y, start=starts[1], end=ends[1])
CS2.plot(ax=ax3,y="z", cmap = "turbo", levels = levels_conc)
plt.title(f"CS2 metamodel")
#plt.ylabel("[Cl] [g/l]")

ax5 = plt.subplot(10,2,5, sharex=ax_OM) # position 5 (left)
CS2 = imod.select.cross_section_line(conc_meta_39y, start=starts[2], end=ends[2])
CS2.plot(ax=ax5,y="z", cmap = "turbo", levels = levels_conc)
plt.title(f"CS3 metamodel")
#plt.ylabel("[Cl] [g/l]")

ax7 = plt.subplot(10,2,7, sharex=ax_OM) # position 7 (left)
CS2 = imod.select.cross_section_line(conc_meta_39y, start=starts[3], end=ends[3])
CS2.plot(ax=ax7,y="z", cmap = "turbo", levels = levels_conc)
plt.title(f"CS4 metamodel")
#plt.ylabel("[Cl] [g/l]")
```

```

ax9 = plt.subplot(10,2,9, sharex=ax_OM) # position 9 (left)
CS2 = imod.select.cross_section_line(conc_meta_39y, start=starts[4], end=ends[4])
CS2.plot(ax=ax9,y="z", cmap = "turbo", levels = levels_conc)
plt.title(f"CS5 metamodel")
#plt.ylabel("[Cl] [g/l]")
#%%
path_4 = pathlib.Path(f"reports/images/3-
scenario_FixedHead_rand/CS_salinity_combined.png")
plt.savefig(path_4, dpi=200)
#%% plotting CS2 and CS3

starts = [
#    (75000.0, 459948.0), # (x,y)
#    (77423.0, 462817.0),
    (79234.0, 464828.4),
    (81880.4, 467911.6),
#    (83718.9, 469707.8),
]

ends = [
#    (87591.0, 449868.0), # (x,y)
#    (92828.2, 450002.0),
    (93914.6, 452120.1),
    (95223.9, 455793.2),
#    (96393.9, 460431.4),
]

plt.figure(figsize=(12,16))
plt.subplots_adjust(hspace=0.5)
# Original model
for i, (start, end) in enumerate(zip(starts, ends)):
    ax_OM = plt.subplot(4,2, (i+1)*2)  # Set the position of the subplot
    CS_OM = imod.select.cross_section_line(conc_OM_39y, start=start, end=end)
    CS_OM.plot(ax=ax_OM,y="z", cmap = "turbo", levels = levels_conc)
    plt.title(f"CS{i+3} original model")


# Metamodel
ax3 = plt.subplot(4,2,1, sharex=ax_OM) # position 3 (left)
CS2 = imod.select.cross_section_line(conc_meta_39y, start=starts[0], end=ends[0])
CS2.plot(ax=ax3,y="z", cmap = "turbo", levels = levels_conc)
plt.title(f"CS3 metamodel")

ax5 = plt.subplot(4,2,3, sharex=ax_OM) # position 5 (left)
CS2 = imod.select.cross_section_line(conc_meta_39y, start=starts[1], end=ends[1])
CS2.plot(ax=ax5,y="z", cmap = "turbo", levels = levels_conc)
plt.title(f"CS4 metamodel")
path_4 = pathlib.Path(f"reports/images/Discussion-intrusion.png")
plt.savefig(path_4, dpi=200)
# %%

```

src\4-analyze\4.4-Species.py

```python
"""
- In this script the cross secitonal plots of (Stuyfzand, 1993) will be plotted.
- This is only for the metamodel, as the OM doesn't have species dimension
- This version shows groundwater salinity, its goal is to highlight the domains:
    - Saline groundwater
    - Fresh groundwater
    - Brackish groundwater
    - infiltration ponds species (AM)

NOTE:   To get the CS par to coastline, do not overwrite the cross sections for AM
        with perpendicular. Skip the cell with perpendicular cross section selection
"""

#%%
import imod
import numpy as np
import xarray as xr
import os
import geopandas
import matplotlib.patches as mpatches
import matplotlib.pyplot as plt
import matplotlib
import pathlib
# %%
os.chdir("c:/projects/msc-thesis")
# %% Data
like   = xr.open_dataarray(r"c:/projects/msc-thesis/data/2-interim/like.nc")
ibound = xr.open_dataarray(r"c:/projects/msc-thesis/data/2-interim/ibound_coarse.nc")
gdf    = geopandas.read_file(r"c:\projects\msc-thesis\data\1-external\Polygon.shp") # study
surface_level = xr.open_dataarray("data/2-interim/surface_level_without_sea.nc")
# output data
c1_meta = imod.idf.open(r"c:\projects\msc-thesis\data\4-output\4-scenario-100y-
fixedrand\conc\conc_c1*.IDF").isel(time=9, drop=True)
c2_meta = imod.idf.open(r"c:\projects\msc-thesis\data\4-output\4-scenario-100y-
fixedrand\conc\conc_c2*.IDF").isel(time=9, drop=True)
c3_meta = imod.idf.open(r"c:\projects\msc-thesis\data\4-output\4-scenario-100y-
fixedrand\conc\conc_c3*.IDF").isel(time=9, drop=True)
# Process data
Cl     = c1_meta.where(c1_meta != 1e30).where(~(c1_meta < 0.0), other=0.0).isel(species=0, d
AM     = c2_meta.where(c2_meta != 1e30).where(~(c2_meta < 0.0), other=0.0).isel(species=0, d
polder = c3_meta.where(c3_meta != 1e30).where(~(c3_meta < 0.0), other=0.0).isel(species=0, d
# for regridding to study area:
mean_regridder = imod.prepare.Regridder(method="mean")

# fresh saline levels
fresh_upper =  0.300   # g/l
brack_upper = 10.0000  # g/l
# %% CROSS SECTIONS
AM_notnull = AM.where(AM>0.01)
# species CS parallel to coastline
start_loosduinen = (75471,453198)
end_katwijk      = (87789.1,468649.0)
# Fresh
CS_par_f = imod.select.cross_section_line(c1_meta.where(c1_meta<fresh_upper), start=start_lo
end=end_katwijk).notnull()
CS_par_f = CS_par_f.where(CS_par_f!=0) # setting NaN to avoid overwriting plots
# Brackish
```

```
CS_par_b = imod.select.cross_section_line(c1_meta.where(c1_meta>fresh_upper).where(c1_meta<b
start=start_loosduinen, end=end_katwijk).notnull()
CS_par_b = CS_par_b.where(CS_par_b!=0) # setting NaN to avoid overwriting plots
# Saline
CS_par_s = imod.select.cross_section_line(c1_meta.where(c1_meta>brack_upper).where(CS_par_f.
start=start_loosduinen, end=end_katwijk).notnull()
CS_par_s = CS_par_s.where(CS_par_s!=0) # setting NaN to avoid overwriting plots
# AM
CS_AM = imod.select.cross_section_line(AM_notnull.where(AM_notnull!=0),  start=start_loosdui
end=end_katwijk).notnull()
#%%
# species CS perpendicular to coastline (Figure 4.10 Stuyfzand SWE 1993)
start_NS = (82478, 463358)
end_SD   = (90407, 456746)
# Fresh
CS_perp_f = imod.select.cross_section_line(c1_meta.where(c1_meta<fresh_upper), start=start_N
end=end_SD).notnull()
CS_perp_f = CS_perp_f.where(CS_perp_f!=0)
# Brackish
CS_perp_b = imod.select.cross_section_line(c1_meta.where(c1_meta>fresh_upper).where(c1_meta<
start=start_NS, end=end_SD).notnull()
CS_perp_b = CS_perp_b.where(CS_perp_b!=0) # setting NaN to avoid overwriting plots
# Saline
CS_perp_s = imod.select.cross_section_line(c1_meta.where(c1_meta>brack_upper).where(CS_perp_
start=start_NS, end=end_SD).notnull()
CS_perp_s = CS_perp_s.where(CS_perp_s!=0) # setting NaN to avoid overwriting plots
# AM
CS_AM = imod.select.cross_section_line(AM_notnull.where(AM_notnull!=0),  start=start_NS,
end=end_SD).notnull()
#%% Plotting
# Colors
yellow      = (1.0    , 1.0    , 143/255)             # should be normalized values, divide
brightblue  = (147/255, 187/255, 226/255)
blue        = (83/255 , 161/255, 224/255)
darkblue    = (35/255 , 130/255, 183/255)
colors = [yellow, darkblue, brightblue, blue  ]    # needed to make a colormap
levels = [1,   2,        3,          4,      5]    # needed to make a colormap
#%% CS PARRALEL
fig, ax = plt.subplots(figsize=(10,8))
# saline
CS_par_s_2 = 2*CS_par_s.where(CS_par_s["z"]<1.0).where(CS_par_s["s"]>1680) #to avoid NaN as
# 2 or 0
plot_3 = CS_par_s_2.plot(ax=ax, y="z", colors=colors, levels=levels, add_colorbar=False)
# fresh
CS_par_f_3 = 3*CS_par_f
plot_1 = CS_par_f_3.plot(ax=ax, y="z", colors=colors, levels=levels, add_colorbar=False)
# 3 or 0
# Brackish
CS_par_b_4 = 4*CS_par_b
plot_2 = CS_par_b_4.plot(ax=ax, y="z", colors=colors, levels=levels, add_colorbar=False)
# 4 or 0
# AM
CS_AM_mod = CS_AM.where(CS_AM>0.01).where(CS_AM!=0).notnull()
# 1 or 0
plot_4 = CS_AM_mod.where(CS_AM_mod!=0).plot(ax=ax, y="z", colors=colors, levels=levels,
add_colorbar=False)

# since quadmesh is not supported, proxy is required
(https://matplotlib.org/2.0.2/users/legend_guide.html#proxy-legend-handles)
Yello_patch     = mpatches.Patch(color=yellow ,      label='Artificial Infiltration')
BBlu_patch     = mpatches.Patch(color=brightblue  , label='Fresh groundwater')
Blu_patch     = mpatches.Patch(color=blue  ,       label='Brackish groundwater')
DBlu_patch  = mpatches.Patch(color=darkblue,     label='Saline groundwater')
```

```python

ax.legend(handles=[Yello_patch, BBlu_patch, Blu_patch, DBlu_patch], loc="lower right")
plt.xlim(2500, 19000)
plt.ylim(-160,11)
plt.text(2800,12,"Loosduinen")
plt.text(18000,12, "Katwijk")
plt.title("Species after 100y simulation, cross section along coastline")
path2 = pathlib.Path(f"reports/images/CS_long_species_40y.png")
plt.savefig(path2, dpi=200)
#%% CS PERP
fig, ax = plt.subplots(figsize=(10,8))
# saline
CS_perp_s_2 = 2*CS_perp_s.where(CS_perp_s["z"]<1.0)#.where(CS_perp_s["s"]>1680)  #to avoid N
# 2 or 0
plot_3 = CS_perp_s_2.plot(ax=ax, y="z", colors=colors, levels=levels, add_colorbar=False)
# fresh
CS_perp_f_3 = 3*CS_perp_f
plot_1 = CS_perp_f_3.plot(ax=ax, y="z", colors=colors, levels=levels, add_colorbar=False)
# 3 or 0
# Brackish
CS_perp_b_4 = 4*CS_perp_b
plot_2 = CS_perp_b_4.plot(ax=ax, y="z", colors=colors, levels=levels, add_colorbar=False)
# 4 or 0
# AM
CS_AM_mod = CS_AM.where(CS_AM>0.01).where(CS_AM!=0).notnull()
# 1 or 0
plot_4 = CS_AM_mod.where(CS_AM_mod!=0).plot(ax=ax, y="z", colors=colors, levels=levels,
add_colorbar=False)

# since quadmesh is not supported, proxy is required
(https://matplotlib.org/2.0.2/users/legend_guide.html#proxy-legend-handles)
Yello_patch     = mpatches.Patch(color=yellow ,      label='Artificial Infiltration')
BBlu_patch      = mpatches.Patch(color=brightblue  , label='Fresh groundwater')
Blu_patch       = mpatches.Patch(color=blue  ,       label='Brackish groundwater')
DBlu_patch      = mpatches.Patch(color=darkblue,     label='Saline groundwater')

ax.legend(handles=[Yello_patch, BBlu_patch, Blu_patch, DBlu_patch], loc="lower right")
#plt.xlim(2500, 19000)
plt.ylim(-160,11)
plt.text(100,12,"North Sea")
plt.text(7500,12, "Starrevaart & Damhouder polder")
path2 = pathlib.Path(f"reports/images/CS_perp_species_40y.png")
plt.savefig(path2, dpi=200)

# %% DUMP: TRransparency plots
# Plotting transparency plots (only AM and polders)
# To show both polder and AM, use combine_first, first the zero values need to be removed
polder_notnull = polder.where(polder>0.0)
AM_notnull = AM.where(AM>0.01)
# Then make one of the dataarrays negative, to plot both without losing species info
polder_negative = -1*polder_notnull
combined_da = AM_notnull.combine_first(polder_negative)
# cross section parallel to coastline
start_loosduinen = (75471,453198)
end_katwijk      = (87789.1,468649.0)
CS_par = imod.select.cross_section_line(combined_da, start=start_loosduinen, end=end_katwijk
# Plot both
fig, (ax1, ax2, ax3) = plt.subplots(3,1, figsize=(10,15))
combined_da.isel(y=35).plot(ax=ax1,y="z", cmap='RdYlBu')
ax1.set_title("Cross section perpendicular to coastline")
combined_da.isel(layer=10).plot.imshow(ax=ax2,cmap='RdYlBu')
```

```python
ax2.set_title("top view, -11mNAP")
CS_par.plot(ax=ax3,y="z", cmap = "RdYlBu")
ax3.set_title("Cross section along coastline: [Loosduinen - Katwijk]")

```

src\4-analyze\4.5.2-depth-interface-time.py

```python
"""
In this script the depth of the fresh-saline interface will be plotted over time:
    - Study area
    - 200y run
To retrieve an equilibrium depth and for plotting.
"""
#%%
import numpy as np
import os
import imod
import xarray as xr
import matplotlib.pyplot as plt
import pandas as pd
import pathlib
import geopandas
import matplotlib.patches as mpatches
#%%
os.chdir("c:/projects/msc-thesis")
# %% Import Data
conc_MM  = imod.idf.open(r"c:\projects\msc-thesis\data\4-output\4-scenario-200y-fixedrand\conc\conc_c1*.IDF").isel(species=0,drop=True)
gdf     = geopandas.read_file(r"c:\projects\msc-thesis\data\1-external\Polygon.shp")
like    = xr.open_dataarray(r"c:/projects/msc-thesis/data/2-interim/like.nc")
#%% Process data

raster        = imod.prepare.rasterize(gdf, like)
# Define f/s interface
fresh_upper = 0.150    # g/l
brack_upper = 8.0000   # g/l
# Remove negative concentrations
conc_meta = conc_MM.where(conc_MM != 1e30)
conc_meta = conc_meta.where(~(conc_meta < 0.0), other=0.0)

# index the conc dataset per timestep
times=np.array([
    "2014-12-31T23:59:59.000000000", "2015-01-01T00:00:00.000000000","2020-01-01T00:00:00.000000000","2025-01-01T00:00:00.000000000",
    "2030-01-01T00:00:00.000000000", "2035-01-01T00:00:00.000000000","2040-01-01T00:00:00.000000000","2045-01-01T00:00:00.000000000",
    "2050-01-01T00:00:00.000000000", "2055-01-01T00:00:00.000000000","2060-01-01T00:00:00.000000000","2065-01-01T00:00:00.000000000","2070-01-01T00:00:00.000000000",
    "2075-01-01T00:00:00.000000000", "2080-01-01T00:00:00.000000000","2085-01-01T00:00:00.000000000","2090-01-01T00:00:00.000000000","2095-01-01T00:00:00.000000000",
    "2100-01-01T00:00:00.000000000", "2105-01-01T00:00:00.000000000","2110-01-01T00:00:00.000000000","2115-01-01T00:00:00.000000000",
    "2120-01-01T00:00:00.000000000", "2125-01-01T00:00:00.000000000","2130-01-01T00:00:00.000000000","2135-01-01T00:00:00.000000000",
    "2140-01-01T00:00:00.000000000", "2145-01-01T00:00:00.000000000","2150-01-01T00:00:00.000000000","2155-01-01T00:00:00.000000000",
    "2160-01-01T00:00:00.000000000", "2165-01-01T00:00:00.000000000","2170-01-01T00:00:00.000000000","2175-01-01T00:00:00.000000000",
    "2180-01-01T00:00:00.000000000", "2185-01-01T00:00:00.000000000","2190-01-01T00:00:00.000000000","2195-01-01T00:00:00.000000000",
    "2200-01-01T00:00:00.000000000", "2205-01-01T00:00:00.000000000","2210-01-01T00:00:00.000000000","2215-01-01T00:00:00.000000000"], dtype="datetime64")
conc_MM_ar = dict()
depth_MM_ds = xr.Dataset()
for i in range(0,41):
    conc_MM_ar[i]  = conc_meta.isel(time=i, drop=True)
    depth_MM_ds[i] = conc_MM["z"].where(conc_MM_ar[i]
<fresh_upper).where(raster==1).min("layer").mean().compute()
depth_MM_da = depth_MM_ds.to_array()
# %% PLOTTING
fig,ax = plt.subplots(sharex=True)
plt.plot(times[:41], depth_MM_da, color=(35/255 , 130/255, 183/255))
plt.title("")
plt.ylabel("depth [m]")
plt.ylim((-90,0))
plt.xlabel("time")
path = pathlib.Path(f"reports/images/4-scenario-200y-fixedrand/depth-
interface_MM_200y.png")
plt.grid(axis="y")
#plt.savefig(path, dpi=200)

# %%

```

src\4-analyze\4.5-depth-interface.py

```python
"""
In this script the depth of the fresh-saline interface will be plotted over time for the study area, various scenarios
 - the idea is to come to an understanding of the artificial infiltration in the dune area,
    specifically the how much time it takes for freshwater to infiltrate  to a certain depth.

"""
#%%
import numpy as np
import os
import imod
import xarray as xr
import matplotlib.pyplot as plt
import scipy.ndimage.morphology
import pandas as pd
import pathlib
import geopandas
import matplotlib.patches as mpatches
#%%
os.chdir("c:/projects/msc-thesis")
#%%
mean_regridder = imod.prepare.Regridder(method="mean")
# Import data
like   = xr.open_dataarray(r"c:/projects/msc-thesis/data/2-interim/like.nc")
ibound = xr.open_dataarray(r"c:/projects/msc-thesis/data/2-interim/ibound_coarse.nc")
gdf    = geopandas.read_file(r"c:\projects\msc-thesis\data\1-external\Polygon.shp")
surface_level = xr.open_dataarray("data/2-interim/surface_level_without_sea.nc")
# output data
conc_OM    = xr.open_zarr(r"c:\projects\msc-thesis\data\1-external\data-25-run-1\conc-selection.zarr")["conc"].astype(np.float64)

conc_meta  = imod.idf.open(r"c:\projects\msc-thesis\data\4-output\3-scenario_FixedHead_rand\conc\conc_c*.IDF").isel(species=0,drop=True)
conc_meta = conc_meta.where(conc_meta != 1e30)
conc_meta = conc_meta.where(~(conc_meta < 0.0), other=0.0)
#%% Process data
raster        = imod.prepare.rasterize(gdf, like)
# Bounds for groundwater types
fresh_upper = 0.300    # g/l
brack_upper = 10.0000  # g/l

#%% Process data OM For study area
# NOTE: Regridding necessary to use the raster. the raster for the study area for 25m model can be found in 4.2 budgets waterbalance
conc_OM_1979   = mean_regridder.regrid(conc_OM.isel(time=0), like).where(raster==1)
conc_OM_1989   = mean_regridder.regrid(conc_OM.isel(time=1), like).where(raster==1)
conc_OM_1999   = mean_regridder.regrid(conc_OM.isel(time=2), like).where(raster==1)
conc_OM_2009   = mean_regridder.regrid(conc_OM.isel(time=3), like).where(raster==1)
conc_OM_2018   = mean_regridder.regrid(conc_OM.isel(time=4), like).where(raster==1)
conc_OM_re     = [conc_OM_1979, conc_OM_1989,conc_OM_1999,conc_OM_2009, conc_OM_2018]

depth_OM_ds = xr.Dataset()
for i in range(0,5):
    depth_OM_ds[i] = conc_meta["z"].where(conc_OM_re[i] < fresh_upper).where(raster==1).min("layer").mean().compute()
depth_OM_da = depth_OM_ds.to_array()
```

```python
#%% Process MM Data for study area
conc_meta_14 = conc_meta.isel(time=0, drop=True)
conc_meta_20 = conc_meta.isel(time=1, drop=True)
conc_meta_25 = conc_meta.isel(time=2, drop=True)
conc_meta_30 = conc_meta.isel(time=3, drop=True)
conc_meta_35 = conc_meta.isel(time=4, drop=True)
conc_meta_40 = conc_meta.isel(time=5, drop=True)
conc_meta_45 = conc_meta.isel(time=6, drop=True)
conc_meta_50 = conc_meta.isel(time=7, drop=True)

conc_meta_ar = [conc_meta_14, conc_meta_20, 
conc_meta_25,conc_meta_30,conc_meta_35,conc_meta_40,conc_meta_45,conc_meta_50]

depth_meta_ds = xr.Dataset()
for i in range(0,8):
    depth_meta_ds[i] = conc_OM["z"].where(conc_meta_ar[i]
<fresh_upper).where(raster==1).min("layer").mean().compute()
depth_MM_da = depth_meta_ds.to_array()
# %% PLOTTING
times_OM = np.array(["1979", "1989", "1999", "2009", "2018"], dtype="datetime64")
times_MM = np.array(["1984", "1989", "1994", "1999", "2004", "2009", "2014", "2018"], 
dtype="datetime64") # not the actual times but for plotting

times_sim_OM = ["0", "10", "20", "30", "39"]
times_sim_MM = ["5", "10", "15", "20", "25", "30", "35", "39"]
# they only share the time dimension. Fix this
fig,ax = plt.subplots(sharex=True)
plt.plot(times_OM, depth_OM_da, color=(147/255, 187/255, 226/255))
plt.plot(times_MM, depth_MM_da, color=(35/255 , 130/255, 183/255))
plt.grid(axis="y")
#plt.title("Depth of fresh-saline interface over time")
plt.ylabel("depth [m]")
plt.ylim((-80,-50))
plt.xlabel("time [y]")
blu_patch     = mpatches.Patch(color=(147/255, 187/255, 226/255)  ,     label='Original 
Model')
orn_patch     = mpatches.Patch(color=(35/255 , 130/255, 183/255),     label='Metamodel ')
ax.legend(handles=[blu_patch, orn_patch],loc="lower right")
path = pathlib.Path(f"reports/images/depth-interface_OM_MM_cond1.png")
plt.savefig(path, dpi=200)

#%%
# MM
times_MM = ["2014", "2020", "2025", "2030", "2035", "2040", "2045","2050"]
fig,ax = plt.subplots()
plt.plot(times_MM, depth_MM_da)
plt.title("Depth of fresh-saline interface, MM")
plt.ylabel("depth [m]")
#%%

# %%

```

src\5-visualize\5.1-cross-sections-overview.py

```python
"""
Give a top view plot of the cross sections:
    - along coastline (see 4.1, 4.3)
    - perpendicular to coastline (see 4.4)
"""
#%%
import numpy as np
import os
import imod
import xarray as xr
import matplotlib.pyplot as plt
import scipy.ndimage.morphology
import pandas as pd
import pathlib
import geopandas as gpd
from shapely.geometry import Point, LineString
import contextily as cx

#%%
os.chdir("c:/projects/msc-thesis")
#%%
starts = [
    (75000.0, 459948.0), # CS1
    (77423.0, 462817.0),
    (79234.0, 464828.4),
    (81880.4, 467911.6),
    (83718.9, 469707.8),
]
ends = [
    (87591.0, 449868.0), # CS1
    (92828.2, 450002.0),
    (93914.6, 452120.1),
    (95223.9, 455793.2),
    (96393.9, 460431.4),
]

CS1_s = Point(starts[0])
CS1_f = Point(ends[0])

CS2_s = Point(starts[1])
CS2_f = Point(ends[1])

CS3_s = Point(starts[2])
CS3_f = Point(ends[2])

CS4_s = Point(starts[3])
CS4_f = Point(ends[3])

CS5_s = Point(starts[4])
CS5_f = Point(ends[4])

CS1 = LineString([CS1_s, CS1_f])
CS2 = LineString([CS2_s, CS2_f])
CS3 = LineString([CS3_s, CS3_f])
CS4 = LineString([CS4_s, CS4_f])
CS5 = LineString([CS5_s, CS5_f])
```

```python

gdf = gpd.GeoDataFrame(data={"Cross section":["CS1", "CS2", "CS3", "CS4", "CS5"]},
geometry=[CS1, CS2, CS3, CS4, CS5])
gdf.crs = "EPSG:28992"
gdf_wm = gdf.to_crs(gdf.crs, epsg=28992)

ax=gdf.plot(figsize=(12,12))
cx.add_basemap(ax, crs="EPSG:28992")
ax.set_axis_off()
ax.set_title("Cross sections perpendicular to coastline")

# %% CS ALONG COASTLINE - See (4.4-Species)

start_loosduinen = Point(75471,453198)
end_katwijk      = Point(87789.1,468649.0)
CS_long = LineString([start_loosduinen, end_katwijk])

long = gpd.GeoDataFrame(data={"Cross section along coastline"}, geometry=[CS_long])
long.crs = "EPSG:28992"
long_wm = long.to_crs(long.crs, epsg=28992)

ax=long.plot(figsize=(12,12))
cx.add_basemap(ax, crs="EPSG:28992")
ax.set_axis_off()
ax.set_title("Cross sections along to coastline")

# %%

```