ARISTOTLE UNIVERSITY OF THESSALONIKI
DEPARTMENT OF GEOLOGY
Laboratory of Engineering Geology and Hydrogeology

Research Project (Code 97824)

Groundwater depletion. Are Eco-friendly Energy Recharge Dams a solution?

REPORT OF WORK PACKAGE 3 / (WP3/D3-5)

PRINCIPAL INVESTIGATOR:
KAZAKIS NERANTZIS
Dr. HYDROGEOLOGIST

Funding: Hellenic Foundation for Research

& Innovation H.F.R.I.

H.F.R.I. Hellenic Foundation for Research & Innovation

July 2022

Table of Contents

1 Introduction 3
2 Dam code 4
2.1 Overview 4
2.2 Hydropower in Greece 4
2.2.1 Dams in Greece 6
2.2.2 Small Agriculture Dams 6
2.2.3 Ecological flow 7
2.3 Simulation Model 11
2.4 Optimization Model 17
2.5 Optimization Algorithms 18
2.5.1 Metaheuristic optimization algorithms 18
2.5.2 Harmony Search Algorithm 18
2.6 Harmony Search Algorith on Dams 24
2.7 Conceptual Model 27
2.8 Source Code 28
3 Software for dam operation 32
3.1 Overview 32
3.2 Python for GUIs –Tkinter lybrary 32
3.3 Source Code of the GUI 34
3.4 Graphical User Interface – Software 40
3.5 Cost scenarios for pumping from an underground aquifer 43
3.5.1 Introduction 43
3.5.2 Calculation of Pumping Costs 46
3.6 DAM-MAR Model 50
4 Snow algorithm 63
4.1 Snow parameters 66
4.2 Algorithm description 67
4.3 Snow parameters evaluation 69
4.3.1 Primary data collection 69
4.4 Source Code 70
5 Meteorological parameters 87
5.1 Standardized Precipitation Index (SPI) and Standardized Precipitation Evapotranspiration Index (SPEI) 87

5.1.1 Forecasted SPEI 94
5.2 PDSI 96
5.3 Rain intensity 96
6 Groundwater Level measurements 101
7 Geoelectrical measurements 106
7.1 Site of the monitoring - Instruments 110
7.1.1 Weather station and air conditions monitoring 110
7.1.2 Soil electrical resistivity monitoring 111
7.1.3 Soil temperate and soil water content monitoring probe 111
7.2 Methodology 111
7.2.1 Weather station data 112
7.2.2 Temperature and water content probe 112
7.2.3 Electrical resistivity tomography 113
7.2.4 Temperature compensation of the inversion results 116
7.2.5 Time durations estimation of water infiltration 117
7.2.6 Overview of the collected data and the inversion results 119
7.3 Precipitation incidents 121
7.4 Conclusions 140
8 Conclusions 142
9 References 143

1 Introduction

Within this report included all results from the work package 3. More specific within this report are summarized all deliverables including a) dam code (D3-1), b) software (D3-2) and c) snow algorithm (D3-3). Additionally, is presented the analysis of the meteorological parameters, the results of the chemical analysis of surface and groundwater samples and the geoelectrical measurements in the study areas. Finally, during the period of work package 3 the research team have been participated in two conferences and published five (5) articles until May 2022. The conferences and the article titles are presented below.

- Voudouri K. A., Ntona M. M., Kazakis N.. (2021) Investigating the snow water equivalent in Greece. 15th International Conference on Meteorology, Climatology and Atmospheric Physics-COMECAP, Ioannina, Greece, 26-29 September 2021, pp. 315-319.
- Ntona M.M., Kazakis N. (2022). An Overview of managed aquifer recharge applications using simulation models. 12th International Hydrogeological Conference, Cyprus, 20-22 March 2022. pp. 177-180.
- Ntona M.M., Busico G., Kazakis N., Mastroccico M. (2022). Simulating historical, actual and future water balance in mountainous watershed. 12th International Hydrogeological Conference, Cyprus, 20-22 March 2022, pp. 172-175.
- Karakatsanis D., Patsialis T., Kougias I., Ntona M.M., Theodosiou N., Kazakis N. (2022). Simulation software for small eco-friendly energy recharge dams. 12th International Hydrogeological Conference, Cyprus, 20-22 March 2022, pp. 115-119.
- Patsialis T., Karakatsanis D., Kougias I., Theodosiou N., Kazakis N. (2022). The small hydropotential in Greece. Current projects and future challenges. 12th International Hydrogeological Conference, Cyprus, 20-22 March 2022, pp. 378-381.

2 Dam code

Within this report is presented the code for the dam operation and the conceptual model (milestone M3.1). The dam code is part of work package 3, while also constitute a milestone (dam code - M3.2) for the project. Within this report is also presented the dam dynamic of Greece and general information of dam operation.

2.1 Overview

Due to the effects of climate change and population growth, reservoirs play an increasingly important role in water resources management. Reservoirs can be used for multiple-purposes such as irrigation, municipal and industrial water supply, hydropower generation, flood protection, water quality management, recreation, low flow augmentation and so on (Nay Myo Lin 2016). Development of optimal operational solutions for multi-scope reservoir systems is often complicated by a multiplicity of conflicting project uses and purposes. The complexity of their operation has also increased especially in hydropower reservoirs, due to the increasing variability of hydroelectric generation. The management of a multi-reservoir system is complex due to the curse of dimensionalities, nonlinearities and conflicts between different objectives. The decision model involves oprimization and hydrological simulation models (combined simulation–optimization). Optimization techniques have shown high efficiency when used with simulation modeling and the combination of the two methods had given the best results in reservoir management (S. S. Fayaed 2013).

2.2 Hydropower in Greece

The production of energy through the utilization of the available hydrodynamic potential of an area, is a process that is applied through many years and continues to improve with the development of technology (Kaldellis JK 2007). In Greece, the development of hydropower plants was historically implemented by the incumbent public power utility. Development by private entities and individuals concerns small-scale projects are began in 1994 with the corresponding law that provided for such development projects (RAE) with the majority of cases concerning projects with a capacity of 0.5 to 3 MW. Usually, such projects are not visible from crowded places because they do not involve significant water collection and storage, nor the

construction of large dams and reservoirs, making them more environmentally friendly (Patsialis, et al 2016). According to the existing legislation of Greece, small hydroelectric means small units that use watercourses or small dams for electricity production and their power does not exceed 15MW (YPEKA).

Also, due to the morphology and climate of Greece, the hydroelectric plants perform relatively high efficiency indices (EMSYE) with favourable techno-economic analyzes (Kaldellis, J.K 2006) Today, a large part of the energy produced from RES in the world still comes from water. The advantages are significant and, in combination with the increasing demands for more energy, they become even more important.

The utilization of the small hydro potential of Greece is a big gap in the development of RES (CRES). According to the European Energy Institute (EEI), in the last 25 years there have been no initiatives by the authorities to rationalize the institutional framework and exploit the country's small hydro potential, as manifested in other countries. In Greece, the growth rate of the four plants per year is rather low and it takes many years to reach satisfactory levels. The European Renewable Energy Federation (EREF) has set high goals for increasing RES and reducing gas emissions.

Small hydropower electric plants (SHEPs) have the highest energy efficiency (energy produced per unit of installed power) of all RES technologies. Indicatively, it is stated that the average energy efficiency of SHEPs exceeds 40% while the corresponding efficiency amounts to 25% for wind and 16% for photovoltaics. According to recent data from PPC (Public Power Corporation), each kWh produced by SHEP is compensated with a price that amounts of about 49% of the average energy price of all RES and about 26% of the average cost of a photovoltaic kWh.

According to the data of RAE and CRES, the percentage of utilization of the available hydro potential of Greece does not exceed 12%, while other countries of the European Union have exceeded 70%. In our country, at the moment 119 projects are operating with a total installed capacity of 243.9MW. Also, based on RAE data, it appears that the growth of MUS in recent years is minimal and ranges from 3-5%, at the moment that the corresponding rate of increase in the installed capacity of other technologies ranged from 19% to 70% (wind - photovoltaic) while the total rate of increase in the capacity of RES reached 43%.

In recent years, efforts have been made to utilize existing hydraulic projects (water supply and irrigation networks, dams, etc.) for energy production, giving a second use to water management (Droege, P 2009). These applications have a high efficiency index, as they utilize existing infrastructure. Services such as EYDAP, DEYA, TOEB are some that participate in such development projects in Greece.

It is clear that SHEPs can significantly contribute to sustainable regional and local development but also to achieving energy goals. It is imperative to utilize the hydrodynamics of our country at least at the average level of European countries.

The small hydro potential of our country is an environment for the development of many promising projects. Small watercourses in the mountains, water supply and irrigation networks, small dams, etc. are cases where with proper planning important energy production projects can be developed. More flexible legislation is needed in such cases with faster procedures. RAE already excludes from the power generation license the projects smaller than 50KW.

2.2.1 Dams in Greece

The role of dams is to regulate the flow of rivers and to use valuable water resources in a more cost-effective and safer way. Their construction began many centuries ago before questions and concerns were raised about the environmental changes they cause and objections to their utilitarian expediency. The first dams were built mainly to provide flood protection and water storage for irrigation and water purposes, while later they were used for hydroelectric power generation, fish farming, tourism and recreation. Today, dams have different characteristics than other civil engineering structures, are much larger than in the past, utilize knowledge of hydrology and hydraulics, and the magnitude of the direct or indirect, economic or non-economic impact has increased.

In Greece, more than 150 dams have been built, which range from small to very large. The table in the appendix presents a list of existing dams in Greece and their technical characteristics. In none of the cases of the Greek dams is there the use of artificial enrichment, although Greece is a country where the management of water resources is a critical and sensitive issue.

2.2.2 Small Agriculture Dams

Small reservoirs are created by constructing perpendicular to the flow of dams in small and medium-sized permanent flow streams. These reservoirs are mainly used

for irrigation purposes as they contribute to the annual balancing of water demand on the downstream farm. Depending on the species, the diversity of the crops, the subsoil of the irrigation method but also the weather conditions, the water demand shows great seasonal fluctuations. Also important parameter related to the demand for irrigated water is the know-how of the workers and the disposition for proper water consumption. In the literature there are several irrigation methods with the most common being the method of rotational distribution and the method of free demand.

2.2.3 Ecological flow

Ecological flow, also known as environmental flow or environmental water allocation, embodies the essence of maintaining the natural rhythm and balance of freshwater ecosystems. It encompasses the quantity, timing, and quality of water flows necessary to sustain the ecological integrity and biodiversity of rivers, streams, and estuaries, while also supporting the vital services they offer to both humans and the environment. At its core, ecological flow mirrors the natural ebb and flow of water in rivers and streams. It acknowledges the importance of seasonal variations, including floods, low flows, and base flows, as well as the intricate interplay between these flow events and the surrounding landscape. By preserving these natural patterns, we safeguard the intricate web of life that thrives within aquatic habitats. The significance of ecological flow extends far beyond mere water movement. It underpins a myriad of ecosystem services that benefit society, including water purification, flood regulation, sediment transport, and nutrient cycling. Moreover, it sustains the diverse array of plants and animals that inhabit freshwater environments, providing essential habitat and resources for their survival. Riparian zones and floodplains, intimately connected to riverine ecosystems, rely on the periodic flooding and drying cycles facilitated by ecological flow. These dynamic habitats serve as corridors for wildlife movement, hubs of biodiversity, and buffers against the impacts of floods and droughts. Disrupting the natural flow regime can unravel the intricate tapestry of life woven within these ecosystems. However, human activities such as dam construction, water abstraction, and land use changes often disrupt the delicate balance of ecological flow. Altering flow patterns can lead to habitat degradation, loss of connectivity, and declines in species diversity. Consequently, there is an urgent need for proactive management strategies that prioritize ecological needs while reconciling competing water demands. Climate change further complicates the

picture, exacerbating existing challenges related to ecological flow. Shifts in precipitation patterns, rising temperatures, and increased hydrological variability pose significant threats to water resources and ecosystem resilience. Adapting to these changes requires innovative solutions and collaborative efforts at local, regional, and global scales. In essence, ecological flow serves as a cornerstone of freshwater conservation and sustainable water management. It represents a harmonious coexistence between humans and nature, where the needs of both are met in a balanced and equitable manner. By safeguarding the natural flow regime of rivers and streams, we ensure the continued health and vitality of these precious ecosystems for generations to come.

In the present work, since the purpose of the reservoir was to define a minimum required ecological supply, a maximum value during the dry season was set and the remaining values as a normal distribution of random numbers according to Equation 1.

Equation 1:
$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

Where μ is the mean value and σ is the standard deviation. The assumption of random distribution of water with minimum and maximum flow simulates the free use of stream water by downstream consumers with the logic of the Monte Carlo method which usually happens in practice. Monte Carlo method provides a flexible and robust framework for assessing ecological flow in streams, taking into account the inherent uncertainty and variability of natural systems. By integrating probabilistic approaches with hydrological and ecological modeling, it enables stakeholders to make more informed and adaptive decisions to sustainably manage water resources and protect stream ecosystems. The Monte Carlo method can be a valuable tool for assessing ecological flow in streams, particularly in situations where there is uncertainty or variability in the data inputs or model parameters.

Expanding further on the concept of converting small reservoirs into small hydroelectric power stations (MIS), let's explore the intricacies and benefits of this innovative approach in greater detail. In regions where reservoirs typically accumulate water volumes that exceed immediate irrigation demands throughout the annual cycle, there exists a promising opportunity to harness this surplus energy for electricity generation. By repurposing existing reservoir infrastructure, such as dams and intake structures, into mini hydroelectric facilities, it becomes feasible to

capitalize on the dynamic energy surplus while simultaneously addressing both water demand and energy requirements. This dual-purpose functionality not only enhances the sustainability of water resource management but also contributes to the diversification of the energy mix, thereby fostering resilience and self-sufficiency in local energy supply. While it's true that small-scale hydroelectric plants may not achieve the same level of efficiency as larger-scale counterparts, their construction costs are significantly mitigated by the utilization of pre-existing facilities. This cost-effectiveness makes MIS an economically viable solution for regions seeking to optimize their energy infrastructure without incurring exorbitant capital investments. Moreover, the modular nature of small hydroelectric projects allows for scalability and adaptability to varying site conditions and resource availability, further enhancing their appeal as sustainable energy solutions.

Multipurpose reservoirs can be categorized based on the allocation of useful volumes for various purposes. Some reservoirs segregate volumes for distinct uses, such as irrigation, flood control, and hydroelectricity generation. In contrast, others integrate all functions within a single volume, allowing for seamless resource allocation and operational flexibility based on demand fluctuations. This integrated approach maximizes the utilization of available water resources while minimizing wastage and inefficiencies, thereby optimizing the overall performance and resilience of the reservoir system. When transitioning a single-use reservoir into a multipurpose facility, careful consideration must be given to optimizing the utilization of available volumes for diverse functions. This integration facilitates efficient resource management while maximizing the benefits derived from the reservoir infrastructure.

However, it's essential to recognize that reservoirs configured for multipurpose use may not be suitable for flood protection purposes, as their design prioritizes maintaining high water levels to ensure consistent hydroelectric potential. Therefore, additional measures such as the construction of levees or embankments may be necessary to mitigate flood risks and safeguard surrounding communities and ecosystems. In summary, the conversion of small reservoirs into small hydroelectric power stations represents a forward-thinking approach to sustainable water and energy management. By leveraging surplus water resources to generate clean electricity, these multipurpose facilities contribute to environmental conservation, economic development, and social well-being. Through strategic planning, innovative

engineering, and collaborative partnerships, we can harness the power of nature to build a more resilient and prosperous future for generations to come.

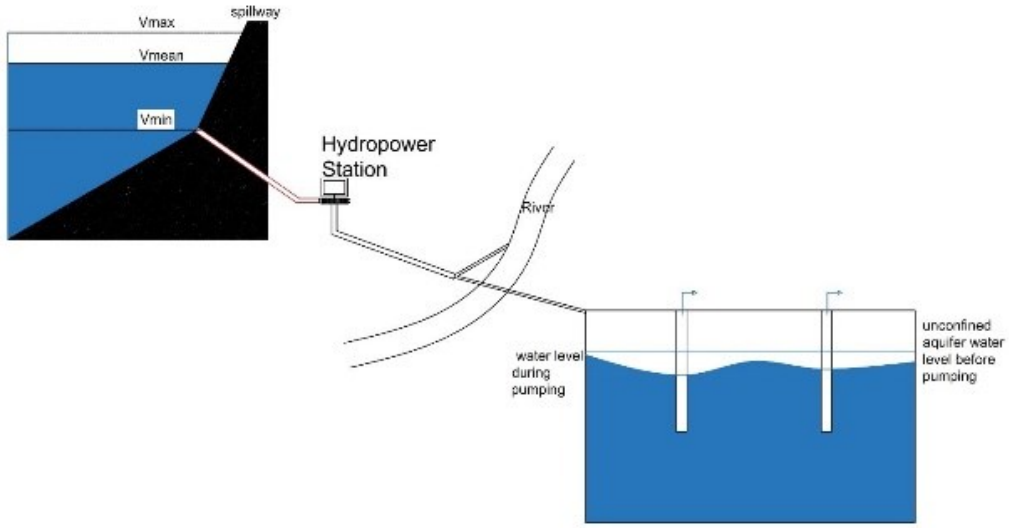

Figure 2.1 Schematic representation of a reservoir conversion into a small hydroelectric plant.

2.3 Simulation Model

The useful volume of the reservoir in relation to the height of the water in the dam and its surface is given by the equation 2.

Equation 2 : $$V = \int_{h1}^{h2} Adh$$

The current volume at any given time according to the continuity equation is the linear composition of the cumulative input curve, the cumulative output-loss curve and the cumulative consumption curve. Two restrictions are imposed on the above volume, the first concerns a minimum volume of Vmin corresponding to a level below which water is unusable for energy production. The second concerns a maximum volume of Vmax corresponding to a level beyond which the water through the dam overflow ends up at the receiver. The current volume for time T is given by equation 3.

Equation 3 : $$V_{cur} = \int_0^T Q_{in}dt + \int_0^T Q_{out}dt + \int_0^T Q_{irrigation}dt + \int_0^T Q_{turbine}dt$$

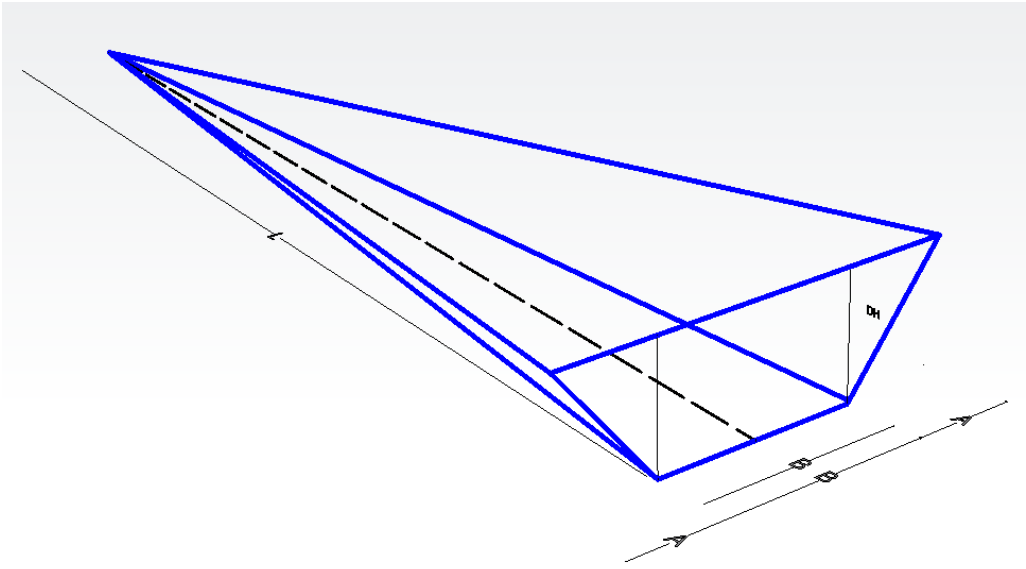

Figure 2.2 Geometric reservoir model.

Evaporation is ignored in the model due to the small size of the reservoirs. The geometric model is the one shown in Figure 2.2. Given that this type of dams is constructed perpendicular to the flow of small torrents, the above geometric model is quite close to reality and offers an easy and simple way of describing the useful volume with relatively little geometric data.

Understanding the dynamics between water level and volume in small multipurpose dams is paramount for effective water resource management. This relationship is vividly captured by the storage curve, also known as the stage-storage curve, a fundamental tool in hydrology and reservoir engineering. This curve graphically illustrates how the volume of water stored in the reservoir fluctuates in response to changes in water level. Derived from the reservoir's physical characteristics such as its shape, size, and topography, the storage curve serves as a blueprint for operational decision-making. It embodies the intricate interplay between inflows, outflows, and storage capacity, providing valuable insights into the reservoir's behavior under various conditions.

At its essence, the storage curve reflects the principle of water accumulation: as the water level rises, the volume of stored water increases. This relationship, while intuitive, often exhibits nonlinear behavior. Factors such as reservoir geometry, sedimentation patterns, and hydraulic conditions influence the rate of volume increase for each incremental rise in water level. Reaching a critical juncture, the reservoir attains its maximum storage capacity, symbolized by the crest elevation of the dam or spillway. Beyond this point, further increases in water level result in minimal or no additional volume gain, underscoring the importance of efficient reservoir management to mitigate flood risks and optimize water allocation. Conversely, during drawdown periods, as water is released from the reservoir, the water level decreases, and the stored volume diminishes accordingly. This drawdown process, characterized by the drawdown curve, is equally integral to understanding reservoir behavior and optimizing operational strategies. Furthermore, the storage curve may exhibit hysteresis, a phenomenon where the relationship between water level and volume during filling differs from that during drawdown. This hysteresis, influenced by factors such as sedimentation dynamics and structural characteristics, adds a layer of complexity to reservoir management but also offers valuable insights into system behavior and resilience. Ultimately, understanding the intricacies of the storage curve empowers water managers to make informed decisions regarding flood control, water supply management, hydropower generation, and ecological preservation. By harnessing the knowledge embedded within this curve, managers can optimize reservoir operations, ensure the sustainable utilization of water resources, and safeguard the resilience of aquatic ecosystems for generations to come.

For all the above reasons, it is quite difficult to make a single model of the water volume-still water curve. In the present work we followed the following strategy: in the event that there is no data at all for the curve in question then the function between current useful volume and change in height is calculated in Equation 4.

Equation 4 :

$$V_{cur} = \frac{1}{3} L * A_{base} = \frac{1}{3} L * \frac{DH_{cur}(B + (2A + B))}{2} \rightarrow V_{cur} = \frac{DH_{cur} * L * (A + B)}{3}$$

Equation 4 is a linear relationship between useful volume of water in the reservoir and change in water level. The water surface (As) and water height (Dh) curve in the reservoir is also useful. The calculation of the equation of the two variables is presented in equation 5-6 and Figure 2.3.

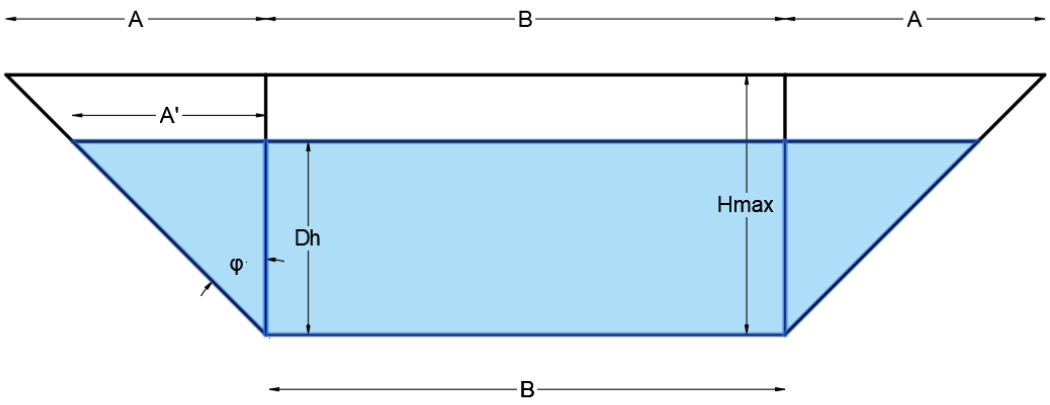

Figure 2.3 Cross Section of reservoir model.

Equation 5 :

$$\tan \varphi = \frac{A}{H_{max}} = \frac{A'}{Dh} \rightarrow A' = \frac{A * Dh}{H_{max}}$$

Equation 6 :

$$A_S = \frac{(B + 2A') * L}{2} = \frac{\left(B + \frac{2 * A * Dh}{H_{max}}\right) * L}{2}$$

In the literature, the current volume curve and the reservoir surface curve are usually shown in a common diagram. For this model the diagram is presented in Figure 2.4.

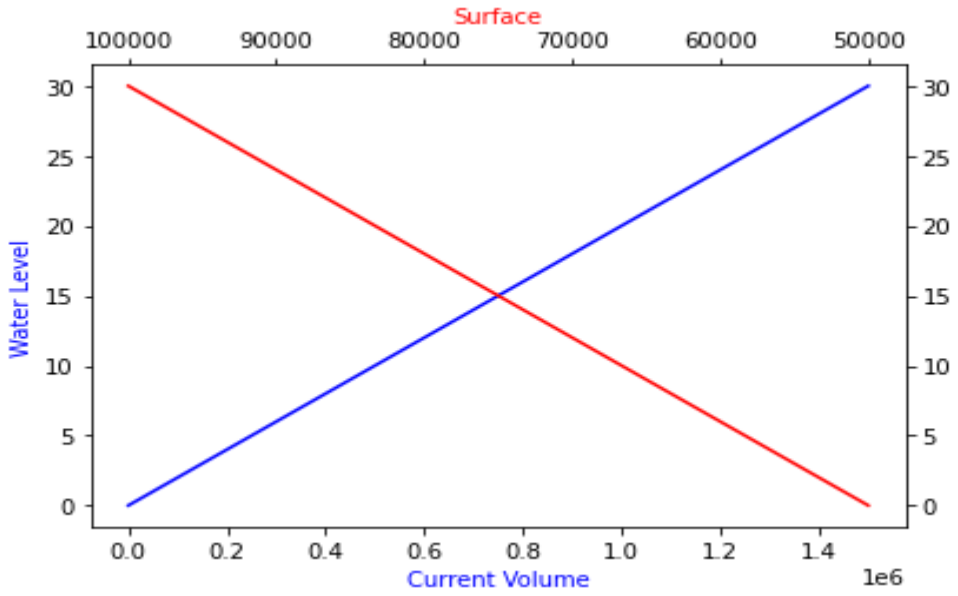

Figure 2.4 reservoir level-storage curve.

The current maximum and minimum water volume are also shown in Figure 2.5.

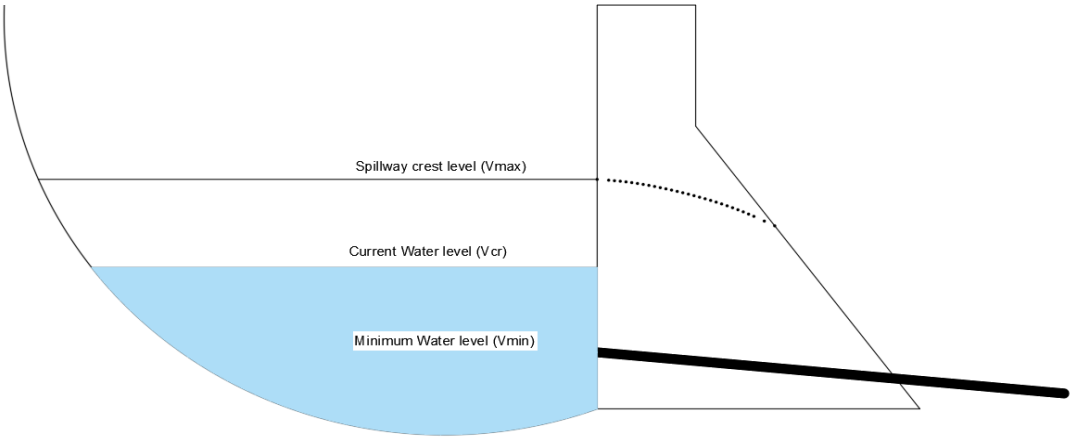

Figure 2.5 Reservoir water's volume.

In the case where there are water volume and supply level data as for example in Table 2.1.

Table 2.1 Volume and water level.

Elevation	Volume	Water Area
350	0	0
352	1176.564519	1176.5645
354	5915.829343	7092.3939
356	15216.27228	21132.102

358	29745.10642	44961.379
360	50028.31289	79773.419
362	76508.23104	126536.54
364	109570.6355	186078.87
366	149559.9864	259130.62
368	196788.9889	346348.98
370	251545.0337	448334.02
372	314094.7691	565639.8
374	384687.4787	698782.25
376	463557.656	848245.13
378	550927.0171	1014484.7
380	647006.1085	1197933.1
382	751995.6128	1399001.7
384	866087.4273	1618083
386	989465.564	1855553
388	1122306.912	2111772.5
390	1264781.886	2387088.8
392	1417054.991	2681836.9
394	1579285.302	2996340.3
396	1751626.896	3330912.2
398	1934229.223	3685856.1
400	2127237.437	4061466.7
402	2330792.694	4458030.1
404	2545032.413	4875825.1
406	2770090.516	5315122.9
408	3006097.638	5776188.2
410	3253181.327	6259279
412	3511466.212	6764647.5
414	3781074.172	7292540.4
416	4062124.474	7843198.6

In this scenario, employing a more sophisticated approach offers a rational approximation of the storage curves compared to the simplistic linear model. By incorporating the nuances of reservoir behavior, such as nonlinear relationships

between water level and volume, we obtain a more accurate representation of the system dynamics. The curves derived from this refined modeling approach encapsulate the complexities inherent in reservoir operations, accounting for factors like reservoir geometry, sedimentation patterns, and hydraulic characteristics. Unlike the linear model, which oversimplifies the relationship between water level and volume, these curves provide a nuanced depiction that aligns more closely with real-world observations. Figure 6 visually presents the intricate forms of these storage curves, illustrating the dynamic interplay between water level and volume across varying operational scenarios. Each curve tells a story of reservoir behavior, offering insights into how the system responds to inflows, outflows, and external influences. By embracing this more comprehensive modeling framework, water managers gain a deeper understanding of reservoir dynamics and can make more informed decisions regarding water allocation, flood management, and environmental stewardship. With Figure 2.6 as a visual aid, stakeholders can navigate the complexities of reservoir operations with confidence, ensuring the sustainable utilization of water resources while safeguarding the resilience of ecosystems and communities.

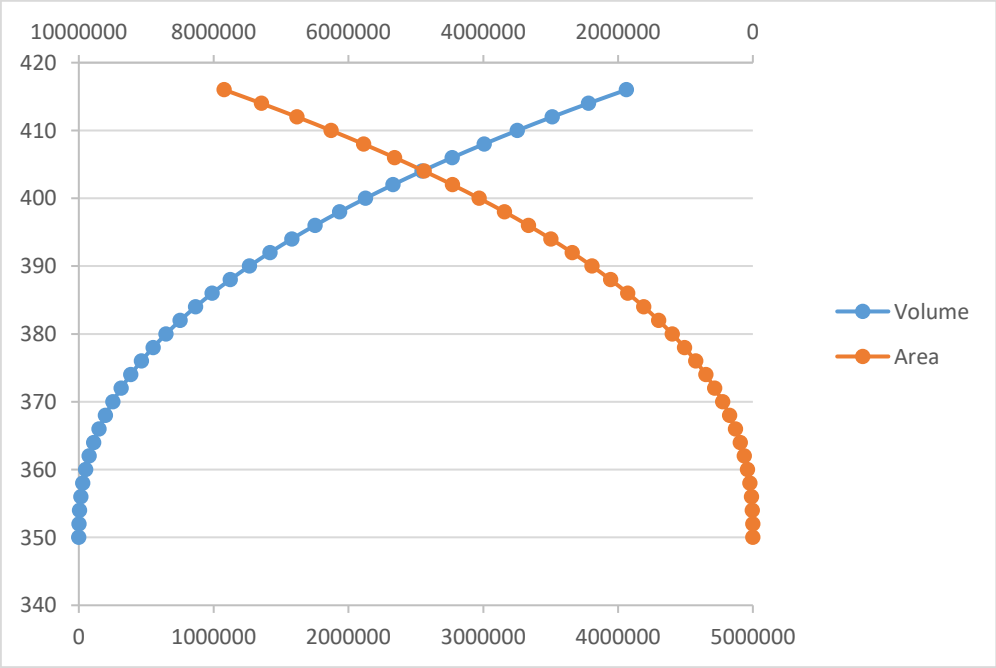

Figure 2.6 reservoir real water volume –level and water surface –level curves.

To implement such an addition, the reservoir code should be modified in a way that is presented in the code appendix.

2.4 Optimization Model

The goal of the optimization model is to find the vector of daily water supplies per year used to generate electricity. That is, the problem has 365 unknown variables. Restrictions include ensuring a minimum volume in the reservoir a maximum volume, ensuring a minimum and maximum ecological flow rate. Also the volume of the reservoir is limited by a maximum value beyond which the excess water escapes from the overflow. The objective function is the annual energy production from the turbine. The energy production is given by equation 7.

Equation 7 : $$E_{turbine} = \rho * g * Dh * Q_{turbine}$$

E_{urbine} : The energy produced
ρ : The density of water
g : The acceleration of gravity
Dh : The net height of water drop
$Q_{turbine}$: The discharge of water

Based on the above the model is presented in the equations of Table 2.2

Table 2.2 Equations for the model optimization.

Variables	Restrictions	Objective Function
$\overrightarrow{Q_{turbine}} = \begin{cases} x1 \\ x2 \\ x3 \\ .. \\ .. \\ x364 \\ x365 \end{cases}$	$V_{min} < V_{cur} < V_{max}$ $Qec_{min} < Qec < Qec_{max}$	$F = \sum_{0}^{365} \rho * g * Dh * Q_{turbine}$

The model is solved by the harmony search algorithm and the whole process is organized in Python language.

2.5 Optimization Algorithms

2.5.1 Metaheuristic optimization algorithms

Metaheuristic algorithms as a rule are inspired by natural or artificial procedures. In that sense they imitate natural or artificial phenomena that continuously advance to better states in order to carry out internal search processes. Genetic Algorithms (Holland, 1973) is probably the most wide-spread optimization technique and imitates the natural evolution process according to Darwin's theory. Many methods known as evolutionary computational methods, such as Evolution Strategies, Evolutionary Programming, Genetic Programming, are based on the principle of evolution. Simulated Annealing is a successful algorithm developed in the early 80's relating to an artificial phenomenon, the metals' characteristic of recrystallizing in an annealing process (Kirckpatrick, 1983). The interest of those involved in optimization continued to be sustained in developing new algorithms such as the Particle Swarm Optimization (Eberhart and Kennedy, 1995) and Ant Colonies (Dorigo, 1996), which were inspired by the behavior of living organisms.

Geem in 2001 introduced Harmony Search Algorithm (HSA), a modern metaheuristic algorithm inspired from the music creation process (Geem et al., 2001). HS Algorithm, which is used in the present study, is a powerful and efficient tool with the extra advantage of having a simple structure. These characteristics attracted the interest of those involved in the optimization field. Initially HSA was designed for the optimum design of water distribution networks (Geem et al., 2002). Since then, there has been sustained and increasing interest in HSA applications. In the current literature, apart from water engineering optimization problems, one can find a vast variety of interesting implementations (Kougias and Theodossiou, 2010).

2.5.2 Harmony Search Algorithm

The relationship between music and mathematics has been close since the ancient times. Mathematicians tried to interpret the governing rules of mathematics using the art of music. On the other hand, composers tried to use mathematics in order to deeply understand music. During recent times, since the Baroque period, this bond has been strengthened. Sometimes as a conscious effort by musicians-composers and sometimes as part of a rumored and almost mystical relationship, mathematics and music came closer. Iannis Xenakis represents a special example. His deep knowledge both in mathematics and music is illustrated in his work on the use of mathematical

functions to compose music (1992) distinguishing him among the most eminent music figures of the 20th century. The Harmony Search Algorithm is a stochastic meta-heuristic method based on the sequential production of possible solutions. It belongs to the category of "neighborhood meta-heuristics" that produce one possible solution (called "harmony") in each iteration. Every possible solution consists of a set of values of the decision variables of the function that needs to be optimized. During the optimization process, a number of "harmonies" equal to the "Harmony Memory Size" are stored in the "Harmony Memory" (HM), a database that includes the produced set of solutions. The optimization process is completed as soon as the predefined total number of iterations has been achieved (Geem, et al. 2001).

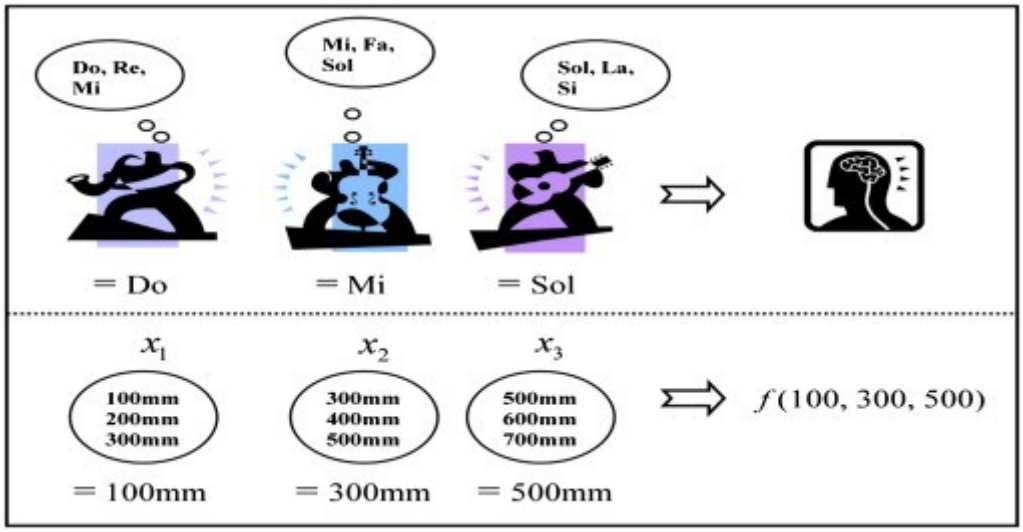

Figure 2.7 Harmony search is inspired by the improvisation process of jazz musicians.

Following the definition of the decision variables, the Harmony Memory matrix is formulated. Harmony Memory is m×n matrix, where m is the Harmony Memory Size and n, the number of decision variables included in the objective function. Then, the algorithm begins producing and evaluating new "Harmonies" through the application of HSA's basic mechanisms:

1. Harmony Memory Consideration uses variables' values already stored in the Harmony Memory. This mechanism ensures that good solutions located during the optimization process will contribute to the formation of even better solutions.

2. Some of the solutions selected by the Harmony Memory Consideration mechanism will be slightly altered. This is the second mechanism of the algorithm named Pitch Adjustment and it is performed by selecting a neighboring value of the decision variables.

3. The third mechanism is Improvisation, which introduces new, random elements to the solutions. The probability of introducing such random values is (100-HMCR)%. In this way the variability of solutions is enriched.

After the creation of a new "Harmony", its performance is evaluated according to the corresponding value of the objective function. If this performance is better than that of the worst "Harmony" stored in the Harmony Memory, it replaces it. This procedure is repeated until the ending criterion, is reached.

The method consists of 4 individual steps.

- Production of initial harmonic memory
- Production of new harmony
- Evaluation of the new harmony and entry or not in the harmonious memory
- Production of initial harmonic memory

The harmonic memory is a matrix m×n where it represents the orchestra. Each line of the harmonic memory is a different musical instrument (problem variable) while each column of the harmonic memory is also a possible solution to a problem, i.e. an orchestra chord (simultaneous playing). The number of possible solutions, i.e. the columns of harmonic memory (n) is a parameter of the method and is called "Harmony Memory Size (HMS)" with typical values in the [10,100]. The number of lines (m), musical instruments, is a parameter of the original problem to be solved. Each column is also a combination of values of the variables, i.e. a possible policy or a solution vector or a chord. When creating the harmonic memory, the constraints must be checked so that it is not possible to enter in the harmonic memory a solution that does not comply with the constraints. In case such a thing is allowed, the import will be done with a penalty. Finally, each chord of harmonic memory is evaluated based on the objective function. The result of the evaluation is placed on a separate row table or as an additional line at the end of the harmonic memory. In short, harmonic memory is a set of possible solutions that is the initialization of the method (relies on it to be able to start).

Production of new harmony:

The next step of the method is the production of new harmony that includes three sub-processes. The new harmony is a new chord (solution vector) like those contained in the harmonic memory and it must be compatible with the constraints of the problem. Initially, 2 possibilities are defined, which in a complementary way define the 3 individual processes of production of new harmony.

These possibilities are:

• The possibility that the new harmony includes values from the harmonic memory (not necessarily from the same column). This possibility is called: Harmonic Memory Considering Rate (HMCR) and ensures that "good" solutions are inherited in the next new harmonies. Typical range for HMCR is interval [0.5 0.95].

• The possibility that the new harmony (solution vector) is slightly different from some existing one. This possibility is called: Pitch Adjustment Rate (PAR) and ensures the "tuning" of new solution vectors. The small change (δ) of a value based on equation 8 with the aim of the best result in the objective function.

Equation 8 : $$X_{new} = X \pm \delta$$

The PAR process simulates the mutation of genetic algorithms. According to the literature, a high probability of PAR implies a high probability of being trapped at a local maximum or minimum. As long as reference δ a value in the interval (0,1] was selected. PAR takes values in the interval [0,05 0,5]

Based on these two possibilities, all three sub-processes are defined. Although various combination techniques of the above 2 possibilities have been proposed for the production of new harmony the most common pattern is described in Figure 2.8:

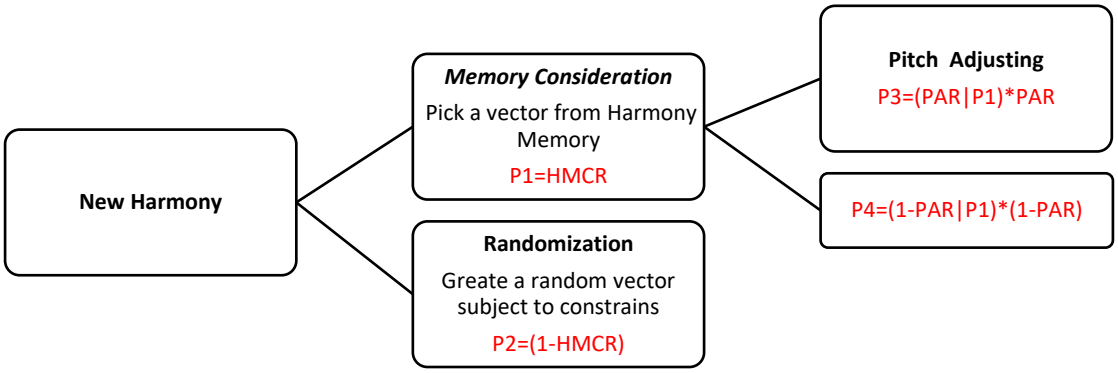

Figure 2.8 Flowchart for New Harmony Production with possibilities.

Evaluation of new harmony:

In the third and last step of the method, the performance of the new harmony in the objective function is evaluated subject to the constraints. In the case where the performance of the new harmony is better than the worst possible in the harmonic memory then the latter leaves, from the harmonic memory, and is replaced by the new harmony. Steps 2 (producing new harmony) and 3 (evaluating new harmony) are repeated for a respectable number of repetitions and thus the method is completed.

The success of convergence at the global optima of the possible policy space depends on the choice of HM, HMCR PAR parameters and the number of iterations. It should be emphasized that the method does not converge completely and therefore its executions are required.

At the Figure 2.8 described the full flowchart of the Harmony Search Algorithm.

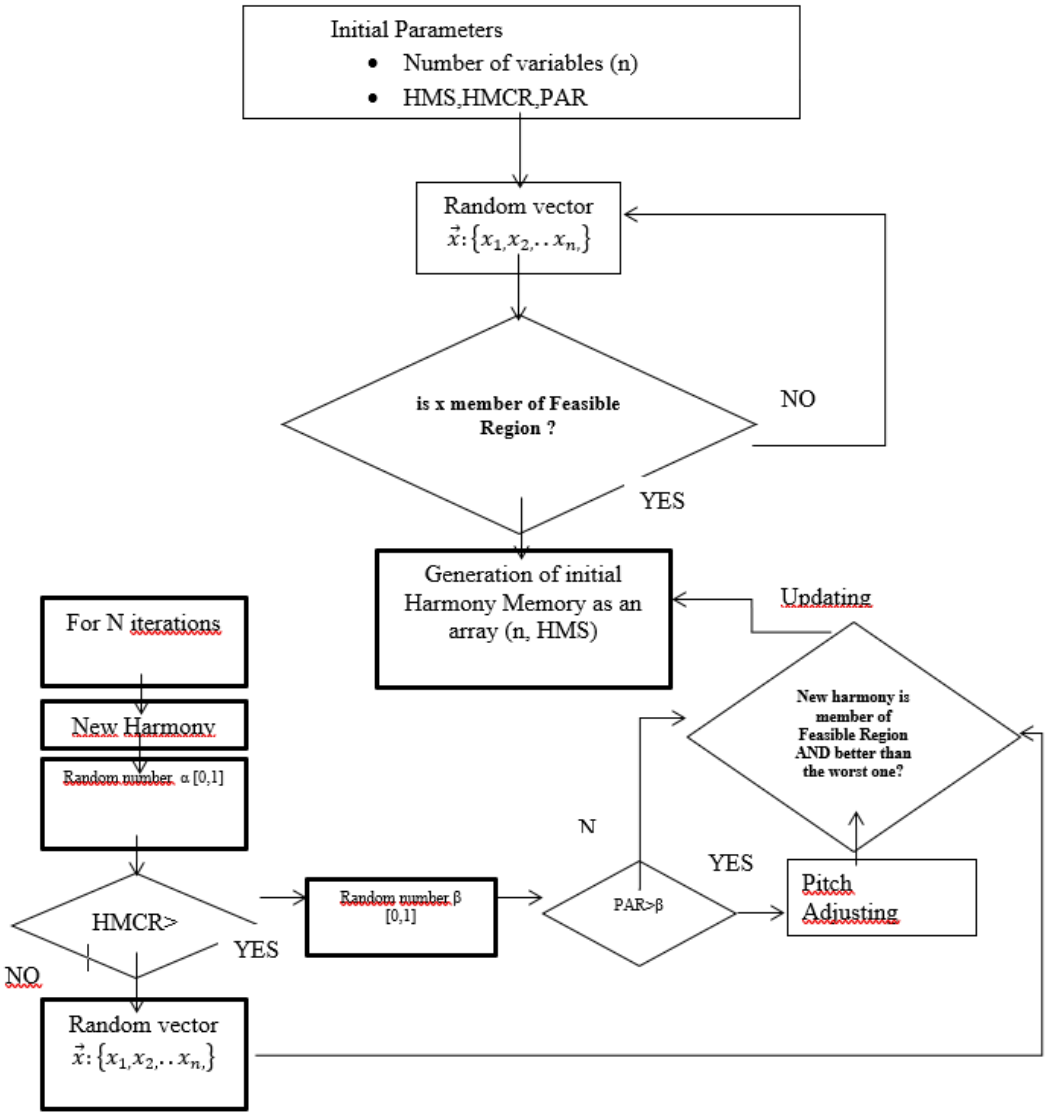

Figure 2.9 The full flowchart of the Harmony Search Algorithm.

2.6 Harmony Search Algorith on Dams

In recent years one can find several applications of the Harmony Search Algorithm on dam system optimization. Over the last twenty years, the HSA has been tested on both benchmark and real-world reservoir systems and has successfully identified areas of optimal solutions. Geem (2007) solved a multiple dam system problem in order to maximize the benefit from hydropower and the benefit from irritation, a typical multi objective problem between competing objectives. Results showed that the HS model found five different global optimal solutions with identical maximum benefit from hydropower generation and irrigation, while the enhanced GA model found only near-optimal solutions under the same number of function evaluations. In the literature, results are almost always compared with older ones, given that metaheuristic algorithms are a modern and constantly evolving technology. An improved version of the algorithm was proposed by Janatrostami et al (2010) in order to optimize Dez dam of Iran. In addition, HSA techniques have been developed by Hasan Torabi et al, (2020) in order to optimize the operation of Dez dam reservoir for a long time period (40 years). The goal was to supply the agricultural water demand of dam downstream. Bashiri-Atrabi et al. (2015) used HSA to minimize the water supply deficit and flood damages downstream of a reservoir (multi objective). In order to determine the flood damage objective function, they used a GIS database. HS showed promising results in terms of speed of convergence to an optimal objective function value compared with other techniques such as honey-bee mating optimization (HBMO) and a global optimization model (LINGO 8.0 NLP solver). A comprehensive methodology was developed to model reservoir operations. The main components of the process are collecting the required data (land use, river cross section, inflows, reservoir characteristics and water demands), run a hydraulic flood routing model, estimate the flood damage function and run a reservoir operation optimization model using HS algorithm. This study combines the HS algorithm, HEC-RAS 2010 river hydraulics simulation model and a geographic information system (GIS). Then the model was applied on multi-purpose Narmab reservoir in the province of Golestan, north of Iran.

One of the main advantages of HSA is the ability to combine with other optimization methods, modify and extend. Kougias et al, (2014) propose a hybrid optimization harmony search algorithm and applied on Huong Dien hydroelectric

dam, Vietnam. Furthermore, harmony search optimization algorithm is applied in the problem of design and operation optimization of Bakhtiari Dam by S. J. Mousavi et al (2017). In order to select spillway type and optimizing dimensions for Qeshlagh Dam, Mohammad R. Hassanvand (2018) et al, propose a multi-criteria decision-making method using meta-heuristic (HSA). Kougias et al (2016) developed a GUI- based interface software using HSA generic and applicable to any scientific field. It was applied on a renewable energy (RE) system's management. The developed model simulates the hydraulic characteristics of a small-scale hydropower (SHP) station. Harmony search algorithm (HSA) toolkit optimized the SHP's operation, without violating the ecological constraints related to environmental flow (EF) regimes. Milan Cisty and Veronika Soldanova (2017) developed a new methodology in which ensemble modeling by data-driven models was applied and in which harmony search was used to optimize the ensemble structure. In this way, authors combine HSA and ANN in order to predict inflows into the Daecheong Dam in Korea and minimize flooding of landscapes and urban areas which cause immense damage to infrastructures and human lives.

An important issue in managing multi-purpose reservoirs is the non-convexity of the potential policy space and the high algorithmic complexity (NP- hard). This is further amplified when population-based heuristic methods are to be used for large scale multi-reservoir real-time operation problems. Because the large number of variables increases the complexity in a non-linear way and thus requires a large computational time. For this purpose, Mohammad Hadi Afshar et al. (2017) propose a method namely Cellular Automata for efficient solution of multi-reservoir hydropower operation problems. The HS method is embedded into a CA framework in which the CA is used to breakdown the large-scale reservoir system operation into a series of small-scale sub-problem with a size equal to the number of reservoirs in the system. HS method is then used to solve each sub-problem and the results are passed to the CA method. To evaluate the use of Harmony method in multipurpose reservoirs Mohamed Shams et al, (2020) used HSA in reservoir engineering-assisted history-matching of Kareem reservoir in Amal field in the Gulf of Suez. Authors compared the results from HSA, genetic and particle swarm optimization algorithms and proved the superiority and validity of HSO. According to the authors, the reasons

why HSO works better in reservoir engineering-assisted history-matching questions than other algorithms are:

- The good balance between exploration and exploitation during searching for optimal solutions
- In HSO algorithm, the diversity of generated solutions is more efficiently controlled by two subcomponents (pitch adjustment and randomization) compared with other optimization algorithms
- The coordination between the three components (retention of harmony memory, pitch adjustment, and randomization) of the HSO enables to find unbiased solutions.
- HSO algorithm is much easier in implementation process than other optimization algorithms. This is because the HSO is less sensitive to the optimization parameters.

2.7 Conceptual Model

Groundwater availability is in many areas at a worrying state. The exploitation of groundwater is facing critical challenges because natural groundwater recharge is a time-consuming process that does not offset the growing rate of groundwater demand. This problem will increase with population growth.

The modern trend of optimization is the combination of individual models into a conceptual system. This methodology has three main advantages: a) It describes the physical problem much better without focusing on individual elements. b) Enables ecological and social parameters to be included in the decision model and c) Allows the model to be optimized in real time. A conceptual model for optimizing reservoir energy production and maximum ecological benefits (Managed Aquifer Recharghe) is presented in Figure 2.10.

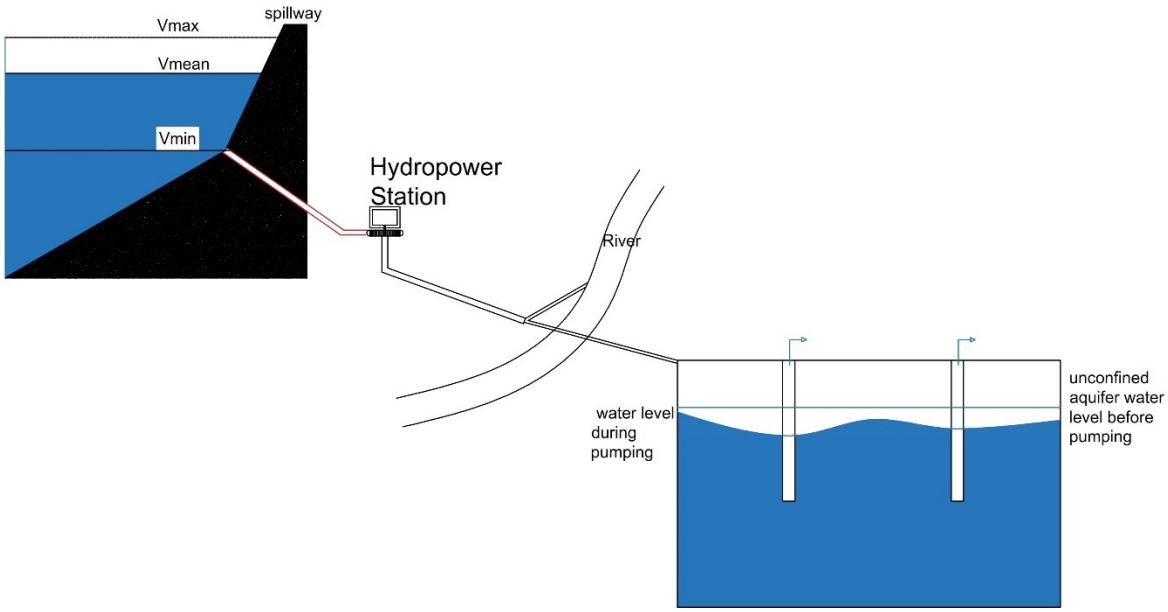

Figure 2.10 Conceptual Model.

2.8 Source Code

Source code was developed in python importing objects form numpy, matplotlib and pandas libraries. The code is presented below:

```
# Import Python libraries
import random
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from random import   seed
from random import randrange
from random import randint
from _datetime import datetime
```

```
# insert flow data from csv file
Qin = np.loadtxt("data2.csv", dtype=float)
```

```
# Water for ecolocial flow subject to constrains
qec = np.array([random.uniform(86400, 259200) for i in range(364)])
```

```
# Dam characteristics
vmax = 1500000
v_initial = 0.75*vmax
A = 50
B = 100
L = 1000
dh_initial = 3 * v_initial / (L * (A + B))
Vin = np.cumsum(Qin) + v_initial - qirr
n = 0.75
vmin = 200000

# ----------------------------------------------------------------------------------------------
# time step
time = [x for x in range(364 + 1) if x > 0]

seed(datetime.now())
#x1 = np.array([randrange(1, 500000, 1) for i in range(364)])
```

```
# Objective Function
def objective_function(Vin,x1,L,A,B,n) :
    Vcur = Vin-x1
    dh = np.array((3 * Vcur) / (L * (A + B)))
    E = (1000 * 9.81 * n * (x1 / 86400) * dh) / 1000000
    return sum(E)
```

```
#print(objective_function(Vin,x1,L,A,B,n))
```

```
# Harmony Search Algorith
# Harmony Memory
hms = 10
hm = np.array([])
for i in range(hms):
    hm=np.append(hm,np.array([random.uniform(10000.00, 600000.00) for i in range(364)]))
hm=np.reshape(hm,(364,hms))
row = []
for i in range(hms):
    row.append(objective_function(Vin,hm[:,i],L,A,B,n))
hm=np.vstack([hm,row])
# sorting harmony memory
hmsort = hm[:, hm[-1, :].argsort()]
#New Harmony
hmcr= 0.7
par = 0.5
#--------------------------------------------------------------------------------
for k in range(10000):
    NewHarmony=np.array([])
    if random.random() < hmcr:

        for i in range(0,364):
            a=randint(0, hms-1)
            NewHarmony=np.append(NewHarmony,hm[i,a])

        if random.random() < par:
            NewHarmony = NewHarmony+random.uniform(-100000.00, 100000.000)

    else:
        NewHarmony = np.array([randrange(1, 500000, 1) for i in range(364)])
    NewHarmony2=np.append(NewHarmony,(objective_function(Vin,NewHarmony,L,A,B,n)))
    if objective_function(Vin,NewHarmony,L,A,B,n) > hmsort[364,0]:
        hmsort[:,0]=NewHarmony2
#sort again
    hmsort = hmsort[:, hmsort[-1, :].argsort()]
    print(k)
#print(hmsort)
vfinal=Vin-hmsort[0:364,-1]
print(vfinal.size)
print(vfinal.shape)
fig, (ax1, ax2) = plt.subplots(2, 1)
fig.subplots_adjust(hspace=0.5)
ax1.plot(time, Qin, linewidth=1.0,label = "Input")
ax2.plot(time,hmsort[0:364,-1] , linewidth=1.0,label = "Hydropower")
ax2.plot(time,qirr , linewidth=1.0,label = "Ecological")
ax1.set(title=r'Reservoir incoming flow per day',
       xlabel='time in days', ylabel=r'Flow $(\frac{m^3}{day})$')
ax2.set(title=r'Reservoir outcoming flow per day',
       xlabel='time in days', ylabel=r'Flow $(\frac{m^3}{day})$')
ax2.set_xlabel('time in days')
ax2.plot(time,vfinal , linewidth=1.0,label = "current reservoir volume")
ax2.axhline(y=vmin,linewidth=1, color='r',label = "Minimum water level")
leg = plt.legend(loc='upper center')
ax1.set_xlim(0, 365)
ax2.set_xlim(0, 365)
ax1.grid(True)
ax2.grid(True)


plt.show()
```

In order for the code to be used object-oriented, it is converted to the following:

```
import random
import numpy as np
from datetime import datetime

class HarmonySearch:
    def __init__(self, data_file, qirr, v_initial, A, B, L, n, vmin, vmax):
        self.Qin = np.loadtxt(data_file, dtype=float)
        self.qirr = qirr
        self.v_initial = v_initial
        self.A = A
        self.B = B
        self.L = L
        self.dh_initial = 3 * v_initial / (L * (A + B))
        self.Vin = np.cumsum(self.Qin) + v_initial - qirr
        self.n = n
        self.vmin = vmin
        self.vmax = vmax
        self.time = [x for x in range(1, 365)]
        self.seed = datetime.now()

    def objective_function(self, x1):
        Vcur = self.Vin - x1
        dh = np.array((3 * Vcur) / (self.L * (self.A + self.B)))
        E = (1000 * 9.81 * self.n * (x1 / 86400) * dh) / 1000000
        return sum(E)

    def initialize_harmony_memory(self, hms):
        hm = np.array([np.array([random.randint(1, 500000) for _ in range(364)]) for _ in range(hms)])
        row = [self.objective_function(hm[:, i]) for i in range(hms)]
        hm = np.vstack([hm, row])
        return hm

    def sort_harmony_memory(self, hm):
        return hm[:, hm[-1, :].argsort()]

    def harmony_search_algorithm(self, hms, hmcr, par, iterations):
        hm = self.initialize_harmony_memory(hms)
        hmsort = self.sort_harmony_memory(hm)

        for k in range(iterations):
            new_harmony = np.array([])

            if random.random() < hmcr:
                for i in range(364):
                    a = random.randint(0, hms - 1)
```

```
                new_harmony = np.append(new_harmony, hmsort[i, a])

            if random.random() < par:
                new_harmony = new_harmony + np.random.uniform(-500000.0, 500000.0, 364)
        else:
            new_harmony = np.array([random.randint(1, 500000) for _ in range(364)])

        new_harmony_eval = self.objective_function(new_harmony)
        if new_harmony_eval > hmsort[364, 0]:
            hmsort[:, 0] = np.append(new_harmony, new_harmony_eval)

        hmsort = self.sort_harmony_memory(hmsort)
        print(k)

    return hmsort

    def plot_hmsort_vs_time(self, hmsort):
        plt.plot(self.time, hmsort[:-1, 0], label='Best Harmony')
        plt.xlabel('Time')
        plt.ylabel('Objective Function Value')
        plt.title('Harmony Search Results')
        plt.legend()
        plt.imshow(img.reshape((28, 28)))
        plt.show()
        plt.savefig

# Example usage:
hs = HarmonySearch("data2.csv", 86400, 1000000, 50, 100, 1000, 0.75, 200000, 1500000)
result = hs.harmony_search_algorithm(hms=3, hmcr=0.7, par=0.5, iterations=1000)

# Plot the results
hs.plot_hmsort_vs_time(result)
```

If we want to insert a more realistic approximation of the supply level curve in the reservoir object in place of Vcur, we insert a list of the current volume and in the place of dh a list of the current height of waterfall. Please note that the prices must refer to an entire year.

3 Software for dam operation

3.1 Overview

The creation of a Graphical User Interface (GUI) using the Tkinter (Lundh, F. 1999, Van Rossum, G. 2020) library in Python marks a pivotal advancement in reservoir management and optimization. With its intuitive design and user-friendly features, the GUI serves as a gateway to seamless interaction and efficient configuration of reservoir parameters. At its core, the GUI is engineered to simplify the input process for reservoir parameters. This includes essential variables such as reservoir capacity, inflow rates, and outflow rates, each of which plays a critical role in shaping reservoir behavior and performance. By providing dedicated input fields users can effortlessly specify these parameters with precision and accuracy. One of the key strengths of the GUI lies in its ability to accommodate a wide range of reservoir configurations and optimization preferences. Whether it's a small-scale reservoir with limited capacity or a large-scale dam with complex inflow patterns, the GUI offers a versatile platform for parameter input, ensuring compatibility with diverse reservoir systems.

To further enhance usability, the GUI incorporates elements that engage users in the optimization process. These features not only streamline the input process but also empower users to explore different optimization scenarios and fine-tune their approach. Furthermore, the GUI provides users with visualization tools to interpret and analyze optimization results. This includes interactive charts and graphs that illustrate key performance metrics such as reservoir storage, hydropower production, current dam volume and ecological flow rate. By visualizing optimization results in an intuitive manner, users can gain valuable insights into reservoir behavior and make informed decisions regarding operation and management strategies.

3.2 Python for GUIs –Tkinter lybrary

The Tkinter library is a fundamental toolkit for creating graphical user interfaces (GUIs) in Python. With its simplicity, versatility, and robust features, Tkinter has become the go-to choose for developers looking to design intuitive and interactive interfaces for their Python applications. In this section, we'll delve into the various aspects of the Tkinter library, exploring its history, functionality, key features, and

benefits. Tkinter provides developers with a wide range of widgets (GUI components) and layout managers to design visually appealing and user-friendly interfaces. Some of the core widgets provided by Tkinter include buttons, labels, entry fields, text areas, checkbuttons, radio buttons, menus, and canvas. These widgets can be arranged and configured using various layout managers, such as pack, grid, and place, to create complex and responsive interfaces. Tkinter's intuitive syntax and straightforward API make it accessible to both beginner and experienced developers. With just a few lines of code, developers can create functional GUI applications with Tkinter. Furthermore, is platform-independent, meaning that GUI applications developed with Tkinter can run on different operating systems, including Windows, macOS, and Linux, without modification. Tkinter allows developers to customize the appearance and behavior of GUI components using various configuration options and styling techniques.

Developers can specify attributes such as colors, fonts, sizes, and alignment to tailor the interface to their preferences. Tkinter follows an event-driven programming paradigm, where user interactions (e.g., button clicks, mouse movements) trigger events that are handled by event handlers (callbacks). This allows developers to create responsive and interactive applications that respond to user input in real-time. Also, Tkinter seamlessly integrates with Python, allowing developers to leverage the full power of the Python programming language to build GUI applications. Developers can use Python's extensive standard library and third-party packages in conjunction with Tkinter to extend the functionality of their applications. For all these reasons, after careful consideration and evaluation, we made the deliberate decision to integrate this specific library into our codebase in order to introduce a graphical environment. Our selection process involved thorough analysis of various factors, including functionality, performance, ease of use, and compatibility with our existing systems. Upon reviewing the available options, it became evident that this particular library excelled in meeting our requirements and offered a robust set of features tailored to our needs. Its comprehensive documentation and active community support further bolstered our confidence in its suitability for our project. Moreover, the library's intuitive API and flexible customization capabilities align seamlessly with our development objectives, allowing us to efficiently implement graphical elements within our codebase while maintaining a high level of control and scalability.

3.3 Source Code of the GUI

This software is a Python script that implements a graphical user interface (GUI) using the Tkinter library for a tool called "GREcoDAM". The main point of the Python code is: The script imports necessary libraries such as random, numpy, matplotlib.pyplot, pandas, and modules from tkinter for creating the GUI.

- An objective function named objective_function, which calculates energy-related values based on input parameters.
- The script defines two functions, on_click1 and on_click, which are likely event handlers for button clicks in the GUI.
- The on_click1 function opens a file dialog to select a csv file for the flow rate data.
- The on_click function seems to perform a complex algorithm related to the "Harmony Search Algorithm" using the parameters provided through the GUI. This algorithm optimizes some values based on the input data and parameters.
- The GUI is set up using Tkinter, with entry fields for various parameters and buttons to trigger the algorithm and file selection.
- Upon executing the on_click function, the algorithm runs, and the results are plotted using Matplotlib in a graphical representation with two subplots.

The final source code is the following:

```
import random
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

from random import   seed
from random import randrange
from random import randint
from _datetime import datetime
from tkinter import *

def objective_function(Vin,x1,L,A,B,n) :
    Vcur = Vin-x1
    dh = np.array((3 * Vcur) / (L * (A + B)))
    E = (1000 * 9.81 * n * (x1 / 86400) * dh) / 1000000
```

```
    return sum(E)
def on_click1():
    root.filename = filedialog.askopenfilename()
    return root.filename




def on_click():
    # ____________________________________________________________
    print(root.filename)
    Qin = np.loadtxt(root.filename, dtype=float)
    qirr = np.array([random.uniform(86400, 259200) for i in
range(364)])
    vmax = float(entry1.get())
    v_initial = float(entry3.get())
    A = float(entry4.get())
    B = float(entry5.get())
    L = float(entry6.get())
    dh_initial = 3 * v_initial / (L * (A + B))
    Vin = np.cumsum(Qin) + v_initial - qirr
    n = 0.75
    vmin = float(entry2.get())
    # ------------------------------------------------------------------
-------------------------------------
    time = [x for x in range(364 + 1) if x > 0]
    seed(datetime.now())
    print(v_initial)
    # Harmony Search Algorith
    # Harmony Memory
    hms = int(entry7.get())
    hm = np.array([])
    for i in range(hms):
        hm = np.append(hm, np.array([random.uniform(10000.00,
600000.00) for i in range(364)]))
    hm = np.reshape(hm, (364, hms))
    row = []
    for i in range(hms):
        row.append(objective_function(Vin, hm[:, i], L, A, B, n))
    hm = np.vstack([hm, row])
    # sorting harmony memory
```

```
    hmsort = hm[:, hm[-1, :].argsort()]
    # New Harmony
    hmcr = float(entry8.get())
    par = float(entry9.get())
    # ----------------------------------------------------------------
------------------
    for k in range(10):
        NewHarmony = np.array([])
        if random.random() < hmcr:

            for i in range(0, 364):
                a = randint(0, hms - 1)
                NewHarmony = np.append(NewHarmony, hm[i, a])

            if random.random() < par:
                NewHarmony = NewHarmony + random.uniform(-100000.00,
100000.000)

        else:
            NewHarmony = np.array([randrange(1, 500000, 1) for i in
range(364)])
        NewHarmony2 = np.append(NewHarmony, (objective_function(Vin,
NewHarmony, L, A, B, n)))
        if objective_function(Vin, NewHarmony, L, A, B, n) >
hmsort[364, 0]:
            hmsort[:, 0] = NewHarmony2
        # sort again
        hmsort = hmsort[:, hmsort[-1, :].argsort()]
        print(k)
    # print(hmsort)
    vfinal = Vin - hmsort[0:364, -1]
    print(vfinal.size)
    print(vfinal.shape)
    fig, (ax1, ax2) = plt.subplots(2, 1)
    fig.subplots_adjust(hspace=0.5)
    ax1.plot(time, Qin, linewidth=1.0, label="Input")
    ax2.plot(time,       hmsort[0:364,       -1],       linewidth=1.0,
label="Hydropower")
    ax2.plot(time, qirr, linewidth=1.0, label="Irrigation")
    ax1.set(title=r'Reservoir incoming flow per day',
```

```
            xlabel='time in days', ylabel=r'Flow
$(\frac{m^3}{day})$')
    ax2.set(title=r'Reservoir outcoming flow per day',
            xlabel='time in days', ylabel=r'Flow
$(\frac{m^3}{day})$')
    ax2.set_xlabel('time in days')
    ax2.plot(time, vfinal, linewidth=1.0, label="current reservoir
volume")
    ax2.axhline(y=vmin, linewidth=1, color='r', label="Minimum water
level")
    leg = plt.legend(loc='upper center')
    ax1.set_xlim(0, 365)
    ax2.set_xlim(0, 365)
    ax1.grid(True)
    ax2.grid(True)

    plt.show()


root = Tk()
root.title("GREcoDAM")
root.iconbitmap("./logo.ico")
root.geometry('800x600')
frame1 =  LabelFrame(root, text="Insert Dam Parameters")
frame1.grid(row=0,column=0,padx=10,pady=10)
label1 = Label(frame1,text="Maximum Volume (m^3)")
entry1 =  Entry(frame1)
entry1.insert(0, "1500000")
label2 = Label(frame1,text="Minimum Volume (m^3)")
entry2 =  Entry(frame1)
entry2.insert(0, "200000")
label3 = Label(frame1,text="Initial Volume (m^3)")
entry3 =  Entry(frame1)
entry3.insert(0, "1125000")
label1.pack()
entry1 .pack()
label2.pack()
entry2 .pack()
label3.pack()
entry3 .pack()
frame2 =  LabelFrame(root, text="Dam Geometry")
```

```
frame2.grid(row=20,column=0,padx=10,pady=10)
label4 = Label(frame2,text="A(m)")
entry4 =  Entry(frame2)
entry4.insert(0, "50")
label4.pack()
entry4.pack()
label5 = Label(frame2,text="B(m)")
entry5 =  Entry(frame2)
entry5.insert(0, "100")
label5.pack()
entry5.pack()
label6 = Label(frame2,text="L(m)")
entry6 =  Entry(frame2)
entry6.insert(0, "1000")
label6.pack()
entry6.pack()
frame3 =  LabelFrame(root, text="Harmony Search Algorith")
frame3.grid(row=40,column=0,padx=10,pady=10)
label7 = Label(frame3,text="HMS")
entry7 = Entry(frame3)
entry7.insert(0, "10")
label8 = Label(frame3,text="HMCR")
entry8 = Entry(frame3)
entry8.insert(0, "0.7")
label9 = Label(frame3,text="PAR")
entry9 = Entry(frame3)
entry9.insert(0, "0.5")
label7.pack()
entry7.pack()
label8.pack()
entry8.pack()
label9.pack()
entry9.pack()
btn1=Button(root,text="Solve",command=on_click)
btn1.grid(row=80,column=0)
btn2=Button(root,text="CSV Data",command=on_click1)
btn2.grid(row=60,column=0)



root.mainloop()
```

In addition to running on Windows environment, this code can also be executed across various operating systems, including Linux distributions, macOS, and Microsoft Windows. Python's cross-platform compatibility ensures that the same codebase can be utilized across different operating systems without modification.

3.4 Graphical User Interface – Software

Utilizing source code can present challenges for individuals unfamiliar with Python. Hence, a graphical user interface (GUI) has been developed to simplify data input and result export processes. For the GUI implementation, the widely adopted tkinter package, an open-source Python library, has been employed. Tkinter stands as Python's standard GUI toolkit, offering a straightforward means of constructing GUI applications swiftly. Combining Python with Tkinter furnishes developers with a speedy and efficient approach to GUI development. Tkinter furnishes a robust object-oriented interface to the Tk GUI toolkit. Moreover, Tkinter comes pre-installed with standard Python installations on GNU/Linux, Microsoft Windows, and macOS platforms. The primary interface of the software is depicted in Figure 3.1.

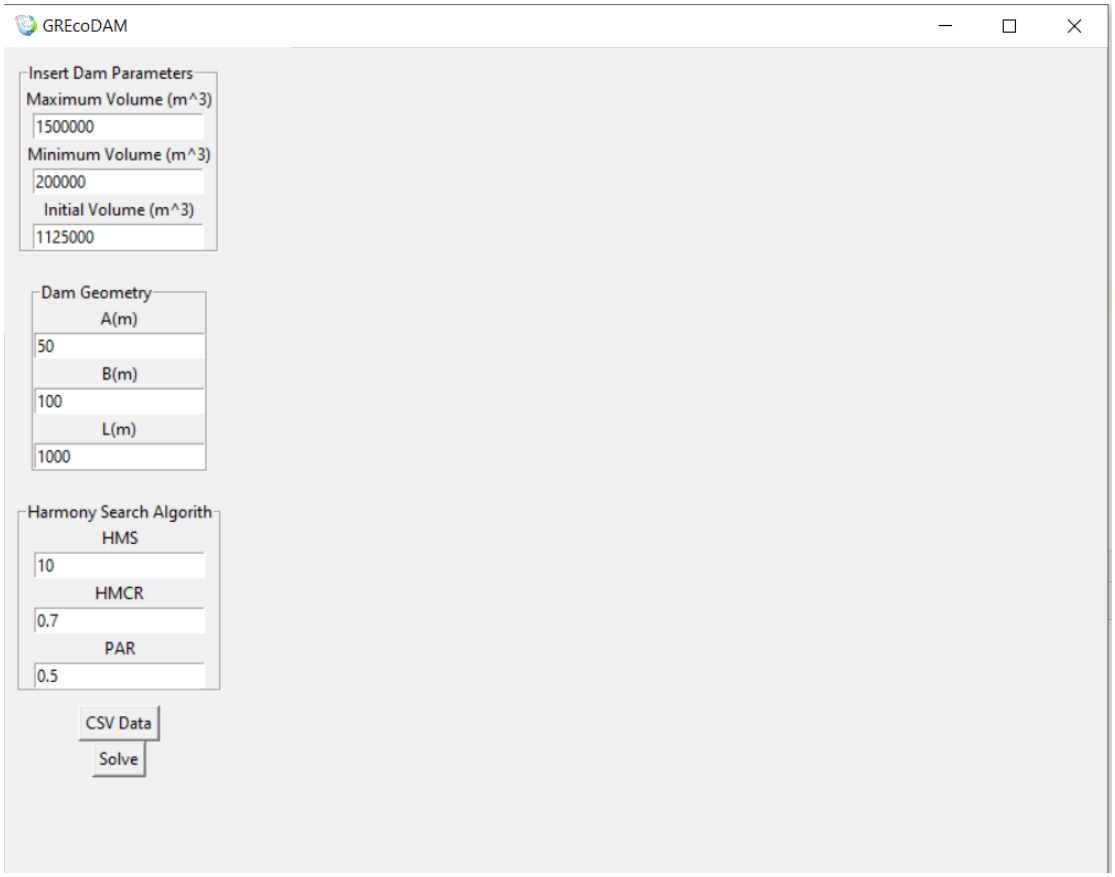

Figure 3.1 Software main form.

The user must complete three data classes, where each class is in a separate framebox. At the first framebox the user fills in the textboxes with the reservoir capacity sizes. User must enter three parameters: the maximum volume of water in the

reservoir, the minimum and the initial. The explanation of these three parameters is given in the following Table in relation to the Figure 3.1.

Table 3.1 Parameters explanation.

Parameter	*Units*	*Details*
Maximum Volume	m^3	This parameter describes the maximum volume of water in the reservoir and is defined by two surfaces. The lower surface is the bottom and the upper the location of spillway.
Minimum Volume	m^3	This parameter describes the minimum volume of water in the dam and is defined by the bottom of the reservoir and the location of the turbine pipe.
Initial Volume	m^3	Initial volume is defined as the volume of water in the dam at the beginning of the simulation. The first daily water supply value is set as the start day of the simulation.

In the next framebox the user fills in the geometric features of the dam as shown in Figure 3.1. In the last framebox the values from the parameters of the harmony search method are filled in, as such are presented in a previous report (D3-1). To get acquainted with the software and avoid executable errors, some default

values exist at textboxes. The user then presses the "CSV Data" button and enters a CSV file with daily water supplies for a period of one year. Finally, by pressing the "Solve" button, the program is executed and the diagram with the results is displayed. The final form is shown in Figure 3.2.

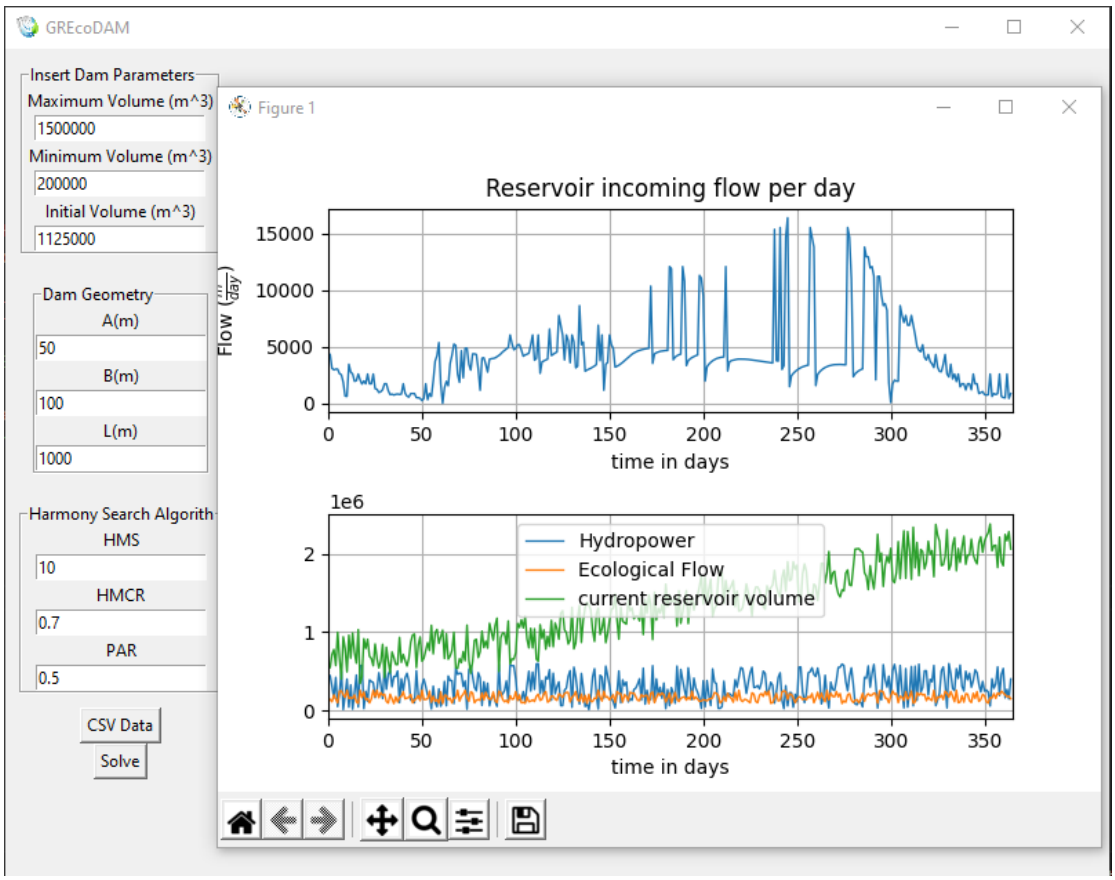

Figure 3.2 Software final form.

3.5 Cost scenarios for pumping from an underground aquifer

3.5.1 Introduction

This technical data sheet examines various cases of total cost of pumping from an underground aquifer to a water supply tank. In this case, a well is considered a well within a homogeneous and isotropic and infinite aquifer, of constant permeability. The general arrangement from which the variations of this issue arise is presented in Figure 3.3.

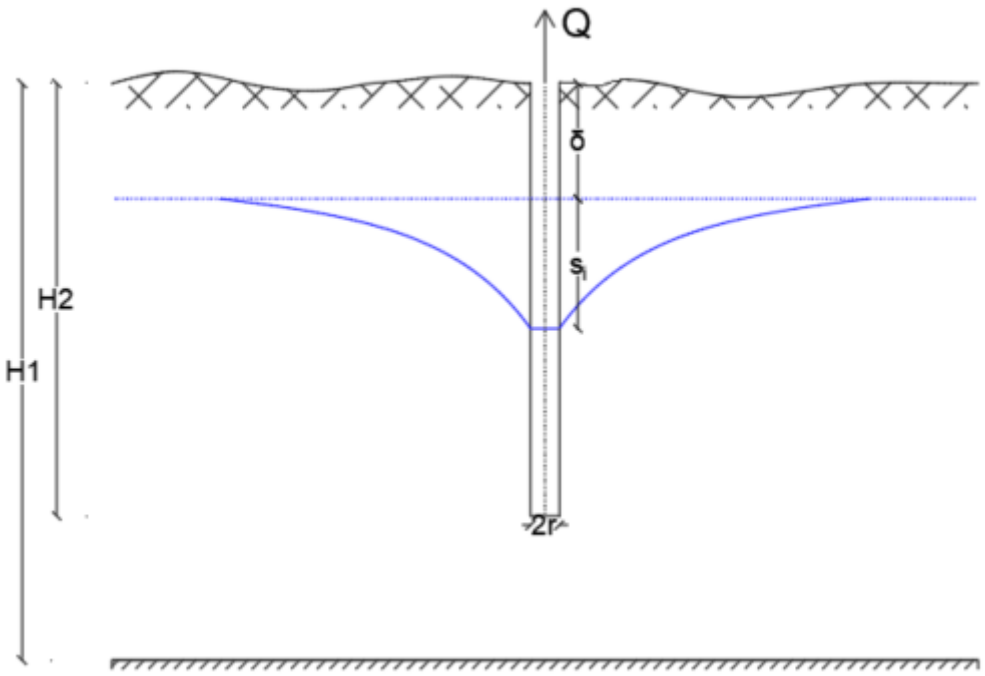

Figure 3.3 General arrangement.

The pumping cost (Katsifarakis 2008) is given by equation 1.1:

$$K = P \cdot T_p \cdot C_{kWh} \qquad 1.1$$

Where:

- P: the required pumping power in kW
- T: the duration of pumping in h
- CkWh: The current price of kWh

The required pump value P is given by equation 1.2:

$$P = \frac{\rho \cdot g}{n_p} \cdot \sum_{i}^{NW} \frac{Q_i \cdot (s_i + \delta)}{10^3} \qquad 1.2$$

Where:

- ρ: the density of the pumped fluid in Kg/m3
- g: the acceleration of gravity in m/s2
- np: The efficiency rating of each pump
- Qi: The pumped flow in L/s
- si: The water level drop at the side of the borehole in m
- d: The distance of the resting level from the ground surface in m

Combining Equation 1.1. with 1.2 we get Equation 1.3:

$$K = T_p \cdot C_{kWh} \cdot \frac{\rho \cdot g}{n_p} \cdot \sum_{i}^{NW} \frac{Q_i \cdot (s_i + \delta)}{10^3} \qquad 1.3$$

This cost concerns the necessary pressure gauge to pump the water up to the ground surface. In the case where additional energy is required, then this should be added to equation 1.3. For example, if the borehole feeds a depression pipeline to the reservoir of a water supply network, as shown in Figure 3.4.

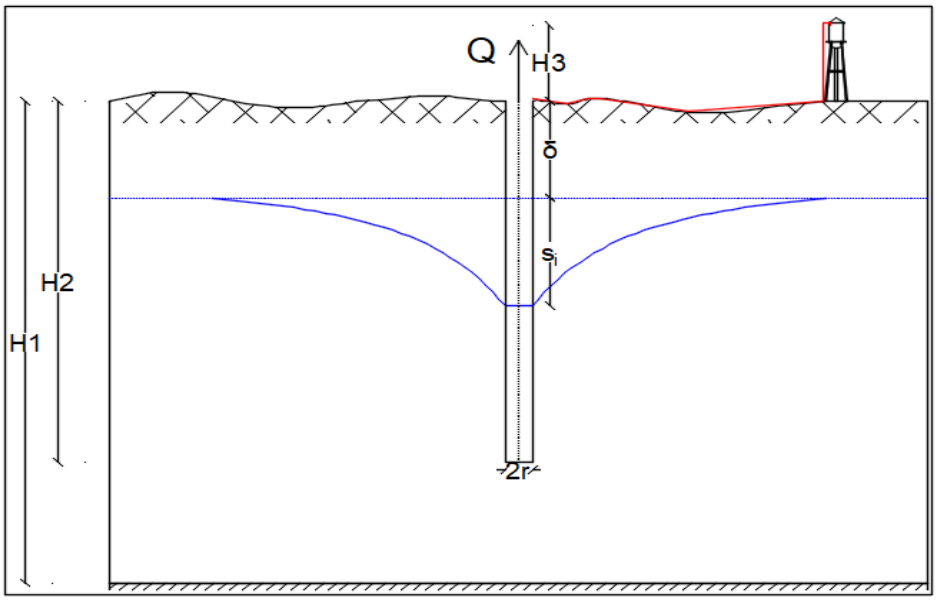

Figure 3.4 Example of a borehole that feeds a depression pipeline to the reservoir of a water supply network.

The total extra energy in this case consists of the sum of the height difference H3, the linear and local losses of the depression pipe. If the local losses are ignored and the Darcy Weisbach (PipeFlow. 2023) model is considered then Equation 1.3 transforms into Equation 1.4.

$$K = T_p \cdot C_{kWh} \cdot \frac{\rho \cdot g}{n_p} \cdot \sum_{i}^{NW} \frac{Q_i \cdot (s_i + \delta + H_3 + \Delta H)}{10^3} \quad 1.4$$

where ΔH , circular conductor is given by Equation 1.5.

$$\Delta H = \frac{8 \cdot f \cdot L \cdot Q^2 \cdot \rho}{2\,\pi \cdot D^5} \quad 1.5$$

Where:

- f stands for the coefficient of friction given by equation 1.6
- L stands for the length of the pipeline in m

$$f = 0.0055 \cdot \left(1 + \left(20000 \cdot \frac{K}{D} + \frac{10^6}{Re}\right)^{\frac{1}{3}}\right) \quad 1.6$$

Where:

- K stands for the roughness coefficient of the conductor
- Re stands for the Reynolds number of the flow

The drop in level due to pumping is usually calculated by numerical models such as MODFLOW (Harbaugh, A.W., 2005). In the simplified case of a well it can be calculated from the model of Theis (1935), according to which the drop in level is given by Equation 1.7

$$s(r,t) = \frac{Q}{4\pi T} W(u) \quad 1.7$$

where the parameter u is given by equation 1.8

$$u = \frac{r^2 S}{4Tt} \quad 1.8$$

where:

- r: the distance from the axis of the well
- S: the storage capacity of the aquifer
- T: the transportability of the aquifer
- t: the time since the start of pumping

W(u) is a function, which Theis called the well function, and is given by Equation 1.9

$$W(u) = \int_0^{\infty} \frac{e^{-x}}{x} dx \quad 1.9$$

For u<0.01 and radius of influence R the level drop for a free aquifer is given by Equation 1.10

$$s \cdot (1 - \frac{s}{2 \cdot h_0}) = \frac{Q}{2\pi \mathrm{K} h_0} ln \frac{R}{r} \quad 1.10$$

Where:

- K: the permeability of the aquifer
- h0: the thickness of the saturated zone

The radius of influence can be given by equation 1.11 (Kirieleies – Sichardt 1930)

$$R = 3000 \cdot s \cdot \sqrt{K} \quad 1.11$$

Combining Equation 1.10 with 1.11 yields Equation 1.12

$$s \cdot (1 - \frac{s}{2 \cdot h_0}) = \frac{Q}{2 \cdot \pi \cdot \mathrm{K} \cdot h_0} ln \frac{3000 \cdot s \cdot \sqrt{K}}{r} \quad 1.12$$

3.5.2 Calculation of Pumping Costs

In this paragraph, the pumping cost will be calculated for various cases of aquifers and hydraulic conditions. The data of each case is presented in Table 2. As the radius of the wells r= 0.1m is taken, as the pumping time is taken a period of one year and as the value of the kilowatt hour CkWh =0.087 €/kWh. A trial procedure is organized to solve Equation 1.12 in terms of s. Regarding the efficiency factor, manufacturers usually provide three curves of the manometric H, the power P and the degree of efficiency np in function of the delivery Q of the pump. In any case we can calculate the pump efficiency using Equation 1.13

$$n_P = \frac{\rho \cdot g \cdot Q \cdot H_{tot}}{P} \quad 1.13$$

- P: is the power transmitted to the pump shaft by the motor (for centrifugal pumps). For the sake of simplicity, a constant degree of efficiency is considered, as well as the operating point for each combination of Q and H that belongs to the point where the efficiency factor is maximized (Best Efficiency Point).

Case	H1 (m)	d (m)	Q (l/s)	K (m/s)	n.p
Case 1	100	5	5	10-5	0.8
Case 2	70	10	20	5*10-5	0.85
Case 3	55	15	15	10-3	0.85
Case 4	50	10	30	10-2	0.8
Case 5	40	5	10	10-4	0.9

In each of the above cases, the four parameters are kept constant and the fifth is changed. Using Equation 1.12 and 1.4 the change in hourly cost is calculated. The results are shown in following Figures.

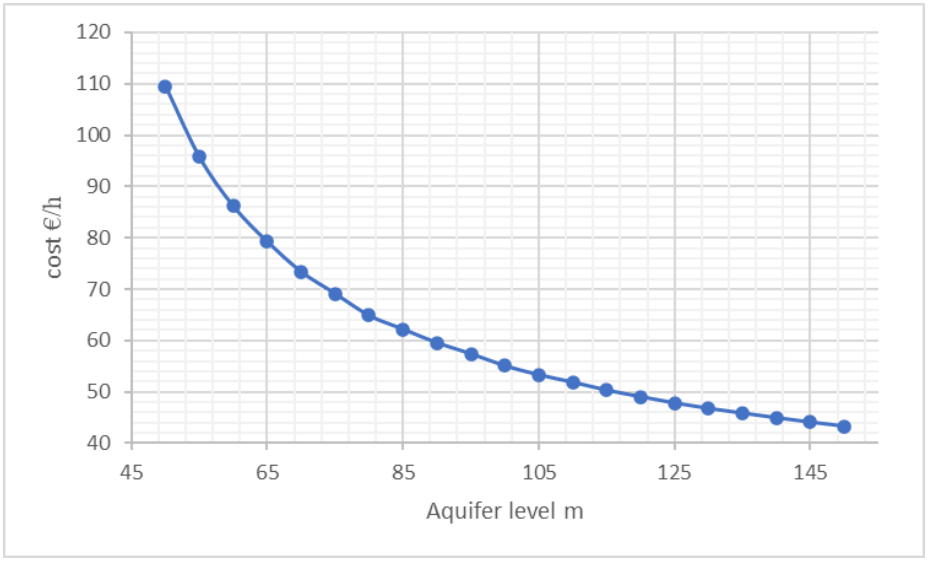

Figure 3.5 Variation of cost with aquifer depth.

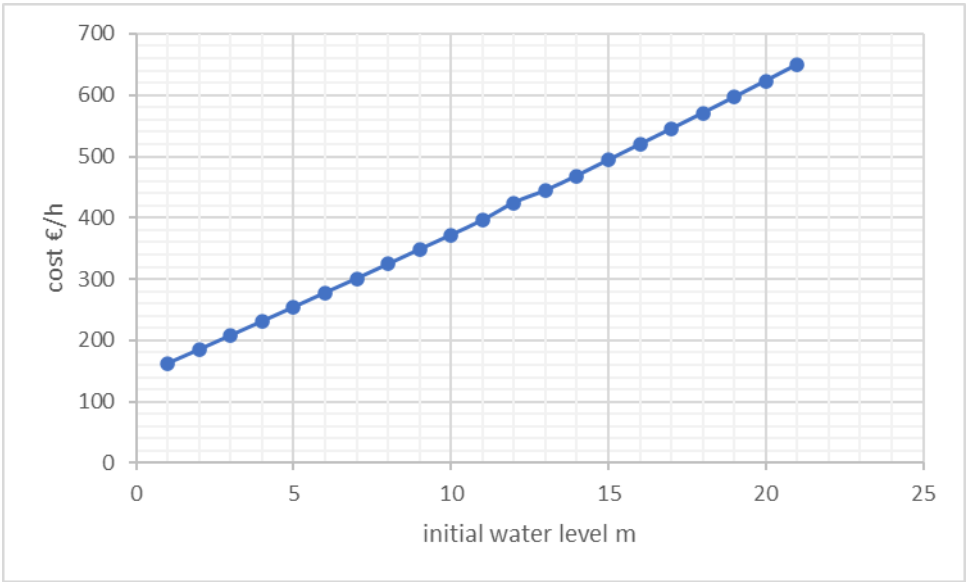

Figure 3.6 Variation of cost with the initial depth of the free level of the aquifer.

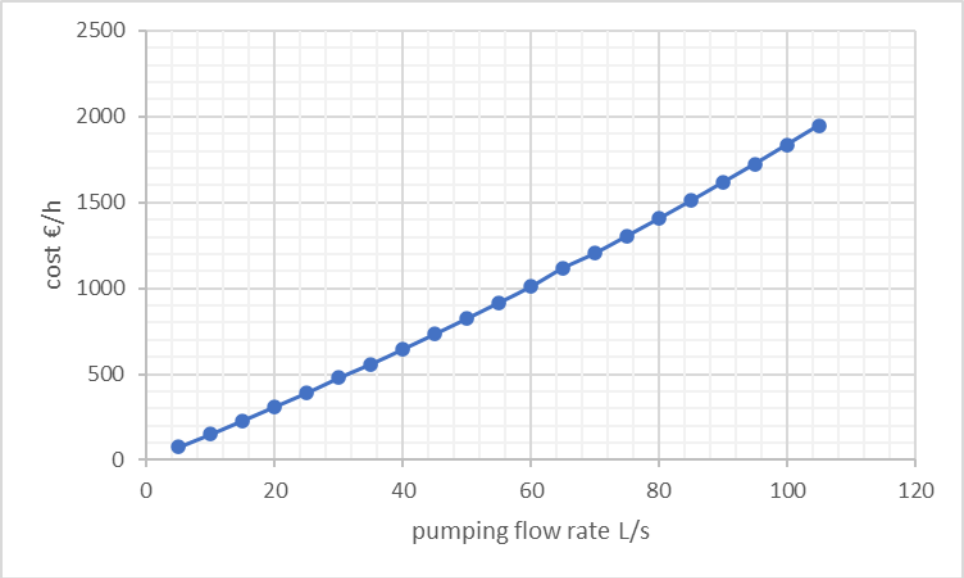

Figure 3.7 Variation of cost with well pumping supply.

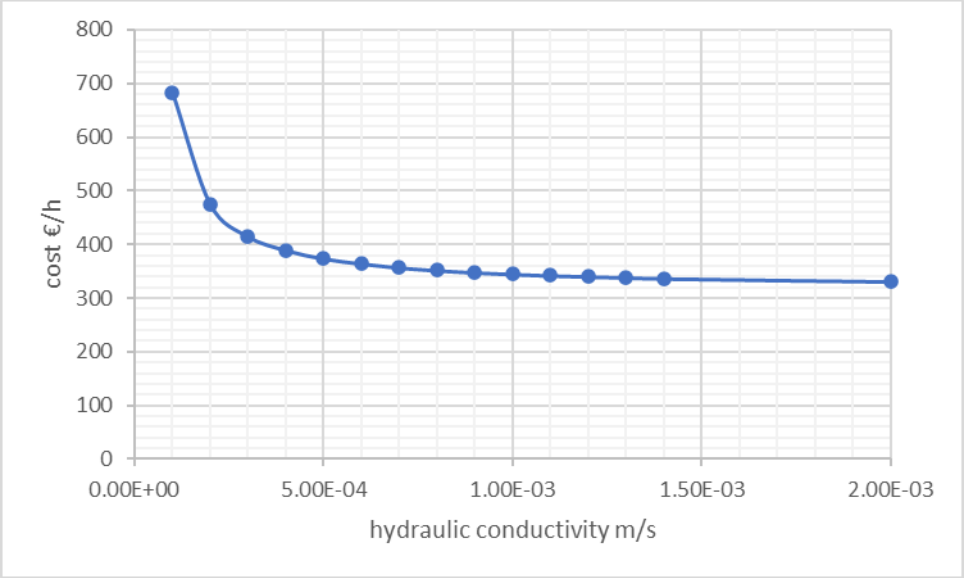

Figure 3.8 Variation of cost with soil permeability.

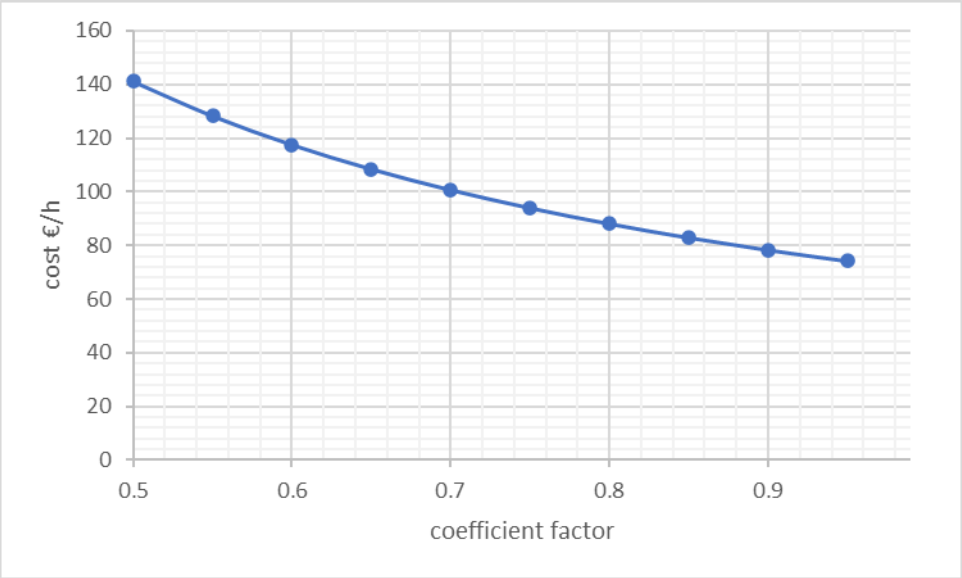

Figure 3.9 Variation of cost with the coefficient of performance.

3.6 DAM-MAR Model

In this paragraph, we will request the formulation of a unified optimization model that encompasses both the optimization of power generation from reservoir hydroelectric plants and the replenishment of groundwater resources. This integrated model aims to harmonize the management of these interconnected systems, considering factors such as energy production, water availability, environmental sustainability, and economic efficiency. By optimizing both hydroelectric power generation and groundwater recharge simultaneously, the model seeks to achieve a balanced and sustainable utilization of water resources, addressing the multifaceted challenges posed by water-energy nexus management. The conceptual model that describes the optimization process is presented in Figure 3.10.

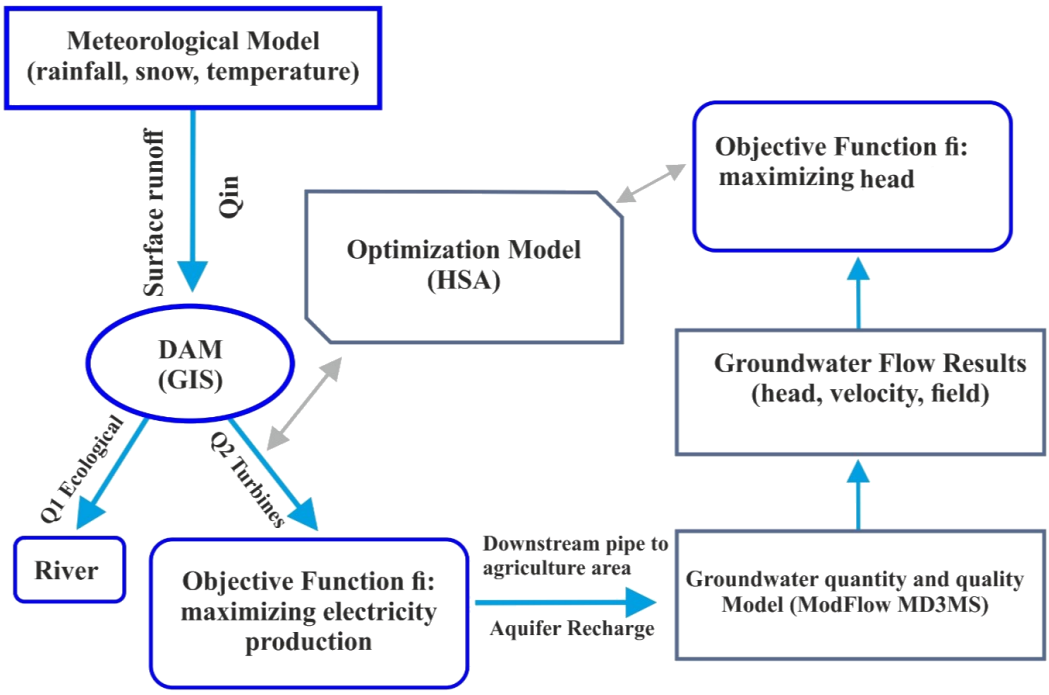

Figure 3.10 Conceptual model of optimization problem.

The new objective equations of the optimization model are shown in 1.14,1.15,1.16

$$F_1 = \sum_{0}^{365} \rho * g * Dh * Q_{turbine} \qquad 1.14$$

$$F2 = S_y \frac{\partial h}{\partial t} \qquad 1.15$$

$$F = \sum_{0}^{365} (F1 + F2) \qquad 1.16$$

The conceptual groundwater model was simulated for the general aquifer case using MODFLOW NT software through Python language and the Flopy library. This approach enables iterative execution of MODFLOW, thereby incorporating the second objective function— the drop in water level—into the optimization problem. Establishing the groundwater flow model required defining classic parameters such as aquifer depth, as well as length and width dimensions. Cell dimensions of 100x100 meters were chosen for the discretization package. The aquifer boundaries were delineated using the Generalized Boundary Condition (GBH) or Conscatn Head Boundary (CHB), and model calibration was performed using observation well data.

With Flopy, a Python library, one can efficiently construct underground flow models. It's important to note that Flopy doesn't directly solve the subsurface hydraulics equations; instead, it aids in the creation of input files for MODFLOW and facilitates the retrieval and visualization of output files. Similar functionality is found in other software packages like modelMuse, GMS, and others. These tools serve as interfaces for constructing, simulating, and analyzing groundwater flow models. They streamline the process by providing user-friendly interfaces and automated workflows, allowing hydrogeologists and engineers to focus on model design, analysis, and interpretation rather than intricate coding or manual file manipulation. The main steps for develop a flopy model are present in Figure 3.11.

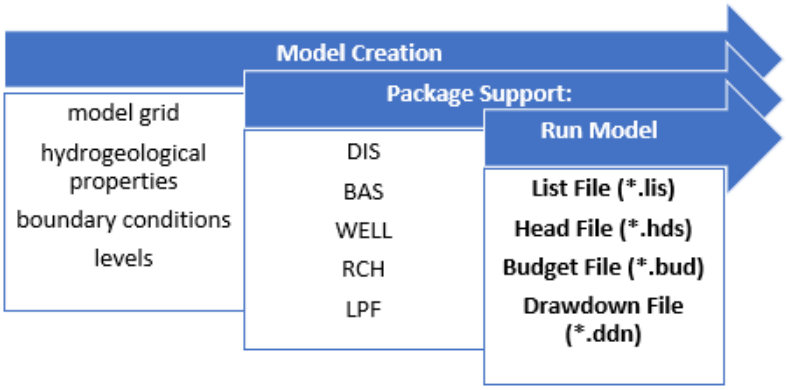

Figure 3.11 FloPy chartflow.

The main files typically used in a MODFLOW –Flopy model are (Bakker et al 2016; Leaf, A. T et al 2022; Lie, B. et al 2015; Hughes, J et al 2023):

1. **Name File (.nam)**:
    - The name file serves as the entry point for MODFLOW. It specifies the names of all input and output files associated with the model simulation, including the MODFLOW executable, list file, and output data files.
2. **Discretization File (.dis)**:
    - This file contains information about the model grid, including the number of layers, rows, and columns, as well as the spatial discretization (grid spacing) along rows and columns. It also specifies the elevations of the top and bottom of each model layer.
3. **Basic Package Files**:
    - **Basic Package (.bas)**: Defines the active cells in the model grid.
    - **Layer Property Flow Package (.lpf)**: Specifies hydraulic properties such as hydraulic conductivity, specific storage, and hydraulic conductivity anisotropy.
    - **Boundary Condition Packages**:
        - **Constant Head Package (.chd)**: Specifies constant head boundary conditions.
        - **Well Package (.wel)**: Defines best-placed locations and pumping rates.
        - **Recharge Package (.rch)**: Specifies recharge rates.
        - **River Package (.riv)**: Defines the river's properties and the boundary conditions.
        - **General-Head Boundary Package (.ghb)**: Specifies general-head boundary conditions.
        - **Drain Package (.drn)**: Defines drain properties and boundary conditions.
4. **Solver Package (.pcg, .de4)**:
    - The Preconditioned Conjugate Gradient (PCG) solver package (.pcg) or another solver package is used to specify solver parameters,

convergence criteria, and other settings related to solving the groundwater flow equation.

5. **Output Control File (.oc)**:
    - The output control file specifies the frequency and format of output data written during the model simulation, including head and flow values at specified locations and times.

Depending on the model complexity and specific requirements, additional packages or files may be necessary. Flopy simplifies the process of creating and managing these files programmatically, allowing users to efficiently set up and run MODFLOW simulations.

The locations of pumping and artificial recharge wells for a given recharge period play a crucial role in groundwater management and hydrogeological studies. Pumping wells are strategically positioned to extract groundwater for various purposes such as irrigation, municipal water supply, and industrial use. On the other hand, artificial recharge wells are strategically located to replenish groundwater reserves through injection of surface water or treated wastewater, especially in areas experiencing groundwater depletion or contamination. The selection of well locations involves careful consideration of several factors, including hydrogeological characteristics, land use patterns, water demand, environmental concerns, and regulatory requirements. Hydrogeological factors such as aquifer properties, hydraulic conductivity, transmissivity, and groundwater flow patterns influence the optimal placement of wells to ensure efficient extraction or recharge operations. Moreover, well locations should be chosen to minimize potential adverse impacts such as groundwater depletion, land subsidence, saltwater intrusion, and contamination risks. Geospatial analysis techniques, groundwater modeling, and decision support systems are often employed to identify suitable well locations based on a comprehensive assessment of these factors. Furthermore, stakeholder engagement and community consultation are essential in the well siting process to address concerns, gather local knowledge, and ensure the acceptance and sustainability of groundwater management initiatives. By carefully selecting the locations of pumping and artificial recharge wells, water resource managers can effectively balance water supply needs, environmental protection, and long-term groundwater sustainability.

The code to create the groundwater flow model and dam model is presented below:

```
import random
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
import flopy
import itertools

from random import   seed
from random import randrange
from random import randint
from _datetime import datetime

def objective_function(Vin,x1,L,A,B,n) :
    Vcur = Vin-x1
    dh = np.array((3 * Vcur) / (L * (A + B)))
    E = (1000 * 9.81 * n * (x1 / 86400) * dh) / 1000000
    return sum(E)

Qin = np.loadtxt("data_Campania.csv", dtype=float)
qirr = np.array([random.uniform(3456, 4320) for i in range(364)])
vmax = 2600000
v_initial = 0.75*vmax
A = 15
B = 100
L = 3000
dh_initial = 3 * v_initial / (L * (A + B))
Vin = np.cumsum(Qin) + v_initial - qirr
n = 0.75
vmin = 200000
hmin=3 * vmin / (L * (A + B))
# ----------------------------------------------------------------------
----------------------------------
time = [x for x in range(364 + 1) if x > 0]
seed(datetime.now())
#x1 = np.array([randrange(1, 500000, 1) for i in range(364)])

#print(objective_function(Vin,x1,L,A,B,n))
```

```
# Harmony Search Algorith
# Harmony Memory
hms = 10
hm = np.array([])
for i in range(hms):
    hm=np.append(hm,np.array([random.uniform(1000.00, 6000.00) for i
in range(364)]))
hm=np.reshape(hm,(364,hms))
row = []
for i in range(hms):
    row.append(objective_function(Vin,hm[:,i],L,A,B,n))
hm=np.vstack([hm,row])
# sorting harmony memory
hmsort = hm[:, hm[-1, :].argsort()]
#New Harmony
hmcr= 0.7
par = 0.5
#-----------------------------------------------------------------------
--------------
for k in range(10):
    NewHarmony=np.array([])
    if random.random() < hmcr:

        for i in range(0,364):
            a=randint(0, hms-1)
            NewHarmony=np.append(NewHarmony,hm[i,a])

        if random.random() < par:
            NewHarmony     =     NewHarmony+random.uniform(-100000.00,
100000.000)

    else:
        NewHarmony  =  np.array([randrange(1,  500000,  1)  for  i  in
range(364)])

NewHarmony2=np.append(NewHarmony,(objective_function(Vin,NewHarmony,L
,A,B,n)))
    if objective_function(Vin,NewHarmony,L,A,B,n) > hmsort[364,0]:
        hmsort[:,0]=NewHarmony2
#sort again
    hmsort = hmsort[:, hmsort[-1, :].argsort()]
```

```
    print(k)
#print(hmsort)
vfinal=Vin-hmsort[0:364,-1]
hcr = 3 * vfinal / (L * (A + B))
Energy = (9.81*hcr*hmsort[0:364,-1]*n*1000)/86400
print(vfinal.size)
print(vfinal.shape)
fig, (ax1, ax2) = plt.subplots(2, 1)
fig.subplots_adjust(hspace=0.5)
ax1.plot(time, Qin, linewidth=1.0,label = "Input")
ax2.plot(time,hmsort[0:364,-1] , linewidth=1.0,label = "Hydropower
Flow")
ax2.plot(time,qirr , linewidth=1.0,label = "Environmental Flow")
ax1.set(title=r'Reservoir incoming flow per day',
       xlabel='time in days', ylabel=r'Flow $(\frac{m^3}{day})$')
ax2.set(title=r'Reservoir outcoming flow per day',
       xlabel='time in days', ylabel=r'Flow $(\frac{m^3}{day})$')
ax2.set_xlabel('time in days')
#ax2.plot(time,vfinal , linewidth=1.0,label = "Current reservoir
volume")
#ax2.axhline(y=vmin,linewidth=1, color='r',label = "Minimum water
level")
#ax2.axhline(y=vmax,linewidth=1, color='r',label = "Maximum water
level")
leg = plt.legend(loc='center right')
ax1.set_xlim(0, 365)
ax2.set_xlim(0, 365)
ax1.grid(True)
ax2.grid(True)

plt.show()
print(hmsort)
file_path = 'Campania_Energy.txt'
np.savetxt(file_path, Energy)
qrecharge = np.mean(hmsort[0:364,-1])
print(qrecharge)
# Assign name and create modflow model object
modelname = 'GrecoDam'
mf = flopy.modflow.Modflow(modelname, exe_name="mf2005")
# Model domain and grid definition
Lx = 5000.
```

```
Ly = 5000.
ztop = 100.
zbot = 0.
nlay = 1
nrow = 100
ncol = 100
delr = Lx / ncol
delc = Ly / nrow
delv = (ztop - zbot) / nlay
x_coord = np.linspace(delr/2, Lx-delr/2, num=ncol)
y_coord = np.linspace(Ly-delc/2, delc/2, num=nrow)
botm = np.linspace(ztop, zbot, nlay + 1)
hk = 1.
vka = 1.
sy = 0.1
ss = 1.e-4
laytyp = 1
# define boundary conditions: 1 everywhere except
# left and right edges, which are -1
ibound = np.ones((nlay, nrow, ncol), dtype=np.int32)
ibound[:,:,(0,ncol-1)] = -1.0
# initial conditions
strt = 80. * np.ones((nlay, nrow, ncol), dtype=np.float32)
# Time step parameters
nper = 2
perlen = [100,100]
nstp = [100,100]
steady = [False,False]
# Flopy objects
dis = flopy.modflow.ModflowDis(mf, nlay, nrow, ncol, delr=delr,
delc=delc,
                               top=ztop, botm=botm[1:],
                               nper=nper, perlen=perlen, nstp=nstp,
steady=steady)
bas = flopy.modflow.ModflowBas(mf, ibound=ibound, strt=strt)
lpf = flopy.modflow.ModflowLpf(mf, hk=hk, vka=vka, sy=sy, ss=ss,
laytyp=laytyp)
pcg = flopy.modflow.ModflowPcg(mf)
# set up pumping well
r_well = round(nrow/2)
c_well = round(ncol/2)
```

```
wel_sp1 = [[0, r_well, c_well, -100],[0, 10, 10, -1000],[0, 20, 40, -150],[0, 30, 50, -1200],[0, 50, 50, -1900],[0, 70, 70, -2000],[0, 80, 80, -1100],[0, 80, 30, -1200],[0, 90, 90, 0],[0, 90, 10, -1800],[0, 85, 20, -1900]]
wel_sp2 = [[0, r_well, c_well, 0],[0, 10, 10, -100],[0, 20, 20, -150],[0, 30, 30, -120],[0, 50, 50, -190],[0, 70, 70, -200],[0, 80, 80, -110],[0, 80, 30, -120],[0, 90, 90, +1000],[0, 90, 10, -180],[0, 85, 20, -190]]
stress_period_data = {0: wel_sp1,
                      1: wel_sp2}
wel                =                  flopy.modflow.ModflowWel(mf, stress_period_data=stress_period_data)
# Add recharge package
recharge_rate = 1e-4  # Adjust as needed
rch = flopy.modflow.ModflowRch(mf, rech=recharge_rate)
# Output control
#oc = flopy.modflow.ModflowOc(mf, save_every=True, compact=True)
oc  =  flopy.modflow.ModflowOc(mf,  save_every=True,  compact=True, stress_period_data={(0,  1):  ['save  head',  'save  drawdown',  'save budget']})
# Write the model input files
mf.write_input()

# Run the model
success,   mfoutput   =   mf.run_model(silent=True,   pause=False, report=True)
if not success:
    raise Exception('MODFLOW did not terminate normally.')
# Imports
import matplotlib.pyplot as plt
import flopy.utils.binaryfile as bf

# Create the headfile object
headobj = bf.HeadFile(modelname+'.hds', text='head')

# get data
time = headobj.get_times()[0]
head = headobj.get_data(totim=time)
extent = (x_coord[0],x_coord[ncol-1],y_coord[0],y_coord[nrow-1])

# Well point
```

```
wpt = (float(round(ncol/2)+0.5)*delr, float(round(nrow/2)+0.5)*delc)
# plot of head
plt.subplot(2,1,1)
plt.imshow(head[0,:,:], extent=extent, cmap='BrBG')
plt.colorbar()
plt.plot(wpt[0], wpt[1], lw=0, marker='o', markersize=8,
             markeredgewidth=0.5,
             markeredgecolor='black',
             markerfacecolor='none',
             zorder=9)
# cross-section (L-R) of head through the well
plt.subplot(2,1,2)
plt.plot(x_coord, head[0,r_well,:])
plt.show()
# timeseries
idx = (0, r_well, c_well)
ts = headobj.get_ts(idx)
plt.subplot(1, 1, 1)
ttl = 'Head at cell ({0},{1},{2})'.format(idx[0] + 1, idx[1] + 1,
idx[2] + 1)
plt.title(ttl)
plt.xlabel('time')
plt.ylabel('head')
plt.plot(ts[:, 0], ts[:, 1])
plt.show()
import flopy.utils.postprocessing as post
# Period 1 (end of 10 years)
heads_period1 = headobj.get_data(kstpkper=(0, 0, 9))
water_table_period1 = post.get_water_table(heads_period1[0], nodata=-
999.99)  # Use the correct format

# Period 2 (end of 20 years)
heads_period2 = headobj.get_data(kstpkper=(0, 0, 19))
water_table_period2 = post.get_water_table(heads_period2[0], nodata=-
999.99)  # Use the correct format
# Contour plot for the water table at the end of the first period
plt.contour(water_table_period1, cmap='viridis', extent=(0, ncol *
delc, 0, nrow * delr))
plt.title('Water Table at the end of the first period (10 years)')
plt.colorbar(label='Water Table Elevation (meters)')
plt.show()
```

```
# Contour plot for the water table at the end of the second period
plt.contour(water_table_period2, cmap='viridis', extent=(0, ncol * delc, 0, nrow * delr))
plt.title('Water Table at the end of the second period (20 years)')
plt.colorbar(label='Water Table Elevation (meters)')
plt.show()
# Plot the head versus time
idx = (0, int(nrow / 2) - 1, int(ncol / 2) - 1)
ts = headobj.get_ts(idx)
fig = plt.figure(figsize=(6, 6))
ax = fig.add_subplot(1, 1, 1)
ttl = f"Head at cell ({idx[0] + 1},{idx[1] + 1},{idx[2] + 1})"
ax.set_title(ttl)
ax.set_xlabel("time")
ax.set_ylabel("head")
ax.plot(ts[:, 0], ts[:, 1], "bo-")
```

A series of figures is produced when the code is exposed.

The locations of pumping and artificial recharge wells for a given recharge period:

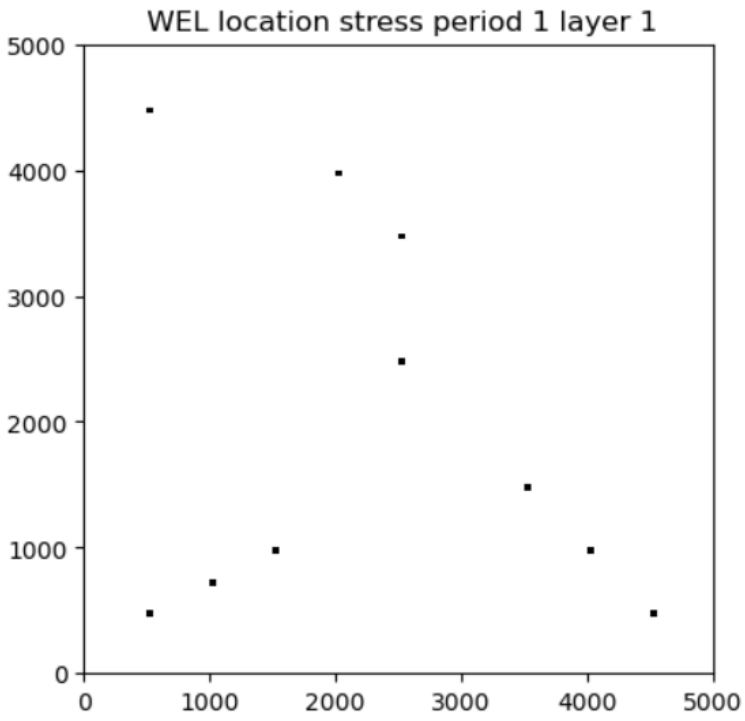

Figure 3.12 Wells position.

Color map of water head for the given groundwater level

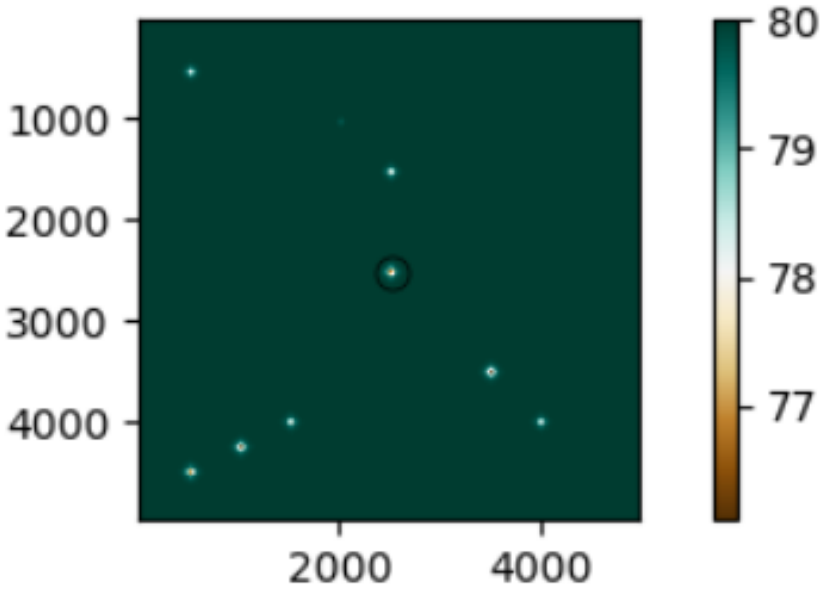

Figure 3.13 Head color map.

Cross-section of the aquifer along a given direction:

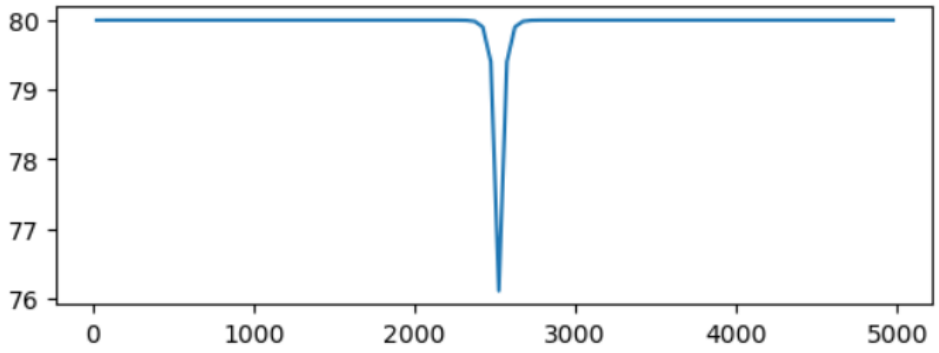

Figure 3.14 Groundwater Cross Section.

Change in Head in specific cells before and after applying the artificial enrichment:

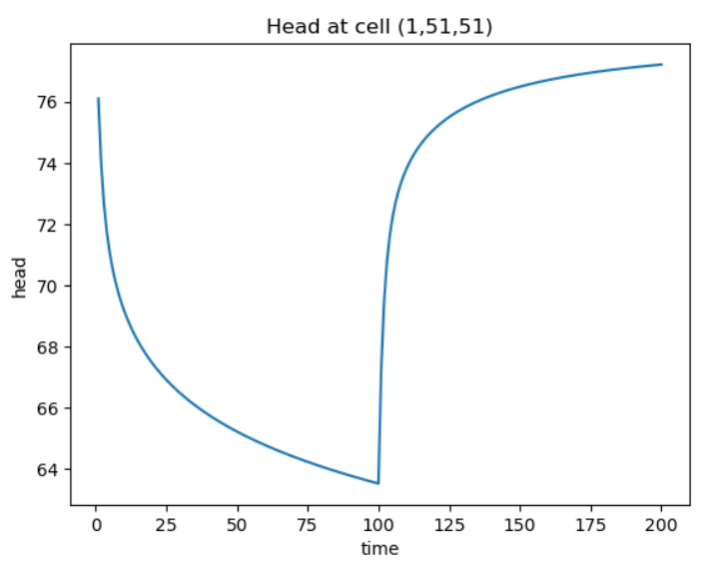

Figure 3.15 Head on a cell during time.

Water table at the end of given stress period:

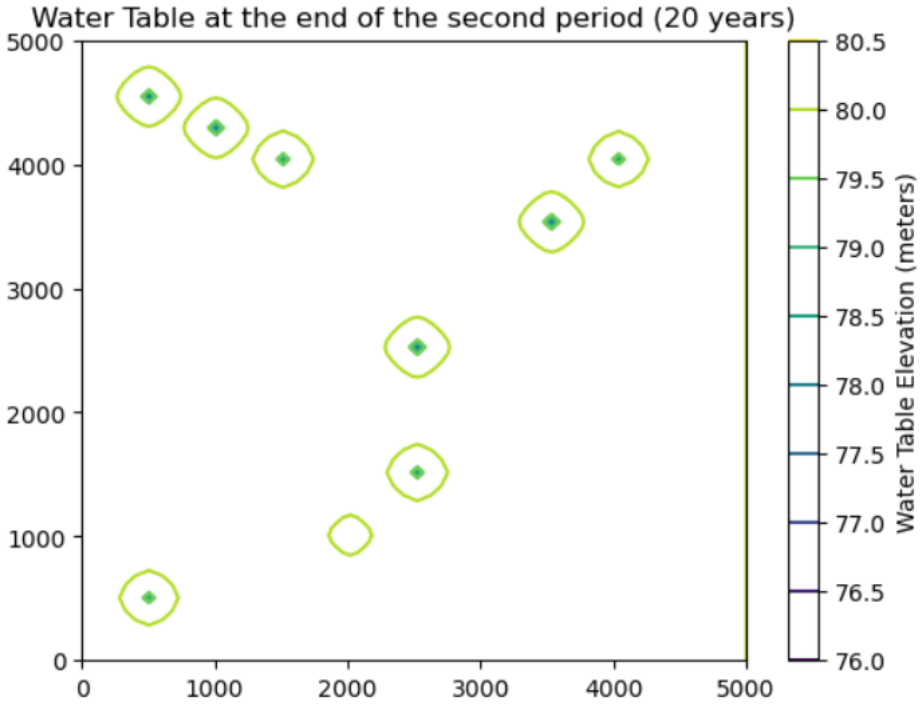

Figure 3.16 Water table of aquifer.

4 Snow algorithm

Snow is a critical parameter in the hydrological balance and is often the form of precipitation with a particularly significant inflow rate. It is an important source of supply for groundwater enrichment and the surface runoff of rivers and torrents (Figure 4.1), especially during the spring period (Sturm et al., 2010; Wesemann et al., 2018). Snow is therefore necessary to be considered in the hydrological planning and is important to accurately calculate the amount of snow height and its accumulation due to the affection morphological conditions, the climatic conditions and the vegetation cover (Pistocchi et al., 2017).

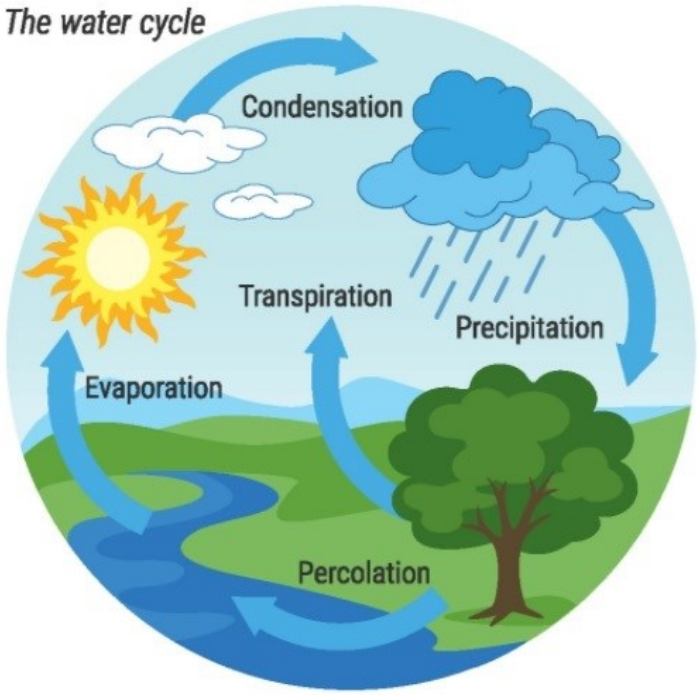

Figure 4.1*: The water cycle. Source: https://www.noblemg.com.au/articles/condensation-and-the-water-cycle.html*

At the same time, the snow parameters constitute critical input data in models for calculating the inflow and outflow in a basin (the volume of water stored in the snow is calculated as inflow), thus the accurate calculations from field measurements, with a small error rate is essential. For instance, the snow depth (SD) and the snow water equivalent (SWE) are key variables in many hydrological models and studies

involving groundwater recharge rates, basic flow preservation of rivers and hydropower dam sustainability.

To fully understand snow water trends, the most fundamental metric to monitor is SWE and SD. For monitoring these metrics, two potential methods are available (or combination of them): (i) passive microwave remote sensing and (ii) estimations based on direct measurements. Each of these methods has its strengths and limitations. For example, ground-based monitoring networks provide valuable and quick measurements in the field (Marty and Meister, 2012). However, field measurements are limited to specific regions and for certain periods of time. From the other hand, satellite monitoring of the snow parameters (Wang et al., 2009) can provide the long-term changes in a global scale. Satellite dataset limitations include cloud cover preventing the surface view (Gafurov et al., 2009), the obstruction of snow by dense vegetation, as well as the surface heterogeneity in mountain areas, which complicates the interpretation of the satellite dataset (Foppa, and Seiz, 2012; Hüsler et al., 2014) and induce an offset to the dataset.

Knowledge of snow parameters with high temporal and geographical resolution is used to improve the accuracy of the estimation procedure. Thus, a baseline regression model, based on satellite observations, is developed in order to analyze the snow parameters and optimize them. SWE patterns will be further examined, in order to configure whether they exhibit some level of repetition from year to year and to check any changes/trends in snow information, aiming to better estimate and improve the water resource management. In a following step, we will evaluate whether the historical (spatial and temporal) patterns of the snow coverage and snow water equivalent (SWE) can be used to estimate the future snow distribution over the Eastern Mediterranean.

The developed algorithm uses as input the GLDAS satellite data from the NASA website (https://search.earthdata.nasa.gov/search). The aim of the Global Land Data Assimilation System (GLDAS) is to ingest satellite and ground-based observational data products, using advanced land surface modeling and data assimilation techniques, to generate optimal fields of land surface states and fluxes (Rodell et al., 2004). The high-quality, global land surface fields provided by GLDAS support several current and proposed weather and climate predictions, water resources

applications, and water cycle investigations. The products are processed in high resolutions (2.5-degrees to 1 km), providing 3-hourly and daily and monthly mean values of snow depth and snow water equivalent. In the current study, the 3-hourly and daily values of the snow parameters are analyzed. The data set currently covers from January 1948 up today. An example of the satellite data analysis is given below in Figure 4.2. The dataset includes the information for the daily mean value of snow water equivalent (kg m^{-2}) for a broad area and then the user has to search for the region of interest (i.e., for specific coordinates).

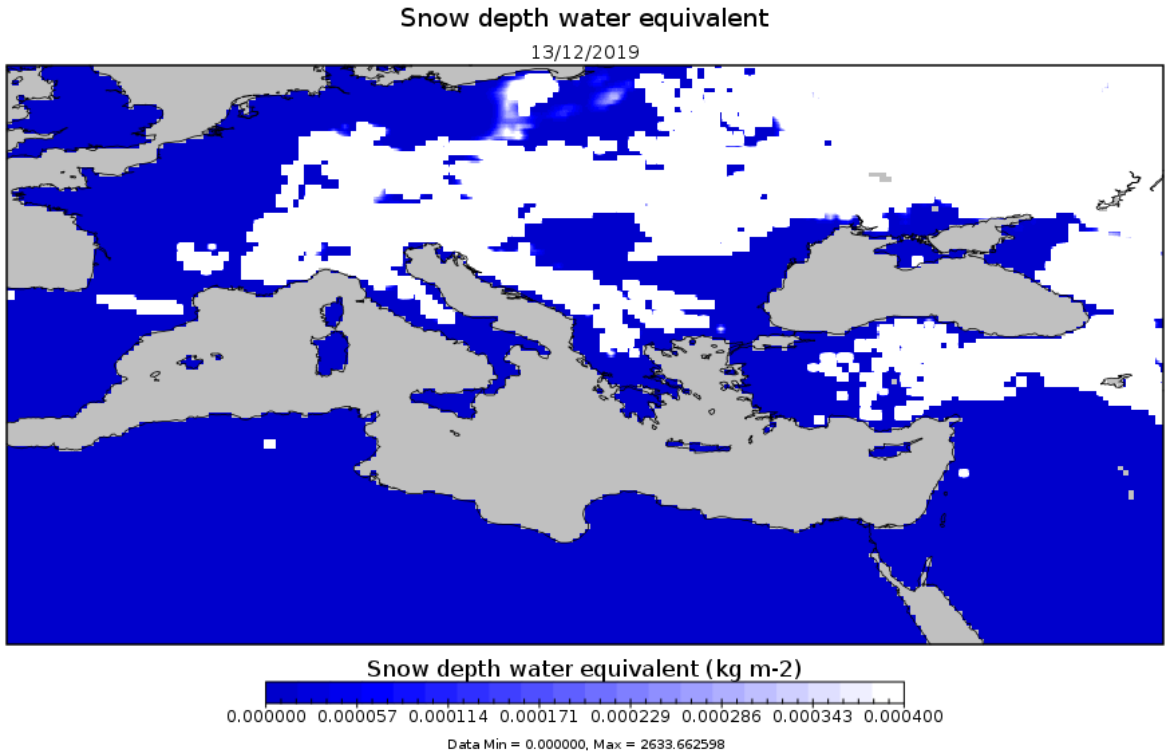

Figure 4.2 Mean monthly snow water equivalent for the period 13/12/2019 – 14/12/2019 from GLDAS.

4.1 Snow parameters

Generally, the snow water equivalent (SWE) represents the amount of water that is contained in a snowpack. Using SI units, it is measured in kg/m^2, which can be considered as the weight of the meltwater per square meter that would result if the snowpack was melted entirely. Given that SWE and snow depth (HS) are derived from the satellite data, the snow density (ρ) is then calculated following the equation:

$$\mathbf{SWE = HS * \rho\ ,} \qquad (1)$$

Where: SWE is measured in kg/m^2, HS in m, and ρ in kg/m^3

Typically, the snow density is estimated following the bibliography, following the equation:

$$\mathbf{\rho = 0.1 * \rho w} \text{ (the average density of snow is 1000 } kg/m^3) \qquad (2)$$

The snow density usually ranges from 100 to 500 kg/m^3 (Meløysund et al., 2007). However, the snow density is generally larger (due to the subsidence under the influence of gravity and other mechanisms), depending on the residence time of snow and the snow depth.

4.2 Algorithm description

The developed source code (snow_processing.py file) is written in python (https://www.python.org/), which is an interpreted high-level general-purpose programming language and is often used as a support language for software developers. The code uses as input satellite data (in a standard data netcdf format) from the NASA website (https://search.earthdata.nasa.gov/search). All libraries and modules needed to install and run the code are open source and freely available. Generally, the majority of the modules used in Python programming are accessed by using the import statement (https://docs.python.org/3/tutorial/modules.html). The user has to define the root of the stored satellite data (dir_path) and the storage of the output files (dir_out). Once the user defines the root paths and runs the program, the code starts the processing of the data and the exporting of files. The algorithm's outputs of Routine #1 are: (i) a csv file with written the values of the SWE and the SD in the selected areas and (ii) maps/quicklooks of the SWE for certain days, based on the selection of the user.

The output of the algorithm is then used in the second routine (Routine #2) for calculating the snow depth and for plotting the monthly/seasonal/yearly variation of the snow parameters.

Summarized below are the steps that have been applied during the processing, which consists of the following:

➢ Reading the satellite dataset and selecting the region of interest

➢ Data Screening

➢ Estimation of the spatial distribution of SWE and SD in the user selected area

➢ Calculation of the spatial distribution of the snow depth (HS) in the selected areas.

➢ Calculation of the monthly/seasonal/yearly variation of the snow parameters (SWE, HS, ρ) in the selected areas.

➢ Mapping the snow parameters in the selected areas.

Additionally, the validation of the satellite dataset with the in-situ information is made. The relative root mean squared error (rRMSE) and the Pearson correlation

coefficient are used as metrics of convergence between the satellite retrievals and the snow parameters of the reference ground-based stations. The comparison is made through the snow_comparison.py file.

4.3 Snow parameters evaluation

To make proper water management decisions, it is essential that SWE measurements be as effective and accurate as possible. Generally, water in a snowpack is determined by depth, density, type of snow, changes in the pack, previous freeze/thaw cycles, recent rainfall events, etc. Thus, validation of the satellite dataset with the in-situ information is made (snow_comparison.py). The relative root mean squared error (rRMSE) and the Pearson correlation coefficient will be used as metrics of convergence between the satellite retrievals and the snow parameters of the reference ground-based stations. The code is available in the website under the tab "Models".

4.3.1 Primary data collection

For the aforementioned purposes, two meteorological stations located in northern Greece are selected (Figure 4.3). The chosen reference areas cover different hydrogeological and climatological characteristics. The meteorological stations are located in Mount Athos (40°12'40.60"N, 24°15'48.36"E) and Kozani (40°24'0.77"N, 22°5'20.04"E) at an altitude of 889 m and 1302 m, respectively. These two meteorological stations are selected due to the morphological characteristics and meteorological conditions of both areas. Of the utmost importance is the great values of snow depth in Mount Athos compared to Kozani with an inversely proportional correlation of elevation (Voudouri et al., 2021). Thus, comparing the data of the two different recording points will enhance the results of the snow parameters analysis and optimization.

Specifically, field measurements of snowfall, snow accumulation and melting rate are daily provided. In general, snow depth is in general, quicker and easier to measure in the field than SWE (Sturm et al., 2010). The snow height measurements are recorded and analysed every hour from the stations. Both weather stations in Kozani (Zoodochos Pigi) and Mount Athos are part of the Laboratory of Engineering Geology and Hydrogeology, belonging to the Department of Geology of the Aristotle University of Thessaloniki. The comparison with ground based measurements will help to evaluate the satellite information and estimate possible limitations of the detection of the satellite sensor during snowfall events.

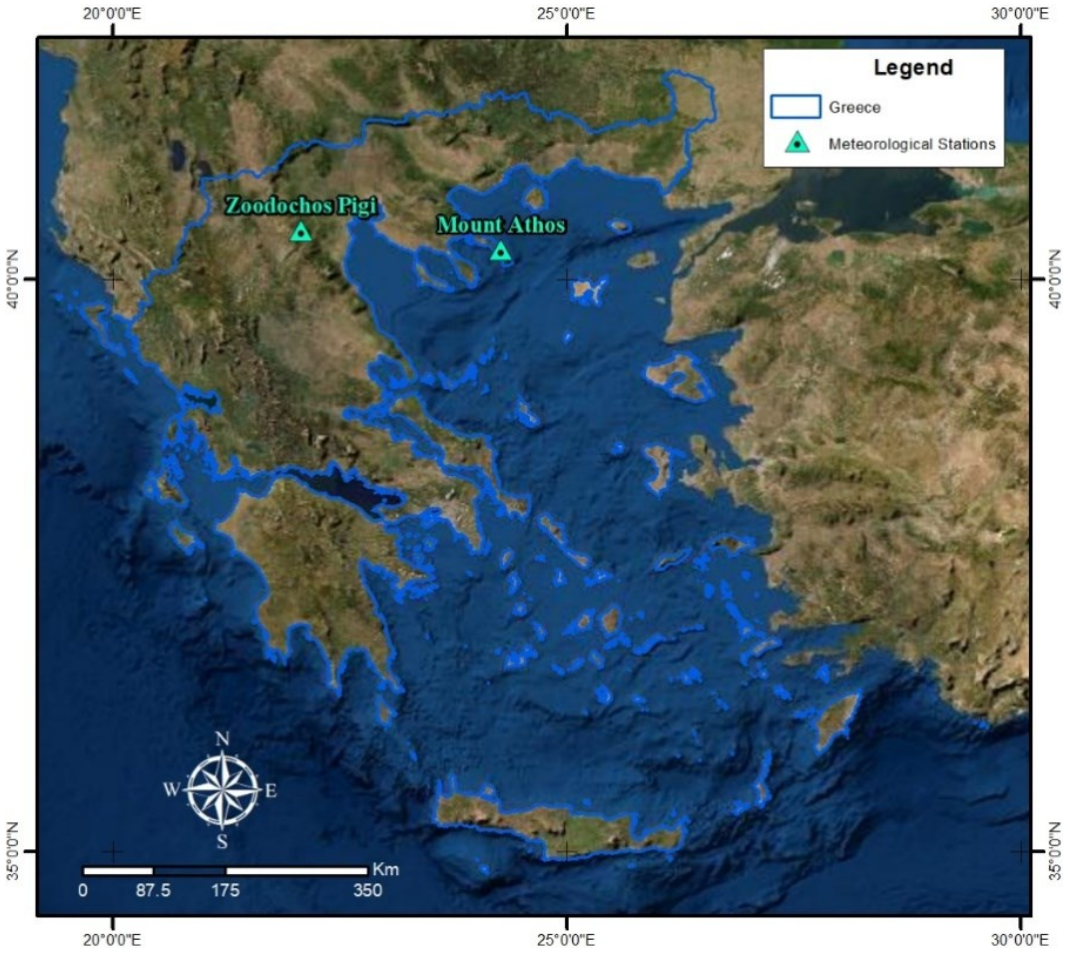

Figure 4.3 The reference meteorological stations in Kozani and Mount Athos regions.

More specifically, in the next step the following questions will be addressed:

- Could previous satellite spatial and temporal patterns of snow coverage and snow water equivalent (SWE) be used to estimate future snow distribution in different ground-based stations (i.e., reference stations)?
- Can the reconstructions from previous years be used as an explanatory variable in a statistical regression model to explain the spatial distribution of SWE in real time?
- How can we relate any observed differences in the geographical areas with the regional characteristics?

4.4 Source Code

In the next lines, the code is provided in a quite easy and readable way.

```
#######Routine 1
####### Importing the libraries and modules ##########
import os
import glob
import numpy as np
from netCDF4 import Dataset
from matplotlib import pyplot as plt
from scipy import interpolate
from pylab import savefig
from matplotlib import cm
import pandas as pd



##########Define the folders ######################
dir_main = os.environ['HOME']+'/'
dir_path=dir_main+'Sxoli/Programs/py_modules/snow/input/'
dir_out=dir_main+'Sxoli/Programs/py_modules/snow/plots/'

#Read filenames
search_log = os.path.join(dir_path, 'GL*')
file_path = glob.glob(search_log)
file_path.sort()
numfile = len(file_path)
print(file_path[0])

layers=600
features=1440

SWE= np.zeros((numfile,layers,features))
SD= np.zeros((numfile,layers,features))
lat=np.zeros((numfile,layers))
lon=np.zeros((numfile,features))
```

```
maskAg= np.zeros((numfile))
mask_lonAg= np.zeros((numfile))
maskK= np.zeros((numfile))
mask_lonK= np.zeros((numfile))

SWE_newAg= np.zeros((numfile))
SD_newAg= np.zeros((numfile))
SWE_newK= np.zeros((numfile))
SD_newK= np.zeros((numfile))
date= np.zeros((numfile))

for l in range (0,numfile):
fh = Dataset(file_path[l], mode='r')
date[l]= file_path[l][(len(file_path[l])-21):(len(file_path[l])-8)]
SWE[l,:,:] = fh.variables['SWE_inst'][:][:][:]
SD[l,:,:]=fh.variables['SnowDepth_inst'][:][:][:]
lat[l,:]=fh.variables['lat'][:][:]
lon[l,:]=fh.variables['lon'][:][:]
SWE[SWE<0] = 0
```

######**Define latitude & longitude of the study areas** #########

```
maskAg=lat[l,:]==40.125
mask_lonAg=lon[l,:]==22.125

maskK=lat[l,:]==40.375
mask_lonK=lon[l,:]==24.125


SD_newAg[l]=SD[l,maskAg,mask_lonAg]
SWE_newAg[l]=SWE[l,maskAg,mask_lonAg]
```

```
SD_newK[l]=SD[l,maskK,mask_lonK]
SWE_newK[l]=SWE[l,maskK,mask_lonK]

temp_panel = np.stack((SD_newAg,SWE_newAg),axis=1)
opt_panel = pd.DataFrame(data=temp_panel,index=date, columns=['Snow
Depth','SWE'])
opt_panel.to_csv(dir_out+'snow.csv')

########## Start plotting the SWE of a selected area##############

plt.figure(figsize=(10,10))

signalplot_interp=interpolate.RectBivariateSpline(lat[0,300:500],lon[0,700:1000],
SWE[0,300:500,700:1000])

plt.contourf(lon[0,700:1000],lat[0,300:500],signalplot_interp(lat[0,300:500],lon[0,70
0:1000]),cmap=cm.binary)
plt.title('Snow Coverage observed at '+str(date[0]),fontsize=22,fontweight='bold')

plt.xlabel('Longitute',fontsize=14,fontweight='bold')
plt.ylabel('Latitude',fontsize=14,fontweight='bold')
plt.annotate('Station ', fontsize=10,color="blue",xy=(lon[0,800],lat[0,400]),
xytext=(lon[0,800],lat[0,400]))

cbar =plt.colorbar(orientation="horizontal")
cbar.set_label('Snow water equivalent (m)',fontsize=14,fontweight='bold')

plt.savefig(dir_out+'SWE'+'.png', bbox_inches='tight',dpi=300)
plt.close()

###To run this routine put as dir_path# the ouput file from Routine 1
```

```
dir_path='C:\\Users\\Admin\\Desktop\\python\\inputs1\\snow.csv'          #### These paths should be defined by the user
dir_path1='C:\\Users\\Admin\\Desktop\\python\\inputs1\\snow_Kozani.csv'
dir_path2='C:\\Users\\Admin\\Desktop\\python\\inputs1\\snow_Italy.csv'
dir_out='C:\\Users\\Admin\\Desktop\\python\\plots\\'


data=np.loadtxt(dir_path, dtype=object,skiprows=1,delimiter=",")
date=data[:,0]


aer_time=np.zeros(date.shape[0],dtype=object)

for i in range(len(data)):
    year=int(date[i][0:4])
    month=int(date[i][5:7])
    day=int(date[i][8:10])
    aer_time[i] = dt.datetime(year,month,day)



SD = data[:,1].astype(float)
SWE = data[:,2].astype(float)

r=SWE/SD
hw=(SD*r)/1000
dataframe1 = pd.DataFrame(hw,aer_time)
data=np.stack((1000*SD,1000*hw),axis=1)
dataframe1 = pd.DataFrame(data=data,index=aer_time)
dataframe1=dataframe1.resample('Y').sum()
dataframe1.to_csv(dir_out+'Hs.csv')



mask=np.where(SD>0)
```

```
SD=SD[mask]

aer_time=aer_time[mask]
SWE=SWE[mask]

snow_panel = np.stack((SD,SWE),axis=1)
SNOW=pd.DataFrame(snow_panel,index=aer_time,columns=['SD', 'SWE'])
SNOW=SNOW.resample('Y').sum()



from scipy.stats import linregress
from sklearn.metrics import r2_score



x = SNOW.index.values.astype(float)
z = np.polyfit(SNOW.index.values.astype(float), SNOW['SD'], 1)
p = np.poly1d(z)



SNOW['SD'].plot(marker='o', linestyle='None')



plt.plot(pd.to_datetime(SNOW.index.values.astype(float)), p(x), "r--")

slope, intercept, r_value, p_value, std_err =
stats.linregress(SNOW.index.values.astype(float), SNOW['SD'])

line = slope*x+intercept
plt.plot(SNOW.index.values.astype(float), line, 'r--',
label='y={:.2e}x+{:.2f}'.format(slope,intercept))

plt.legend(['y={:.2e}x+{:.2f}'.format(slope,intercept)])
plt.legend(['y=6.16e$^{-20}$x+0.04'],loc='upper left')
```

```
plt.ylabel('Snow Depth (m)',color='maroon',fontsize=12,fontweight='bold')
plt.yticks(fontsize=10,fontweight='bold')
plt.xticks(fontsize=10,fontweight='bold')
plt.axvline(pd.to_datetime('2015-11-01'), color='r', linestyle='--', lw=2)
plt.ylim((0, 0.5))
plt.axvline(x=1.4987808e+18 ,linestyle='dotted', linewidth=1, color='red')
plt.savefig(dir_out+'snow'+'.png', bbox_inches='tight',dpi=300)
plt.close()


###################### SWE#################################


x = SNOW.index.values.astype(float)
z = np.polyfit(SNOW.index.values.astype(float), SNOW['SWE'], 1)
p = np.poly1d(z)
SNOW['SWE'].plot(marker='o',linestyle='None')

plt.plot(pd.to_datetime(SNOW.index.values.astype(float)), p(x), "r--")

slope, intercept, r_value, p_value, std_err =
stats.linregress(SNOW.index.values.astype(float), SNOW['SWE'])

line = slope*x+intercept
plt.plot(SNOW.index.values.astype(float), line, 'r--',
label='y={:.2e}x+{:.3f}'.format(slope,intercept))

plt.legend(['y={:.2e}x+{:.3f}'.format(slope,intercept)])
plt.legend(['y=1.40e$^{-17}$x+9.11'])
plt.ylabel('Snow Water Equivalent (kg*m$^{-
2}$)',color='maroon',fontsize=12,fontweight='bold')
plt.yticks(fontsize=10,fontweight='bold')
```

```
plt.xticks(fontsize=10,fontweight='bold')
plt.savefig(dir_out+'SWE'+'.png', bbox_inches='tight',dpi=300)
plt.close()


data1=np.loadtxt(dir_path1, dtype=object,skiprows=1,delimiter=",")

SD_Kozani = data1[:,1].astype(float)
SWE_Kozani = data1[:,2].astype(float)


date_Kozani=data[:,0]

aer_time_Kozani=np.zeros(date_Kozani.shape[0],dtype=object)

for i in range(len(data1)):
    year=int(date[i][0:4])
    month=int(date[i][5:7])
    day=int(date[i][8:10])
    aer_time_Kozani[i] = dt.datetime(year,month,day)


r_Kozani=SWE_Kozani/SD_Kozani
hw_Kozani=(SD_Kozani*r_Kozani)/1000

data_Kozani=np.stack((1000*SD_Kozani,1000*hw_Kozani),axis=1)
dataframe11 = pd.DataFrame(data=data_Kozani,index=aer_time_Kozani)
dataframe11=dataframe11.resample('Y').sum()
dataframe11.to_csv(dir_out+'Hs_Kozani.csv')


mask1=np.where(SD_Kozani>0)
SD_Kozani=SD_Kozani[mask1]
```

```
aer_time_Kozani=aer_time_Kozani[mask1]
SWE_Kozani=SWE_Kozani[mask1]

snow_panel_Kozani = np.stack((SD_Kozani,SWE_Kozani),axis=1)
SNOW_Kozani=pd.DataFrame(snow_panel_Kozani,index=aer_time_Kozani,colum
ns=['SD', 'SWE'])
SNOW_Kozani=SNOW_Kozani.resample('Y').sum()

data2=np.loadtxt(dir_path2, dtype=object,skiprows=1,delimiter=",")

SD_Italy = data2[:,1].astype(float)
SWE_Italy = data2[:,2].astype(float)



date_Italy=data[:,0]

aer_time_Italy=np.zeros(date_Italy.shape[0],dtype=object)

for i in range(len(data1)):
    year=int(date[i][0:4])
    month=int(date[i][5:7])
    day=int(date[i][8:10])

    aer_time_Italy[i] = dt.datetime(year,month,day)



r_Italy=SWE_Italy/SD_Italy
hw_Italy=(SD_Italy*r_Italy)/1000
data_Italy=np.stack((1000*SD_Italy,1000*hw_Italy),axis=1)
dataframe111 = pd.DataFrame(data=data_Italy,index=aer_time_Italy)
dataframe111=dataframe111.resample('Y').sum()
dataframe111.to_csv(dir_out+'Hs_Italy.csv')
```

```
mask2=np.where(SD_Italy>0)
SD_Italy=SD_Italy[mask2]

aer_time_Italy=aer_time_Italy[mask2]
SWE_Italy=SWE_Italy[mask2]

snow_panel_Italy = np.stack((SD_Italy,SWE_Italy),axis=1)
SNOW_Italy=pd.DataFrame(snow_panel_Italy,index=aer_time_Italy,columns=['SD'
, 'SWE'])
SNOW_Italy=SNOW_Italy.resample('Y').sum()


x = SNOW_Kozani.index.values.astype(float)
z = np.polyfit(SNOW_Kozani.index.values.astype(float), SNOW_Kozani['SD'], 1)
p = np.poly1d(z)

SNOW_Kozani['SD'].plot(marker='o',linestyle='None')

plt.plot(pd.to_datetime(SNOW_Kozani.index.values.astype(float)), p(x), "r--")


slope, intercept, r_value, p_value, std_err =
stats.linregress(SNOW_Kozani.index.values.astype(float), SNOW_Kozani['SD'])

line = slope*x+intercept
plt.plot(SNOW_Kozani.index.values.astype(float), line, 'r--',
label='y={:.2e}x+{:.2e}'.format(slope,intercept))

plt.legend(['y={:.2e}x+{:.2e}'.format(slope,intercept)])
plt.legend(['y=1.38e$^{-20}$x+4.2e$^{-3}$'])
```

```
plt.ylabel('Snow Depth (m)',color='maroon',fontsize=12,fontweight='bold')
plt.yticks(fontsize=10,fontweight='bold')
plt.xticks(fontsize=10,fontweight='bold')
plt.savefig(dir_out+'snow_Kozani'+'.png', bbox_inches='tight',dpi=300)
plt.close()



x = SNOW_Italy.index.values.astype(float)
z = np.polyfit(SNOW_Italy.index.values.astype(float), SNOW_Italy['SD'], 1)
p = np.poly1d(z)

SNOW_Italy['SD'].plot(marker='o',linestyle='None')

plt.plot(pd.to_datetime(SNOW_Italy.index.values.astype(float)), p(x), "r--")



slope, intercept, r_value, p_value, std_err =
stats.linregress(SNOW_Italy.index.values.astype(float), SNOW_Italy['SD'])

line = slope*x+intercept
plt.plot(SNOW_Italy.index.values.astype(float), line, 'r--',
label='y={:.2e}x+{:.2e}'.format(slope,intercept))



# plt.legend(['y=6.16e-20x+0.04'])
plt.legend(['y={:.2e}x+{:.2e}'.format(slope,intercept)])
plt.legend(['y=7.52e$^{-22}$x+6.72e$^{-4}$'])
plt.ylabel('Snow Depth (m)',color='maroon',fontsize=12,fontweight='bold')
plt.yticks(fontsize=10,fontweight='bold')
plt.xticks(fontsize=10,fontweight='bold')



plt.savefig(dir_out+'snow_Italy'+'.png', bbox_inches='tight',dpi=300)
```

```
plt.close()

######################################## SWE ########################


x = SNOW_Kozani.index.values.astype(float)
z = np.polyfit(SNOW_Kozani.index.values.astype(float), SNOW_Kozani['SWE'], 1)
p = np.poly1d(z)

SNOW_Kozani['SWE'].plot(marker='o', linestyle='None')

plt.plot(pd.to_datetime(SNOW_Kozani.index.values.astype(float)), p(x), "r--")

slope, intercept, r_value, p_value, std_err =
stats.linregress(SNOW_Kozani.index.values.astype(float), SNOW_Kozani['SWE'])

line = slope*x+intercept
plt.plot(SNOW_Kozani.index.values.astype(float), line, 'r--',
label='y={:.2e}x+{:.2f}'.format(slope,intercept))



# plt.legend(['y=6.16e-20x+0.04'])
plt.legend(['y={:.2e}x+{:.2f}'.format(slope,intercept)])
plt.legend(['y=2.98e$^{-18}$x+0.77'])
plt.ylabel('Snow Water Equivalent (kg*m$^{-
2}$)',color='maroon',fontsize=12,fontweight='bold')
plt.yticks(fontsize=10,fontweight='bold')
plt.xticks(fontsize=10,fontweight='bold')
plt.savefig(dir_out+'SWE_Kozani'+'.png', bbox_inches='tight',dpi=300)
plt.close()

x = SNOW_Italy.index.values.astype(float)
z = np.polyfit(SNOW_Italy.index.values.astype(float), SNOW_Italy['SWE'], 1)
```

```
p = np.poly1d(z)

SNOW_Italy['SWE'].plot(marker='o', linestyle='None')

plt.plot(pd.to_datetime(SNOW_Italy.index.values.astype(float)), p(x), "r--")

slope, intercept, r_value, p_value, std_err =
stats.linregress(SNOW_Italy.index.values.astype(float), SNOW_Italy['SWE'])

line = slope*x+intercept
plt.plot(SNOW_Italy.index.values.astype(float), line, 'r--',
label='y={:.2e}x+{:.2f}'.format(slope,intercept))

plt.legend(['y={:.2e}x+{:.2f}'.format(slope,intercept)])
plt.legend(['y=1.86e$^{-19}$x+0.14'])

plt.ylabel('Snow Water Equivalent (kg*m$^{-
2}$)',color='maroon',fontsize=12,fontweight='bold')
plt.yticks(fontsize=10,fontweight='bold')
plt.xticks(fontsize=10,fontweight='bold')
plt.savefig(dir_out+'SWE_Italy'+'.png', bbox_inches='tight',dpi=300)
plt.close()




##########################Groupby month ########################
# x=[1,2,3,4,5,6,7,8,9,10,11,12]
# SNOW['SD'].groupby([lambda x: x.month]).mean().plot(marker='o',color='blue',
linestyle='None')
# SNOW['SD'].groupby([lambda x: x.month]).std().plot(linestyle='None')
# # SNOW['SD'].groupby([lambda x: x.month]).mean().plot.bar(color='blue')

# plt.xlabel('Month',color='maroon',fontsize=12,fontweight='bold')
```

```
#
plt.xticks(x,['January','February','March','April','May','June','July','August','September
','October','November','December'],fontsize=10,fontweight='bold', rotation=60)

# plt.ylabel('Snow Depth (m)',color='maroon',fontsize=12,fontweight='bold')
# plt.yticks(fontsize=10,fontweight='bold')

# plt.savefig(dir_out+'SD_monthly'+'.png', bbox_inches='tight',dpi=300)
# plt.close()



# SNOW_Kozani['SD'].groupby([lambda x:
x.month]).mean().plot(marker='o',color='blue', linestyle='None')



# # SNOW_Kozani['SD'].groupby([lambda x:
x.month]).mean().plot.bar(color='blue')

# s=SNOW_Kozani['SD'].groupby([lambda x: x.month]).mean()
# # s.boxplot(color='blue')
# seaborn.boxplot(SNOW_Kozani['SD'].groupby([lambda x:
x.month]).mean().index, SNOW_Kozani['SD'].groupby([lambda x:
x.month]).mean())

# plt.xlabel('Month',color='maroon',fontsize=12,fontweight='bold')
# plt.xticks(fontsize=10,fontweight='bold')

# plt.ylabel('Snow Depth (m)',color='maroon',fontsize=12,fontweight='bold')
# plt.yticks(fontsize=10,fontweight='bold')

# plt.savefig(dir_out+'SD_monthly_Kozani'+'.png', bbox_inches='tight',dpi=300)
# plt.close()
```

```
# x=[1,2,3,4]
# SNOW_Kozani['SD'].groupby([lambda x: x.quarter]).mean().plot(marker='o',
color='blue',linestyle='None')
# plt.xticks(x,['Winter','Spring','Summer','Winter'],fontsize=10,fontweight='bold')

# y=[1,2,3,4,5,6,7,8,9,10,11,12]
# g =SNOW['SD'].groupby([lambda x: x.month]).std()
# SNOW['SD'].groupby([lambda x: x.month]).mean().plot(kind = "barh", y =
"mean", legend = False,
#             title = "Average Prices",xerr = g)
# plt.xlim(0,0.08)
# plt.close()
# plt.xticks([0.,0.01,0.02,0.03,0.04,0.05,0.06,0.07,0.08])

#
plt.yticks(y,['January','February','March','April','May','June','July','August','September
','October','November','December'],fontsize=10,fontweight='bold', rotation=10)

# seaborn.boxplot(x = SNOW_Kozani['SD'].index.month,y=SNOW_Kozani['SD'])
# seaborn.boxplot(x = SNOW['SD'].index.month,y=SNOW['SD'],color='navy')
# plt.ylabel('Snow Depth (m)',color='maroon',fontsize=12,fontweight='bold')
# plt.yticks(fontsize=10,fontweight='bold')
# plt.xticks(x=[1,2,3,4,5,6,7,8,9,10,11,12],fontsize=10,fontweight='bold')
# plt.xlabel('Month',color='maroon',fontsize=12,fontweight='bold')
# plt.savefig(dir_out+'SD_barplotmonthly'+'.png', bbox_inches='tight',dpi=300)
# plt.close()

# seaborn.boxplot(x =
SNOW_Kozani['SD'].index.month,y=SNOW_Kozani['SD'],color='navy')
# plt.ylabel('Snow Depth (m)',color='maroon',fontsize=12,fontweight='bold')
# plt.yticks(fontsize=10,fontweight='bold')
# plt.xlabel('Month',color='maroon',fontsize=12,fontweight='bold')
# plt.xticks(fontsize=10,fontweight='bold')
```

```
# plt.savefig(dir_out+'SD_Kozani_barplotmonthly'+'.png',
bbox_inches='tight',dpi=300)
# plt.close()



# seaborn.boxplot(x =
SNOW_Italy['SD'].index.month,y=SNOW_Italy['SD'],color='navy')
# plt.ylabel('Snow Depth (m)',color='maroon',fontsize=12,fontweight='bold')
# plt.yticks(fontsize=10,fontweight='bold')
# plt.xlabel('Month',color='maroon',fontsize=12,fontweight='bold')
# plt.xticks(fontsize=10,fontweight='bold')
# plt.savefig(dir_out+'SD_Italy_barplotmonthly'+'.png',
bbox_inches='tight',dpi=300)
# plt.close()



# seaborn.boxplot(x = SNOW['SWE'].index.month,y=SNOW['SWE'],color='orange')
# plt.ylabel('Snow Water Equivalent
(kg/m2)',color='maroon',fontsize=12,fontweight='bold')
# plt.yticks(fontsize=10,fontweight='bold')
# plt.xticks(fontsize=10,fontweight='bold')
# plt.xlabel('Month',color='maroon',fontsize=12,fontweight='bold')
# plt.savefig(dir_out+'SWE_barplotmonthly'+'.png', bbox_inches='tight',dpi=300)
# plt.close()

# seaborn.boxplot(x =
SNOW_Kozani['SWE'].index.month,y=SNOW_Kozani['SWE'],color='orange')
# plt.ylabel('Snow Water Equivalent
(kg/m2)',color='maroon',fontsize=12,fontweight='bold')
# plt.yticks(fontsize=10,fontweight='bold')
# plt.xlabel('Month',color='maroon',fontsize=12,fontweight='bold')
```

```
# plt.xticks(fontsize=10,fontweight='bold')
# plt.savefig(dir_out+'SWE_Kozani_barplotmonthly'+'.png',
bbox_inches='tight',dpi=300)
# plt.close()

# seaborn.boxplot(x =
SNOW_Italy['SWE'].index.month,y=SNOW_Italy['SWE'],color='orange')
# plt.ylabel('Snow Water Equivalent
(kg/m2)',color='maroon',fontsize=12,fontweight='bold')
# plt.yticks(fontsize=10,fontweight='bold')
# plt.xlabel('Month',color='maroon',fontsize=12,fontweight='bold')
# plt.xticks(fontsize=10,fontweight='bold')
# plt.savefig(dir_out+'SWE_Italy_barplotmonthly'+'.png',
bbox_inches='tight',dpi=300)
# plt.close()
```

5 Meteorological parameters

5.1 Standardized Precipitation Index (SPI) and Standardized Precipitation Evapotranspiration Index (SPEI)

The Standardized Precipitation Index (SPI) (McKee, 1993) and the Standardized Precipitation Evapotranspiration Index (SPEI) (Vicente-Serrano et al., 2010) were applied to identify the drought severity in the study areas. SPI and SPEI can be calculated on as little as 20 years' worth of data, but ideally, the time series should have a minimum of 30 years of data. Unfortunately, the collected data from station Dionisos Attiki in Marathonas basin are not enough for the calculation.

Monthly data of minimum, maximum temperature, and rainfall values were collected for the application of SPI from 6 stations:

- 4 stations in Campania region: Alife, Sorgenti Grassano, Alvignano, Melizzano for about 18 to 21 years of data.
- 1 station in Mouriki basin: Aristotelis station from 2000 to 2019.
- 1 station in Thermaikos Gulf: Makedonia Airport station from 1958 to 2023.

The drought index was calculated for different aggregation periods (1, 3, and 6 months) in the R programming code (Equation 5.1). The selection of these two drought indices was applied due to their limited data needs. The 6-month time scale had the optimum results. Drought events were classified based on SPI values (Table 5.1). R programming language was chosen instead of DrinC (Drought Indices Calculator) software for the calculation of SPI and SPEI. Additionally, R used for the time series analysis of meteorological and other hydrological data in order to obtain statistical values. R is flexible with versatile packages for various applications. Figure 5.2, Figure 5.3 and Figure 5.4 present the monthly temporal time series of drought index (SPI-6) based on the data from meteorological stations in the Campania region. The monthly SPI based on the data from the Aristotelis and Makedonia stations is provided in Figure 5.5 and Figure 5.6 respectively. The results showed no long duration of significant drought events in the Italian region, while mild drought events were mentioned in Mouriki basin. In Thermaikos Gulf, severe drought events were highlighted during 2011-2014.

SPEI was calculated with Hargreaves method, which requires minimum and maximum temperature values and latitude of the point data (Equation 5.2, 5.3 and 5.4). The monthly temporal time series of drought index (SPEI6) based on the data from meteorological stations in the Campania are presented in Figure 5.7, Figure 5.8, Figure 5.9 and Figure 5.10. The variation of monthly SPEI on the data from the Aristotelis and Makedonia stations is provided in Figure 5.11 and Figure 5.12 respectively. Although both indices are highly correlated with each other, SPEI performed better than SPI because evaporative demand has a positive impact on defining drought conditions.

spi <- spi(data_base$PCP, spi_time_step) **(5.1)**

data_base$PET<- Hargreaves (Tmin = data_base $T_min, Tmax = data_base $T_max, lat = latitude) **(5.2)**

WBal <- data_base $PCP - data_base $PET **(5.3)**

spei <- spei(WBal, spi_time_step) **(5.4)**

Where WBal is the water balance

PET is the potential evapotranspiration

PCP is the precipitation

Table 5.1 Drought categories based on SPI and SPEI values (McKee, 1993).

SPI/SPEI	Category
>2.00	Extremely wet
1.50 to 1.99	Very wet
1.00 to 1.49	Moderately wet
-0.99 to 0.99	Mild drought
-1.00 to -1.49	Moderate drought
-1.50 to -1.99	Severe drought
≤-2.00	Extreme drought

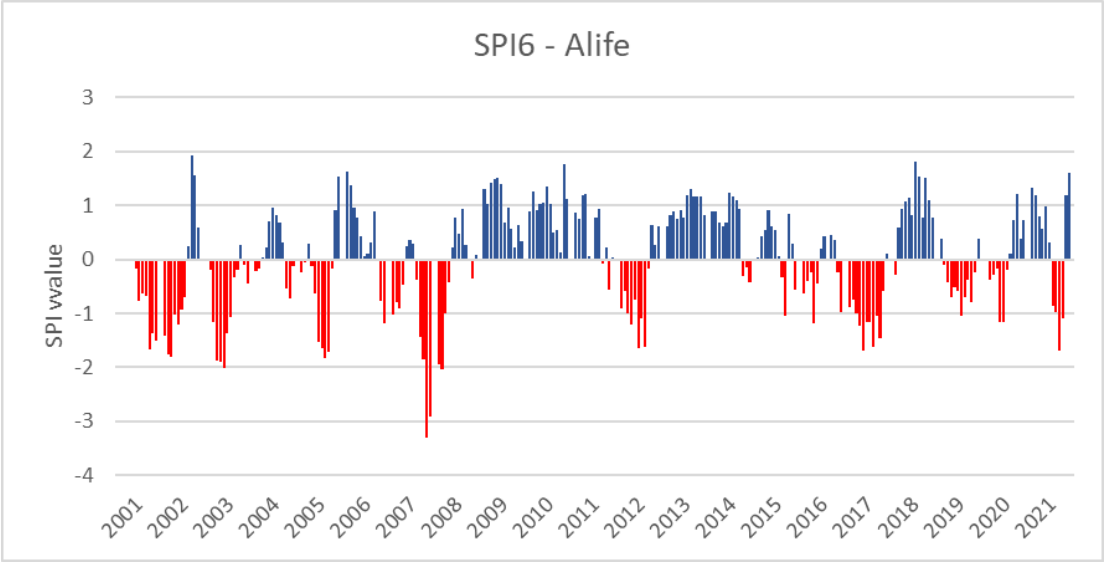

Figure 5.1 SPI in the period 2001-2021 for Alife meteorological station.

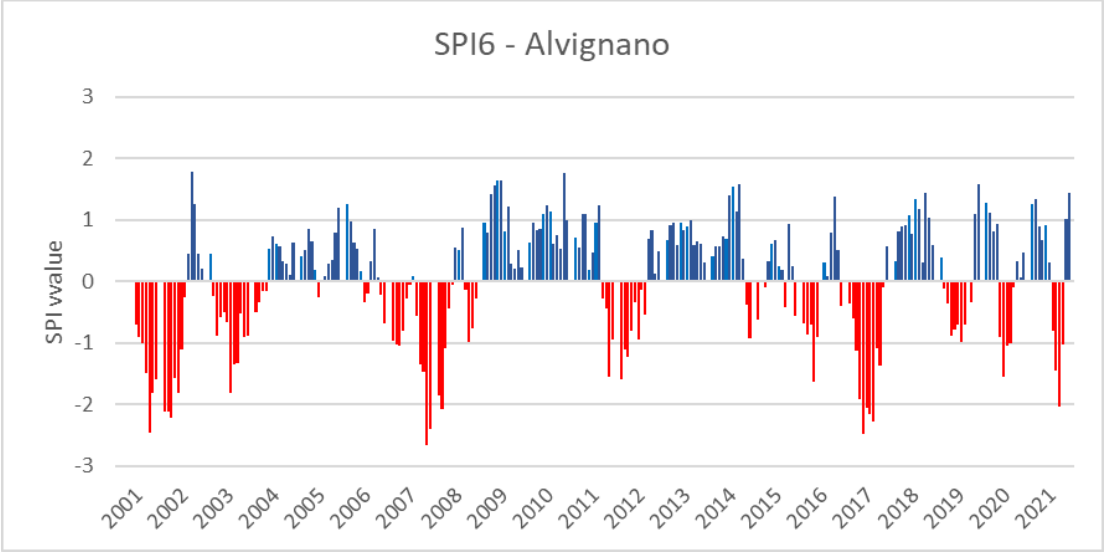

Figure 5.2 SPI in the period 2001-2021 for Alvignano meteorological station.

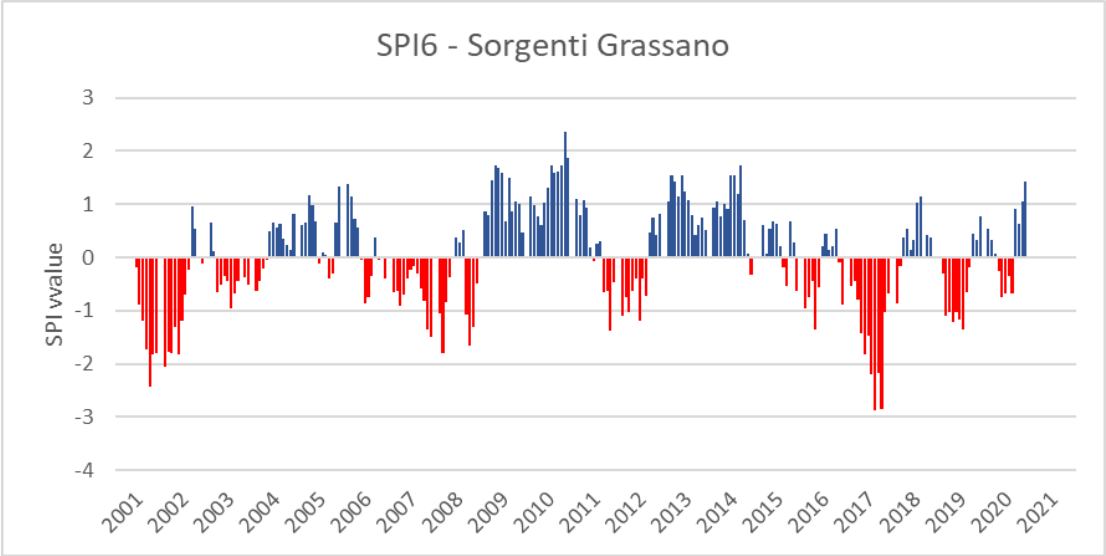

Figure 5.3 SPI in the period 2001-2020 for Sorgenti Grassano meteorological station.

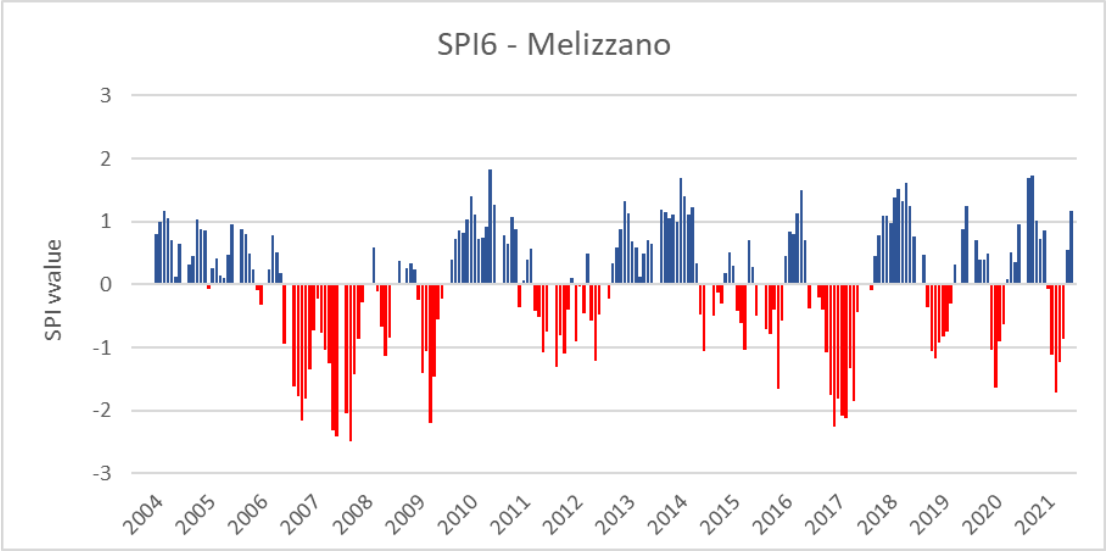

Figure 5.4 SPI in the period 2004-2021 for Mellizano meteorological station.

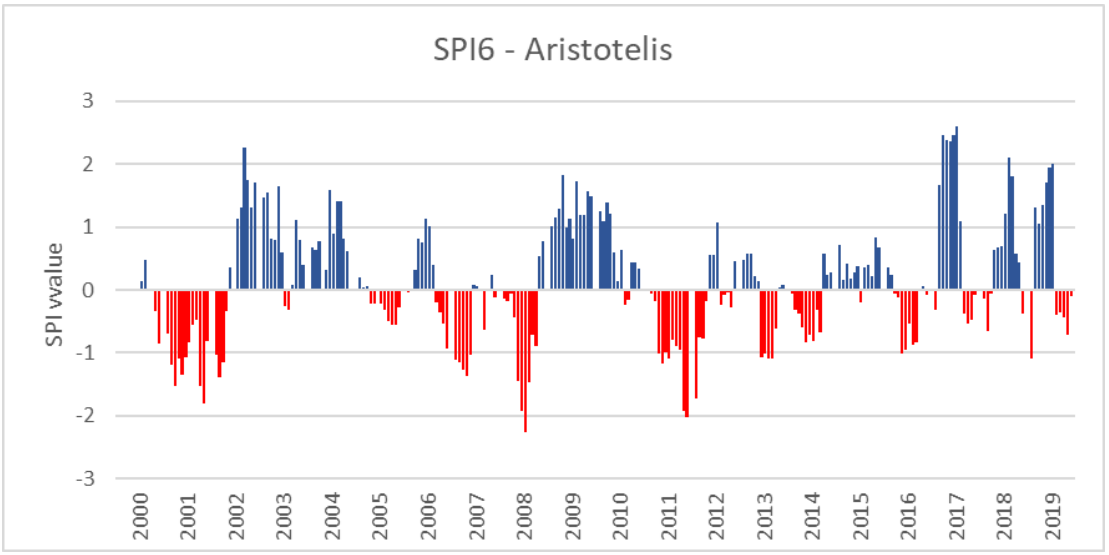

Figure 5.5 SPI in the period 2000-2019 for Aristotelis meteorological station.

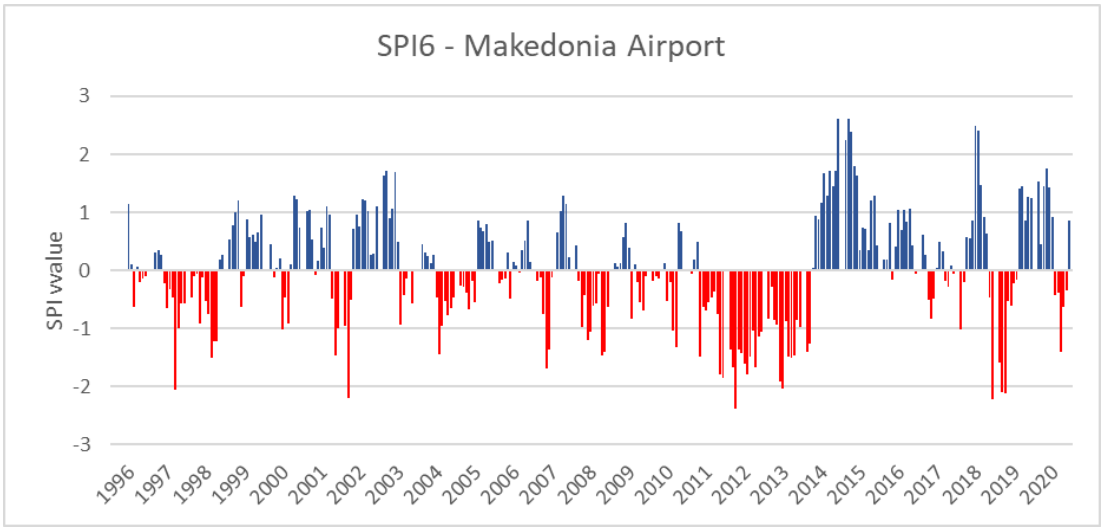

Figure 5.6 SPI in the period 1996-2021 for Makedonia Airport meteorological station.

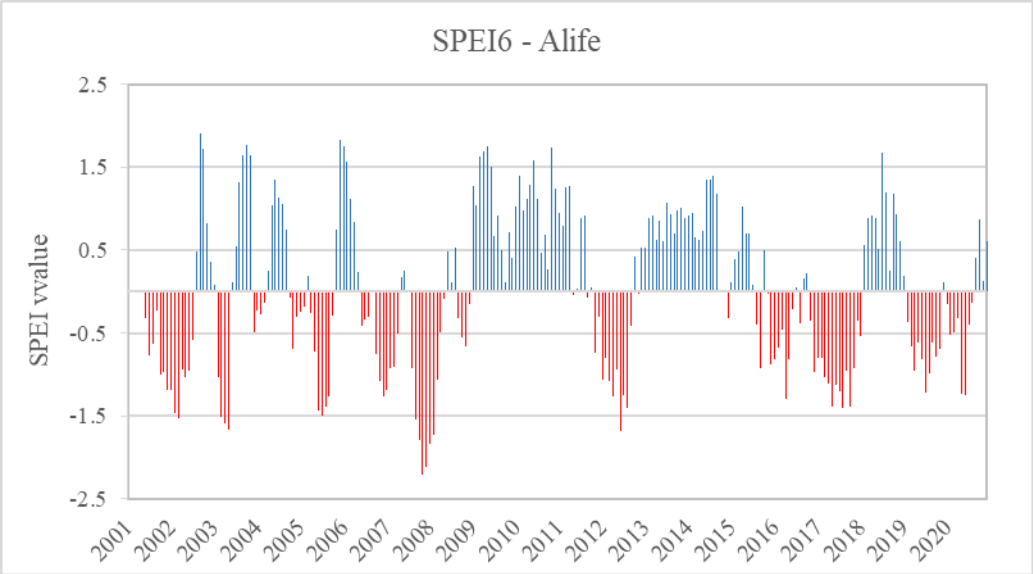

Figure 5.7 SPEI in the period 2001-2020 for Alife meteorological station.

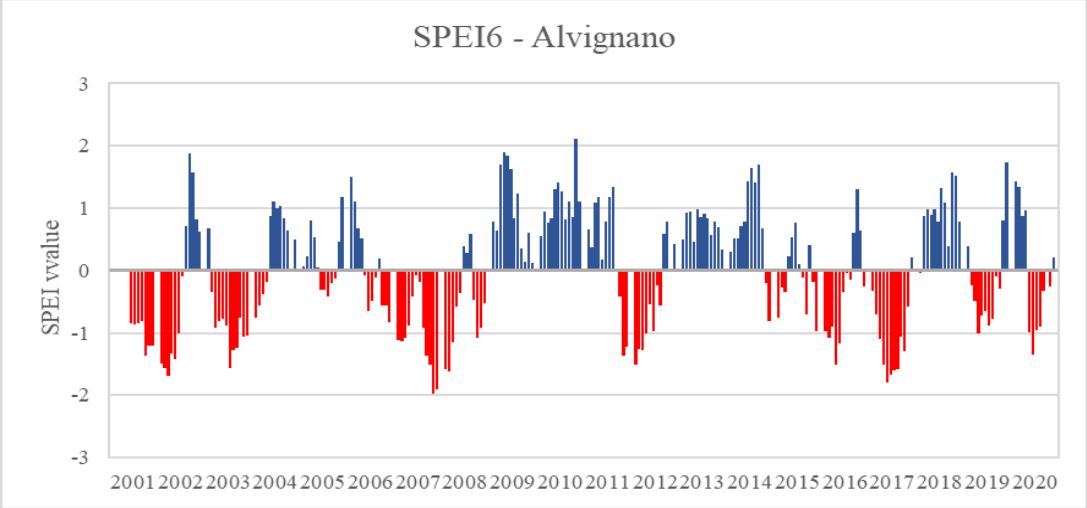

Figure 5.8 SPEI in the period 2001-2020 for Alvignano meteorological station.

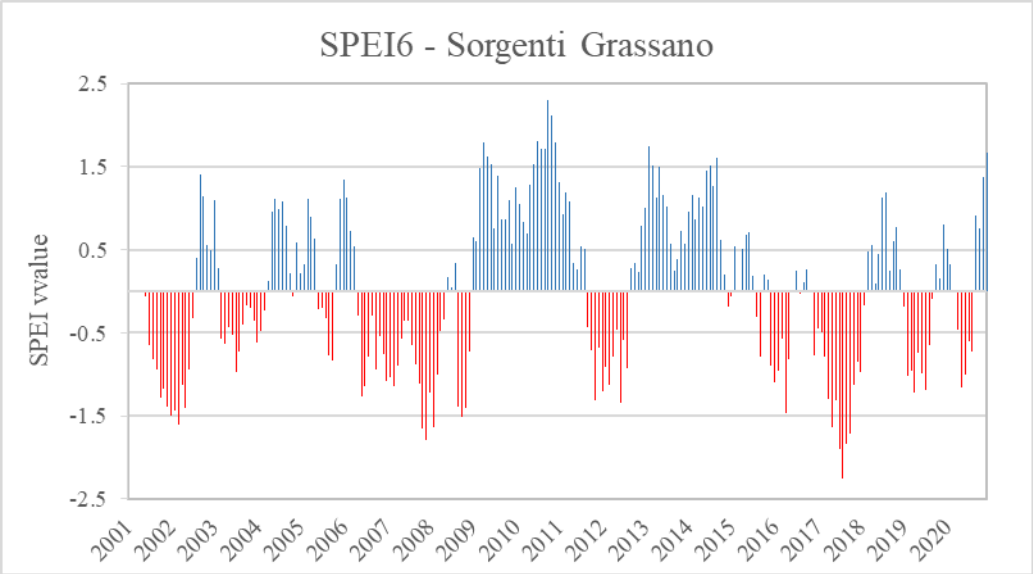

Figure 5.9 SPEI in the period 2001-2020 for Sorgenti Grassano meteorological station.

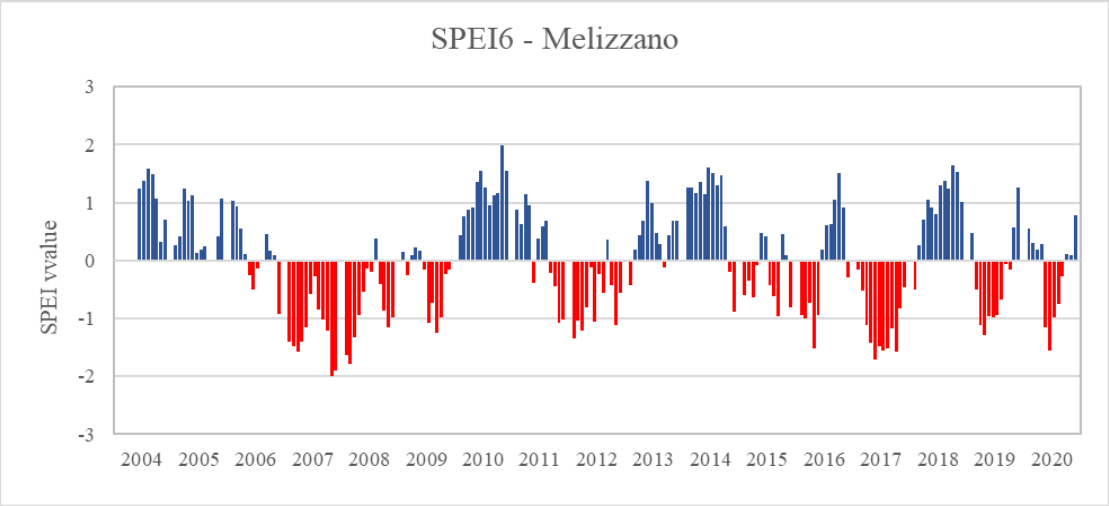

Figure 5.10 SPEI in the period 2004-2020 for Mellizano meteorological station.

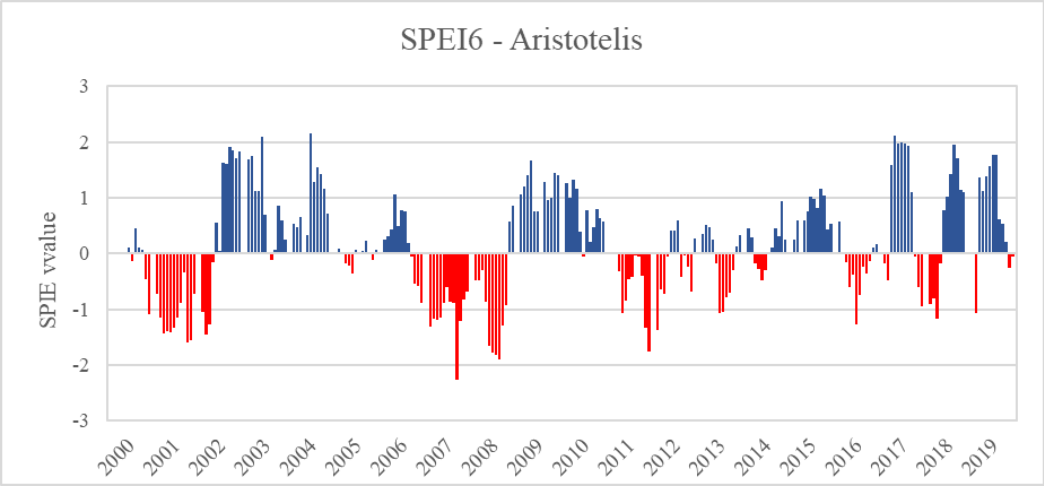

Figure 5.11 SPEI in the period 2000-2019 for Aristotelis meteorological station.

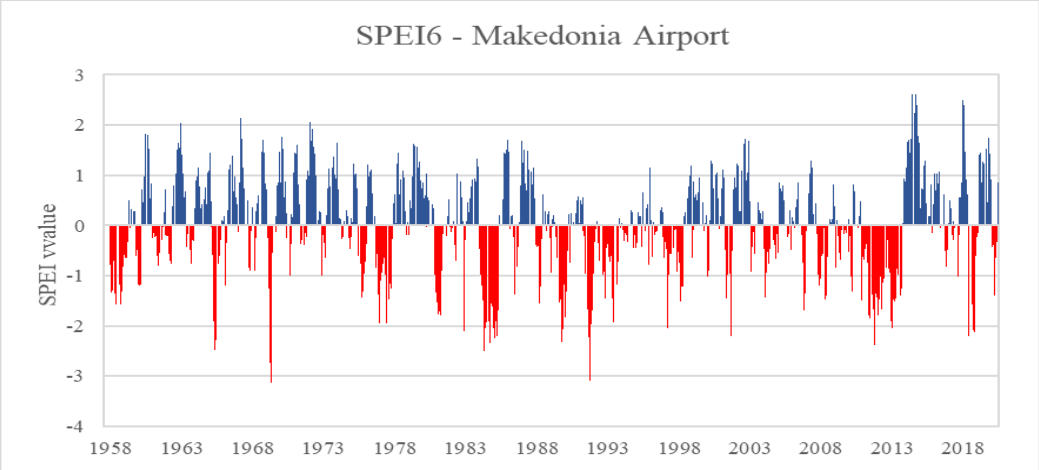

Figure 5.12 SPEI in the period 1958-2020 for Makedonia Airport meteorological station.

5.1.1 Forecasted SPEI

SPEI was calculated based on the future scenario RCP 4.5 in the study areas. In Eastern Thermaikos Gulf, long periods of severe drought are expected based on the RCP 4.5 scenario (Figure 5.13). In Mouriki basin, moderate drought events are mainly highlighted in the forecasted period. The longest moderate episode in the prediction of drought conditions will possibly last three years between 2025 and 2028 (

Figure 5.14). Severe drought events are mentioned during the years 2032, 2039 and 2040. Future very wet conditions are noted during the years 2035, 2036-2038. Severe drought events are mentioned in Campania basin during the period 2032-2040 (

Figure 5.15). Mild drought episodes are mainly highlighted in the forecasted period. During the years 2030-2040, no very wet and extreme wet years are expected

based on the future scenario. In Marathonas basin, moderate to severe drought events are mentioned during 2034-2039 (Figure 5.16) with the absence of complete wet years.

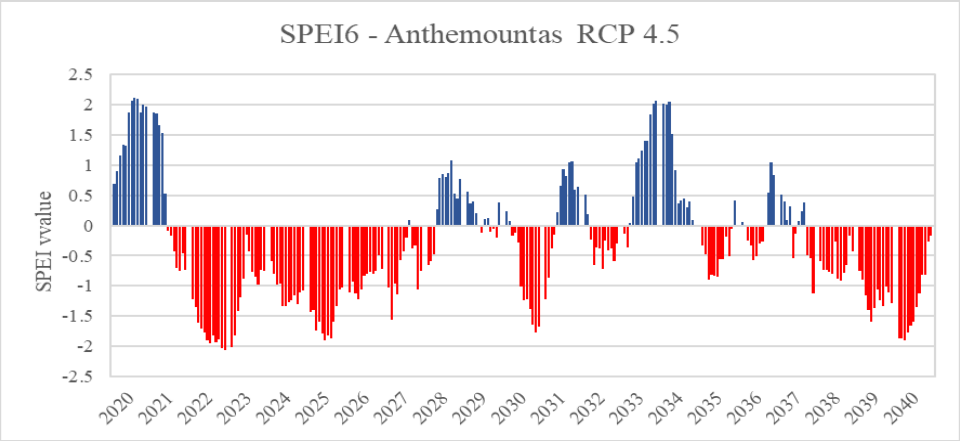

Figure 5.13 Forecasted SPEI in the period 2020-2040 for Eastern Thermaikos Gulf.

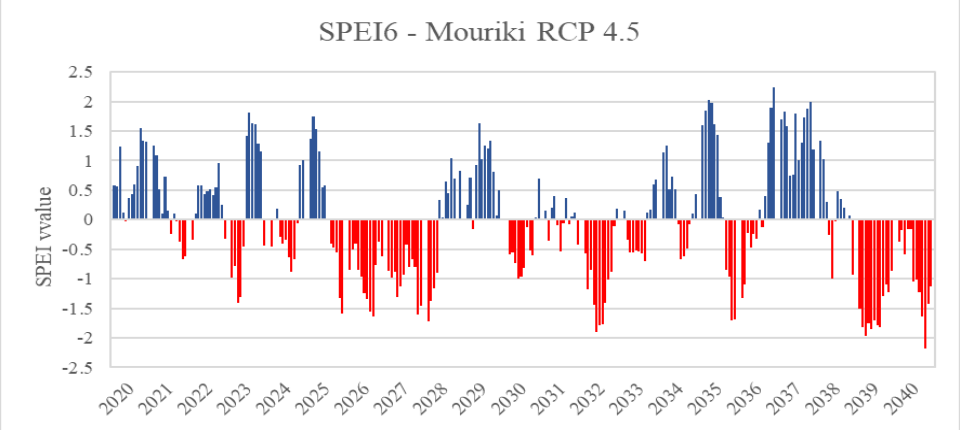

Figure 5.14 Forecasted SPEI in the period 2020-2040 for Mouriki basin.

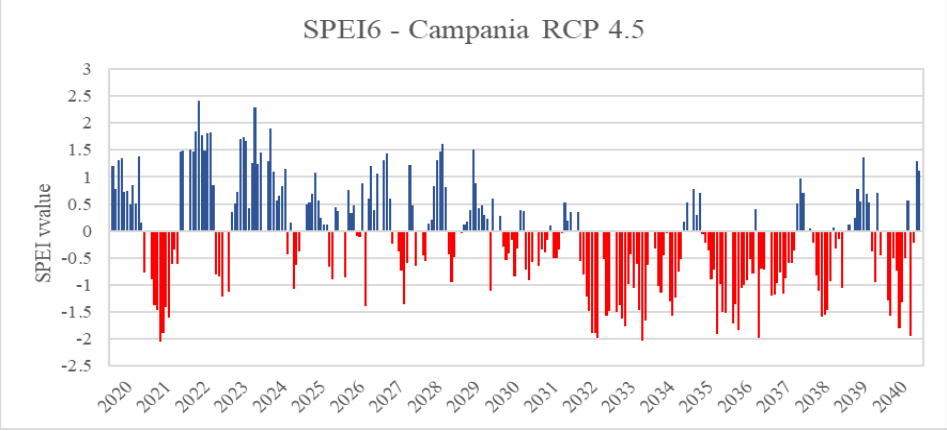

Figure 5.15 Forecasted SPEI in the period 2020-2040 for Campania basin.

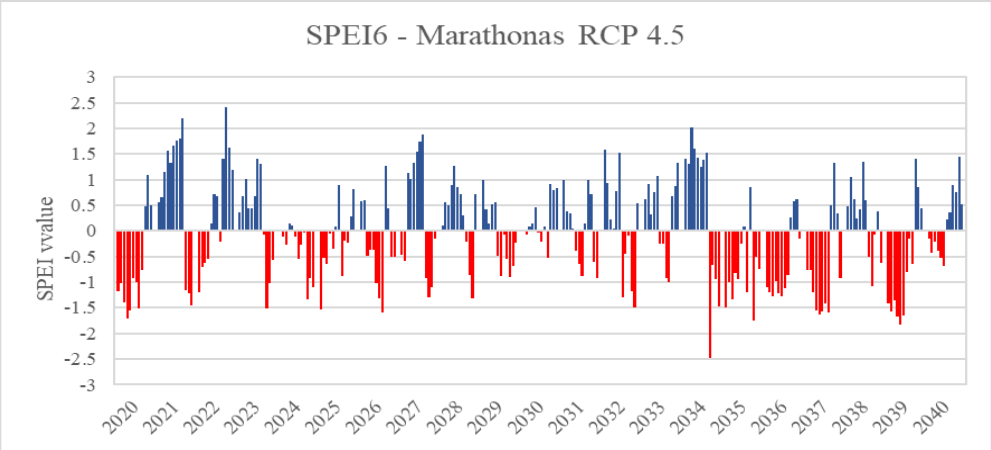

Figure 5.16 Forecasted SPEI in the period 2020-2040 for Marathonas basin.

5.2 PDSI

5.3 Rain intensity

Rainfall intensity plays an important role in runoff and recharge of aquifer. To investigate the variation of rainfall intensity during the years, the daily rainfall intensity index (RII) was applied (Equation 5.5). Rainfall values from six meteorological stations were used for the calculation of the rainfall intensity in the study areas with daily rain values collected from the stations.

$$RII = \frac{R}{D} \quad \textbf{(5.5)}$$

Where,

R is the sum of the rain values in a month (mm)

D is the number of rainy days in a month (day)

The daily rain values were divided by the total rainy days of the month of each year. The days with rain values < 1mm were removed from the calculation and characterized as “no wet days”. In Campania region, many rainfall episodes of high rainfall intensity are highted. The rain data from Alife station show high frequency of

heavy rainfall episodes (Figure 5.17) while the highest values of RII are noted by the rain data recorded by the Sorgenti Grassano (

Figure 5.18) and Trevico (Figure 5.19) meteorological stations. Benevento station recorded manly low intense rainfall episodes (Figure 5.20). Where 94.6 mm of rain are recorded in 4 days rainfall episode in March 2005 and 59 mm of rain in 2 days rainfall episode in July 2012.

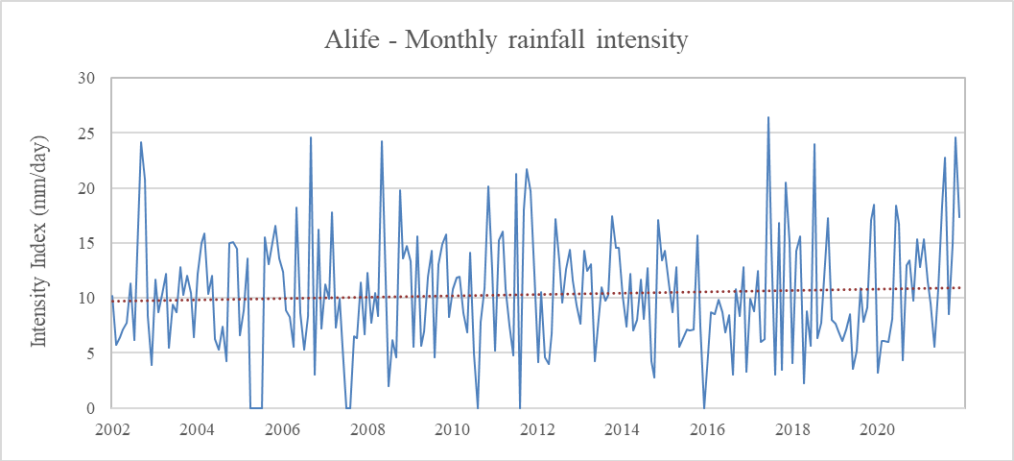

Figure 5.17 Variation of rainfall intensity during the period 2002-2021 for Alife meteorological station.

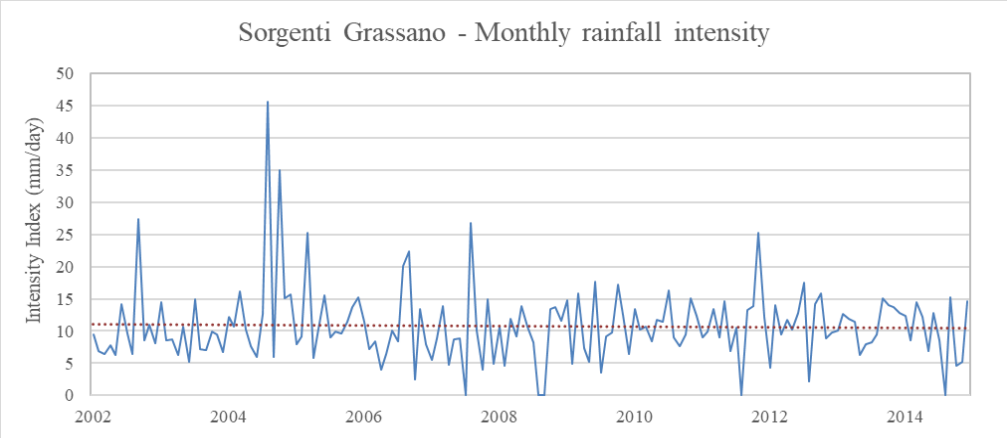

Figure 5.18 Variation of rainfall intensity during the period 2002-2014 for Sorgenti Grassano meteorological station.

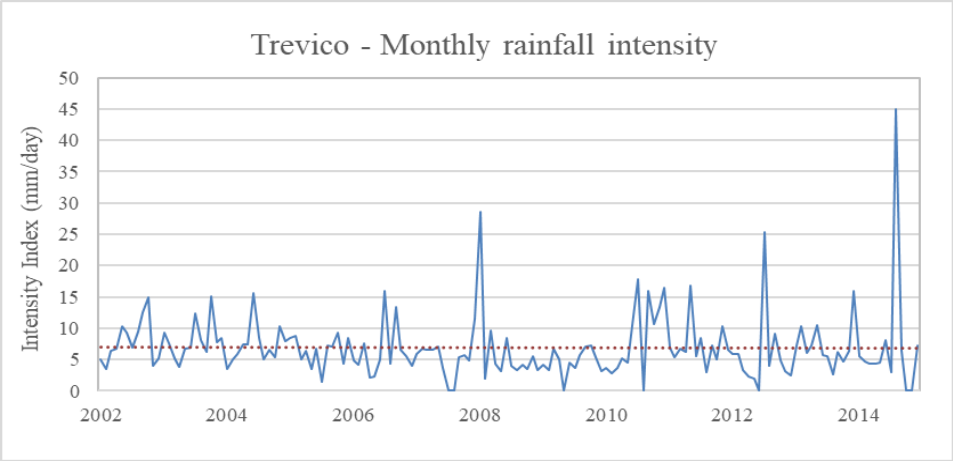

Figure 5.19 Variation of rainfall intensity during the period 2002-2014 for Trevico meteorological station.

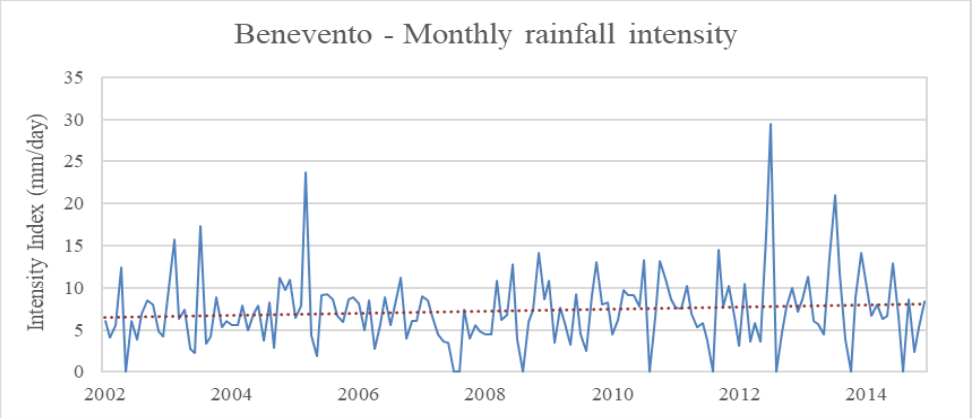

Figure 5.20 Variation of rainfall intensity during the period 2002-2014 for Benevento meteorological station.

The monthly rainfall intensity index varies between 0 to 115 mm/day during the period 1996-2020 in Thermaikos Gulf (Figure 5.21). The daily highest value of the index is mentioned in August of 2019. In addition, a slight increase of the trend line is noted during the monitoring period.

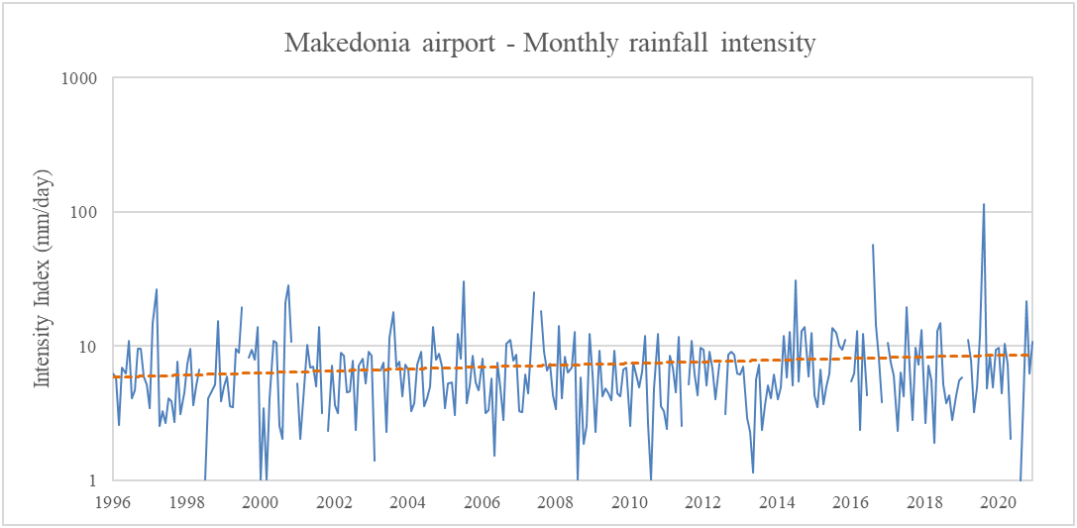

Figure 5.21 Variation of rainfall intensity during the period 1996-2021 for Makedonia Airport meteorological station.

The monthly rainfall intensity index varies between 0 to 34.9 mm/day during the period 2000-2019 in Mouriki basin (Figure 5.22**)**. The monthly highest values of the index are noted in August of 2000 (61 mm in a day) and 2004 (65.2 mm in 2 days), February of 2017 (110 mm in a day) and 2019 (140 mm in 2 days). Other extreme rainfall events are mentioned during the period of 2009-2012. The monthly rainfall intensity index varies between 1 to 32.6 mm/day during the period 2011-2020 in Marathonas basin (Figure 5.23). Many rainfall episodes of high intensity are noted during the monitoring period. Monthly highest value of the index is noted in September 2018 where 23.2 mm of rain were recorded during 2 days of rainfall episode and in 3 days 169.2 mm of total rainfall.

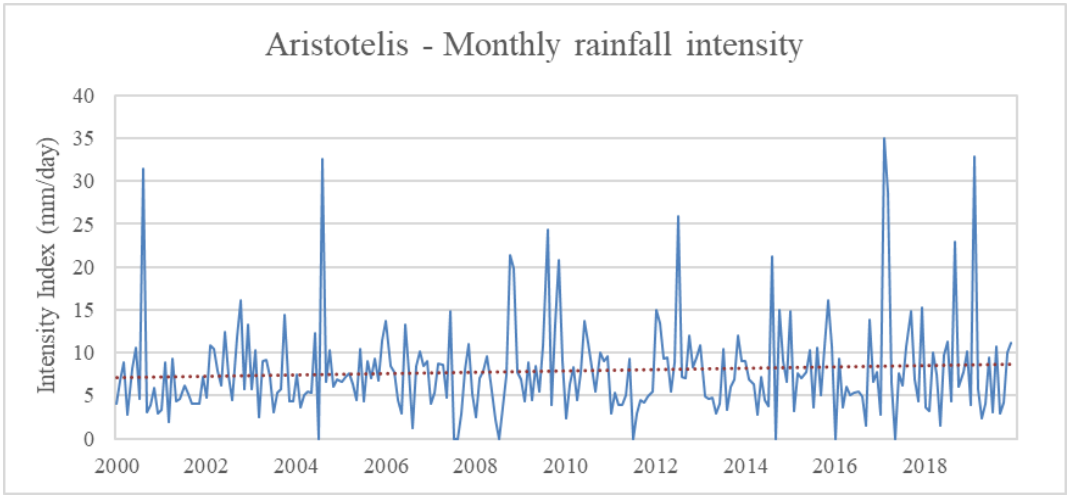

Figure 5.22 Variation of rainfall intensity during the period 2000-2019 for Aristotelis meteorological station.

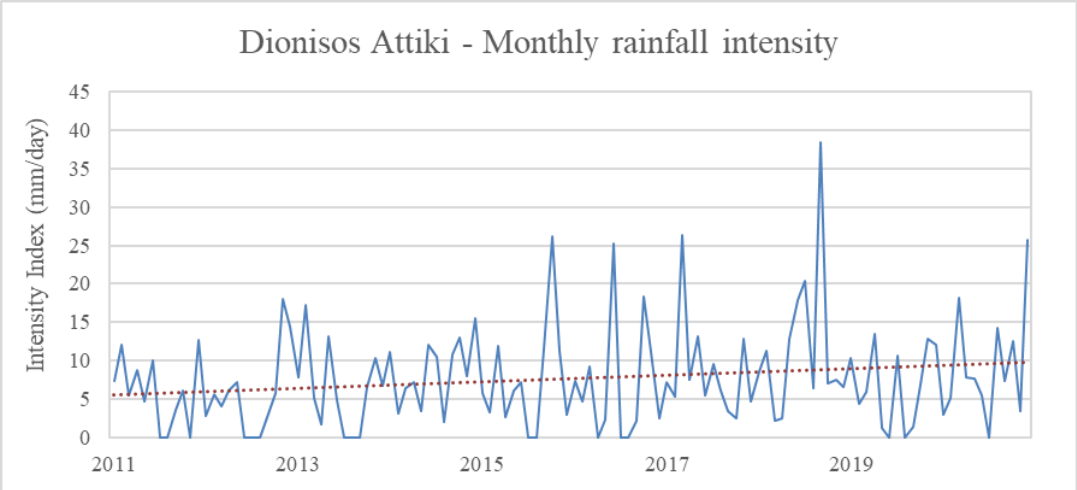

Figure 5.23 Variation of rainfall intensity during the period 2011-2020 for Dionisos Attiki meteorological station.

6 Groundwater Level measurements

In the section are shown the water level measurements in the four case studies as was described in the data network. All data were used for the simulation process. The data are presented in the following tables, while the interpretation is obtained in the corresponding publications.

Table 6.1 Groundwater level measurements in Eastern Thermaikos Gulf and Anthemountas basin.

A_A	Name	Sept_2021	May_2022	Sept_2022	May_2023	Sept_2023
1	1A	29.69	27.44	28.63	28.3	28.42
2	2A	60.29	-	58.87	56.33	59.76
3	3A	18.82	14.8	17.47	16.6	16.9
4	4A	69.35	65.73	68.64	66.2	70.11
5	5A	22.79	8.49	19.64	10.13	9.12
6	6A	17.33	9.2	15.59	11.2	12.06
7	7A	1.81	-	1.86	1.35	2.11
8	8A	15.8	10.62	12.25	9.11	11.68
9	9A	23.35	20.53	22.86	20.07	20.67
10	10A	27.95	25.6	26.44	25.19	26.2
11	11A	29.55	22.71	24.66	26.15	27.41
12	12A	20	18.9	19.4	18.16	20.23
13	13A	19.21	16.46	15.26	14.88	16.03
14	14A	20.9	-	18.8	17.43	20.56
15	15A	-	-	-	57.6	-
16	16A	20.82	17.25	19.88	18.12	20.1
17	17A	18.85	14.62	15.85	14.06	17.13
18	18A	9.44	7.51	8.22	7.46	8.02
19	19A	102.54	102.2	103.56	103.1	102.43
20	20A	67	62.77	65.59	64.2	65.05
21	21A	35.3	35.09	36.27	35.27	35.55
22	22A	32.36	31.18	31.59	30.1	29.17
23	23A	104.98	101.21	101.76	100.41	101.13
24	24A	18.66	15.1	18.41	14.44	14.89
25	25A	8.38	5.4	7.75	5.9	6.22
26	26A	25.37	20.68	22.16	18.72	19.13
27	27A	13.31	8.97	9.76	8.59	9.2
28	28A	21.23	19.33	20.83	20.55	20.87
29	29A	76.87	73.5	74.49	73.12	74.23
30	30A	54.65	49.37	50.12	49.1	50.54
31	31A	30.59	29.83	31.16	30.72	31.06
32	32A	42.54	41.21	42.51	41.89	42.36
33	33A	10.58	9.45	11.1	10.82	11.07
34	34A	3.26	1.04	4.11	3.82	3.2
35	35A	40.2	35.31	37.12	35.66	36.85
36	36A	28.67	37.2	35.67	32.53	32.9
37	37A	12.9	10.74	11.32	10.51	12.11
38	38A	19.9	19.54	20.61	19.38	19.7
39	39A	22.79	17.9	18.9	17.12	18.68
40	40A	25.47	22.1	24.46	23.39	24.32
41	41A	23.04	21.91	22.23	21.42	22.76
42	42A	18.59	18.32	20.16	19.35	20.08
43	43A	15.23	15.37	16.25	15.58	15.92
44	44A	12.2	9.92	10.11	9.62	10.56
45	45A	19.82	12.89	15.39	12.65	12.22

Table 6.2 Groundwater level measurements in Eastern Thermaikos Gulf and Anthemountas basin.

A_A	Name	Sept_2021	May_2022	Sept_2022	May_2023	Sept_2023
46	46A	13.68	12.32	13.19	12.24	12.86
47	47A	20.42	16.33	17.21	16.59	17.1
48	48A	29.52	16.08	20.37	16.12	25.93
49	49A	2.76	1.84	2.41	1.51	2.51
50	50A	7.6	5.1	6.35	5.17	6.22
51	51A	16.23	14.03	15.59	14.88	15.13
52	52A	10.55	8.75	9.12	8.66	9.05
53	53A	5.5	5.29	6.33	5.39	5.57
54	54A	4.8	3.6	4.21	3.38	3.76
55	55A	6.61	5.8	6.12	6	6.53
56	56A	6.83	6.32	6.5	6.11	6.74
57	57A	22.43	15.29	20.86	16.3	19.71
58	58A	13.87	9.7	10.22	8.91	9.2
59	59A	33.92	24.74	25.15	25.82	27.99
60	60A	51.6	39.54	53.32	40.17	40.12
61	61A	45.57	39.4	42.21	40.05	43.36
62	62A	35.7	32.16	34.78	33.14	34.32
63	63A	10.44	-	9.87	7.52	8.61
64	64A	55.79	-	57.31	53.64	53.81
65	65A	21.58	18.91	20.76	19.17	19.62
66	66A	71.59	63.1	65.44	63.38	64.12
67	67A	20.89	20.7	19.95	18.61	19.93
68	68A	79.29	86.86	78.63	77.1	78.32
69	69A	36.07	34.93	36.02	33.64	37.1
70	70A	81.14	80.4	82.54	78.76	81.36
71	71A	23.99	23.71	23.59	23.11	24.03
72	72A	0.52	0.36	0.66	0.29	0.46
73	73A	30.92	30.6	31.08	30.1	30.91
74	74A	15.32	15.64	15.83	15.61	15.99
75	75A	29.48	28.94	29.76	28.3	29.52
76	76A	11.77	11.2	11.81	11	11.8
77	77A	8.29	7.55	7.63	6.7	7.86
78	78A	12.14	10.3	11.41	10.06	11.22
79	79A	128.75	125.34	124	124.47	-
80	80A	37.1	35.95	36.62	35.88	36.3
81	81A	59.94	57.48	58.8	56.94	57.57
82	82A	28.9	28.6	39.72	25.96	33.8
83	83A	51.69	-	60.84	52.46	54.2
84	84A	82.06	80.97	82.3	80.96	81.44
85	85A	6.12	5.97	6.09	-	6.08
86	86A	34.48	-	33.6	34.11	33.39
87	87A	-	-	28.12	18.67	18.69
88	88A	-	16.89	-	17.33	24.16
89	89A	12	9.5	12.88	9.52	11.59
90	90A	66.97	-	-	58.34	66.31
91	91A	101.1	96.88	93.54	90	-
92	92A	74.1	76.9	73.7	73.57	75.51
93	93A	27.9	15.73	-	-	25.75
94	94A	12.2	9.92	11.1	9.78	10.66
95	95A	10.82	8.62	9.55	8.49	9.23

Table 6.3 Monthly Groundwater level measurements in Eastern Thermaikos Gulf and Anthemountas basin.

A/A	Borehole	Sep-21	Oct-21	Nov-21	Dec-21	Jan-22	Feb-22	Mar-22	Apr-22	May-22	Jun-22	Jul-22	Aug-22	Sep-22
		Depth	Depth	Depth	Depth	Depth	Depth	Depth	Depth	Depth	Depth	Depth	Depth	Depth
1	1A	29.69	27.97	27.43	27.1	26.8	26.54	26.33	26.84	27.44	28.07	28.65	28.67	28.63
2	3A	18.82	17.07	15.38	14.69	14.44	14.52	14.56	14.67	14.8	21.87	-	18.33	17.47
3	25A	8.38	7.98	7.55	7.11	6.55	6.27	6.2	5.82	5.4	7.25	5.86	6.12	7.75
4	22A	32.36	32.83	32.28	32.12	32.06	31.9	31.55	31.37	31.18	28.25	31.28	31.33	31.59
5	12A	20	19.17	21.48	20.2	18.9	18.3	18.22	18.55	18.9	18.8	19.37	19.42	19.4
6	13A	19.21	18.27	18.19	17.81	17.68	17.55	17.34	17.4	16.46	18.27	16.13	16.47	15.26
7	14A	20.9	19.13	-	19.1	19.06	18.92	18.88	18.7	-	-	-	19.6	18.8
8	17A	18.85	17.99	17.19	16.7	16.27	15.94	15.31	15.1	14.62	17.32	15.85	15.91	15.85
9	67A	20.89	20.81	20.88	20.75	20.7	20.78	20.66	20.74	20.7	20.8	21.83	20.6	19.95
10	68A	79.29	78.75	78.23	78.16	78.1	80.88	85.07	86.1	86.86	79.33	80.29	79.9	78.63
11	69A	36.07	35.1	34.42	34.18	33.62	33.42	33.55	33.7	34.93	36.57	37.12	36.86	36.02
12	70A	81.14	80.09	84.01	83.77	83.4	82.54	82.11	81.69	80.4	83.31	83.31	83.1	82.54
13	15A	-	-	59.8	60.04	60.12	60.23	60.37	60.55	-	-	-	60.61	-
14	73A	30.92	30.67	30.49	30.52	30.75	30.61	30.58	30.62	30.6	31.76	30.88	30.96	31.08
15	75A	29.48	29.2	28.74	28.54	28.36	28.3	28.16	28.37	28.94	31.13	30.14	30.1	29.76
16	91A	101.1	103.74	93.52	92.84	92.1	94.1	94.26	95.57	96.88	93.63	93.72	93.66	93.54
17	92A	74.1	73.28	73.04	73.1	72.8	73.33	73.86	75.24	76.9	74.59	74.61	74.14	73.7
18	77A	8.29	7.36	-	7.11	7.4	7.35	7.46	7.42	7.55	7.7	7.98	7.82	7.63
19	64A	55.79	54.48	-	53.51	53.1	51.82	51.4	50.92	-	-	59.41	58.8	57.31
20	86A	34.48	33.8	31.18	31.07	30.72	31.15	31.76	31.5	-	-	35.51	34.75	33.6
21	2A	60.29	58.48	51.44	50.87	50.31	51.67	52.2	52.84	-	-	59.78	59.12	58.87
22	84A	82.06	81.75	81.56	81.16	80.94	81.07	81.42	81.19	80.97	81.3	81.26	81.74	82.3
23	83A	51.69	50.26	50.16	50.1	50.02	48.85	48.34	47.72	-	47.37	55	57.63	60.84
24	82A	28.9	24.78	18.56	20.14	22.05	25.45	26.13	27.7	28.6	22.01	45	42.67	39.72
25	90A	-	-	79.59	75.23	-	70.16	66.45	58.27	-	-	-	89.64	-
26	45A	19.82	16.12	13.86	13.23	12.89	13.11	13.28	13.06	12.89	17.4	19.91	20.12	15.39
27	7A	1.81	1.6	1.53	1.56	1.48	1.55	1.67	1.71	-	1.72	2.02	1.9	1.86
28	89A	12	12.01	10.75	10.22	10	9.73	9.55	10.64	9.5	11.75	13.63	13.12	12.88
29	49A	2.76	1.95	1.59	1.63	1.7	1.64	1.57	1.77	1.84	2.31	2.18	2.29	2.41
30	53A	5.5	6.03	5.58	5.53	5.5	5.48	5.42	5.37	5.29	5.51	5.8	6.14	6.33
31	61A	45.57	45.5	41.45	40.22	38.93	38.12	38.26	38.33	39.4	46.2	45.38	45.2	45.15
32	79A	128.75	128.75	-	127.72	126.75	126.14	125.58	125.49	125.34	129.55	126.8	125.36	124
33	87A	-	20.08	19.35	18.84	17.99	16.84	16.22	16.73	-	-	-	-	28.12
34	60A	51.6	46.36	43.39	42.22	41.16	40.24	40.16	40	39.54	50.43	-	55.46	53.32
35	88A	-	23.6	-	22.23	-	20.74	20.33	18.42	16.89	22.08	20.1	24.38	-

Table 6.4 Monthly Groundwater level measurements in Eastern Thermaikos Gulf and Anthemountas basin.

A/A	Borehole	Oct-22	Nov-22	Dec-22	Jan-23	Feb-23	Mar-23	Apr-23	May-23	Jun-23	Jul-23	Aug-23	Sep-23
		Depth	Depth	Depth	Depth	Depth	Depth	Depth	Depth	Depth	Depth	Depth	Depth
1	1A	28.24	28.09	27.7	27.51	27.1	27.22	28.06	28.3	28.38	28.5	28.41	28.42
2	3A	16.62	15.88	15.16	15.31	15.46	15.4	16.03	16.6	16.52	16.44	16.75	16.9
3	25A	7.2	6.87	5.91	6.05	6.11	5.9	6.09	5.96	6.61	6.48	6.3	6.22
4	22A	31.2	31.03	30.87	30.45	30.32	30.1	29.48	29.67	30.3	30.34	30.28	29.17
5	12A	19.52	19.3	18.56	18.3	18.14	18.16	18.37	19.93	20.1	20.16	20.25	20.23
6	13A	15.1	15.06	14.94	14.9	14.92	14.88	14.91	14.96	15.12	15.65	15.88	16.03
7	14A	19.1	19.18	18.86	17.62	17.55	17.43	17.2	17.08	17.25	18.36	18.48	20.56
8	17A	15.8	15.57	15.23	15.1	14.87	14.06	14.1	14.98	16.6	16.22	16.36	17.13
9	67A	20.22	20.14	19.07	18.8	18.76	18.61	18.93	19.1	20.3	20.53	20.19	19.93
10	68A	78.97	79.22	79.1	78.66	77.5	77.1	77.22	78.03	78.4	78.62	79.2	78.32
11	69A	35.5	35.2	34.76	34.12	33.8	33.64	35.8	34.14	34.4	35.65	36.98	37.1
12	70A	81.83	80.49	79.64	79.21	78.88	78.76	79.5	80.1	80.22	80.65	81.1	81.36
13	15A	59.94	59.12	-	58.87	58.1	57.6	58.23	57.6	60.55	60.84	60.9	-
14	73A	30.87	30.71	30.68	30.47	30.33	30.1	29.7	28.53	29.43	29.66	30.12	30.91
15	75A	29.57	29.06	28.57	28.42	28.37	28.3	28.29	28.36	29.22	30.07	29.86	29.52
16	91A	92.54	92.2	91.8	90.57	90.14	90	90.63	91.2	93.84	93.6	92.75	-
17	92A	74.36	74.91	75.2	74.48	74.12	73.57	74.63	75.82	76.69	77.5	76.68	75.51
18	77A	7.58	7.46	7.4	7.07	6.84	6.7	7.1	7.45	7.92	8.47	8	7.86
19	64A	56.36	55.78	-	54.26	54.1	53.64	52.15	50.61	51.23	52.72	53.26	53.81
20	86A	33.42	32.1	32.59	33.37	33.91	34.11	3.36	33.24	34.44	34.92	33.7	33.39
21	2A	58.44	58.1	57.83	57.49	56.77	56.53	57.21	-	58.45	58.92	59.37	59.76
22	84A	81.71	81.86	81.77	81.42	81.11	80.96	80.54	80.6	81.63	82.75	82.3	81.44
23	83A	58.47	53.36	52.83	52.61	52.59	52.46	52.1	51.74	51.94	52.31	52.46	52.2
24	82A	40.19	40.24	40.21	30.34	26.51	25.96	24.67	24	45.95	44.12	40.39	33.8
25	90A	82.45	80.37	77.22	65.73	60.17	58.34	51.92	-	60.57	66.12	70.73	66.31
26	45A	14.59	14.21	13.38	13.06	12.87	12.65	11.72	11.4	25.45	22.62	18.39	12.22
27	7A	1.72	1.65	1.54	1.42	1.38	1.35	1.4	1.46	1.77	1.92	2.03	2.11
28	89A	12.11	11.2	10.66	10.24	9.68	9.52	9.36	9.43	10.16	10.72	11.2	11.59
29	49A	2.11	1.95	1.69	1.52	1.44	1.51	1.54	1.63	1.85	1.92	2.27	2.51
30	53A	6.02	5.94	5.77	5.54	5.46	5.39	5.35	5.31	6.26	6.32	5.73	5.57
31	61A	44.44	44.1	43.72	43.12	42.73	41.11	41.1	40.7	42.22	42.57	42.36	41.36
32	79A	123.72	123.16	-	120.55	120.47	124.47	129.99	130.16	133.36	134.02	134.59	-
33	87A	22.18	21.03	-	20.12	20.04	18.67	17.72	15.9	16.39	17.42	18.12	18.69
34	60A	50.12	48.88	42.1	41.88	41.53	40.17	39.94	38.62	40.22	42.61	41.15	40.12
35	88A	22.14	21.3	20.64	20.45	20.19	17.33	16.97	17.52	18.44	20.39	23.43	24.16

Table 6.5 Groundwater level (depth from the surface) measurements in Marathonas basin.

A/A	NAME	Sep-21	May-22
1	GA 1	6.3	6
2	GA 2	7.4	6.79
3	GA 3	6.78	6.22
4	GA 4	4.45	3.8
5	GA 6	4.15	3.78
6	GA 11	2.7	2.25
7	GA 12	1.92	1.45
8	GA 13	10.17	9.45
9	GA 14	7.3	7.2
10	GA 15	3.34	3.2
11	GA 16	3.6	3.5
12	GA 17	2.12	1.85
13	GA 18	14.3	13.3
14	GA 19	12.4	11.3
15	GA 20	11.2	10.6
16	GA 21	22.3	21
17	GA 22	8.47	8.12
18	GA 23	2.5	1.91
19	GA 24	8.6	8.20
20	GA 25	6.56	6.20

Table 6.6 Groundwater level (depth from the surface) measurements in Campania basin.

A/A	Well	Septeber 2021	May-22
1	C1	100	99
2	C2	150.5	150.2
3	C3	101.86	98.13
4	C4	75	74.19
5	C5	65	64
6	C6	85.64	80.21
7	C7	91	90.66
8	C8	79.5	79.31
9	C9	44.14	42.63
10	C10	38.5	39
11	C11	44	44.5
12	C12	32.58	28.81
13	C13	29	29.5
14	C14	31	30.12
15	C15	35	33.49

Table 6.7 Groundwater level (depth from the surface) measurements in Mouriki basin.

A/A	NAME	Sep-21	May-22	Sep-22	May-23
		Depth	Depth	Depth	Depth
1	1V	36.62	21.8	-	25.51
2	2V	0	0	0.11	0
3	3V	1.06	0.69	0.95	-
4	4V	1.13	3.38	4.32	2.99
5	5V	16.64	15.03	7.3	5.94
6	6V	2.11	1.4	1.35	-
7	7V	13.64	11.67	12.8	11.21
8	8V	5.2	4.78	5.84	6.12
9	9V	0	0	0	-
10	10V	8.93	8.5	5.85	-
11	11V	14.85	12.73	-	12.15
12	12V	12.36	9.47	8.64	7.25
13	13V	11.55	3.82	9.08	5.16
14	14V	15.59	14.3	11.72	10.57
15	15V	16.07	11.68	12.97	10.81
16	16V	17.96	11.72	14.88	12.33
17	17V	16.78	12.23	10.54	11.67
18	18V	-	0	3.55	0
19	19V	9.91	2.48	6.05	4.44
20	20V	13.18	3.51	11	8.65
21	21V	12.85	11.7	5.85	7.36
22	22V	-	0	13.67	0
23	23V	14.13	9.03	12.03	8.88
24	24V	15.07	11.32	9.3	8.21
25	25V	2.72	1.1	2.2	1.29
26	26V	-	1.55	-	1.48
27	27V	6.5	0.82	1.34	1.11
28	28V	0	0	0	0
29	29V	-	1.25	-	1.36
30	30V	-	2.46	-	2.32

7 Geoelectrical measurements

The science of geophysics, through the method of electrical tomography (ERT), provides direct or indirect information about the nature of the material in which rainwater flows on the surface, penetrates or moves internally. Electrical tomography is a method in which, by injecting current pulses to the ground surface and recording the electrical potential imprinted on the surface, a two-dimensional image is extracted that quantitatively presents the electrical resistivity values of underground formations. These two-dimensional images can show the electrical resistivity regime up to 300 meters deep, but the deeper the scan depth, the lower the resolution of the scans. The shallow scan depth provides greater resolution and a more detailed presentation of formations and coexisting phases in the subsurface. A typical example is that a dry soil will have a high value of electrical resistivity while a soil saturated with water will have a low value of electrical resistivity since the water with the ions it contains is a good conductor of electricity and facilitates the flow of current. A wide variety of applications are based on this method, indicatively three works with an agricultural orientation are referred here, Michot et al. 2003, Corwin & Lesch 2005 and Brillante et al. 2015. The direct information provided by electrical tomography is the value of the electrical resistivity of the subsoil. The infiltration rate, redistribution duration, and drainage duration in the vadose zone following a rainfall event, as described by Arrey et al. 2019, they are important features of a geological formation and very useful information in achieving the goals related to dealing with the problems of intense surface runoff of rainfall, the reduction of water reserves and the enrichment of aquifers. The determination of the above values could be approximated numerically, as indirect information, from the electrical tomography values. However, it is not enough to record a tomography, but long-term monitoring of changes in the electrical resistivity of the subsoil is needed, which will record the changes caused by the infiltration of rainwater into the deeper stratigraphic horizons.

But there are two main factors that change the resistivities in long-term monitoring of subsurface electrical resistivities. The first factor that affects the electrical resistivities is the change in soil water content and the second factor is the changes in soil temperature that fluctuating both on a daily basis with a depth of influence of 30 centimeters and on a seasonal basis with a depth of influence of about 5 meters. As a result, it is important to first remove the effect of temperature from the

recorded electrical resistivities, so that only the effect of water changes remains as a residual.

In the present work the electrically mapped area is presented in an Electrical Resistivity Tomography (ERT) with dimensions of 9.2 m on the horizontal axis and 1.8 m on the vertical depth axis. In Marathonas (Figure 7.1), Mouriki (Figure 7.2) and Campania (Figure 7.3) site the measurements obtained twice per month. However, the provided ERT contributed only to the conceptual model of the site. A critical conclusion of this research regarding the geophysical research was that ERT focused on vadose zone hydrology study require high frequency tomographies and measurement of vadose zone temperature. Within this project we achieved to collect such data, which might be a unique data set worldwide. The measurements obtained in N. Rysio site which is part of Eastern Thermaikos Gulf.

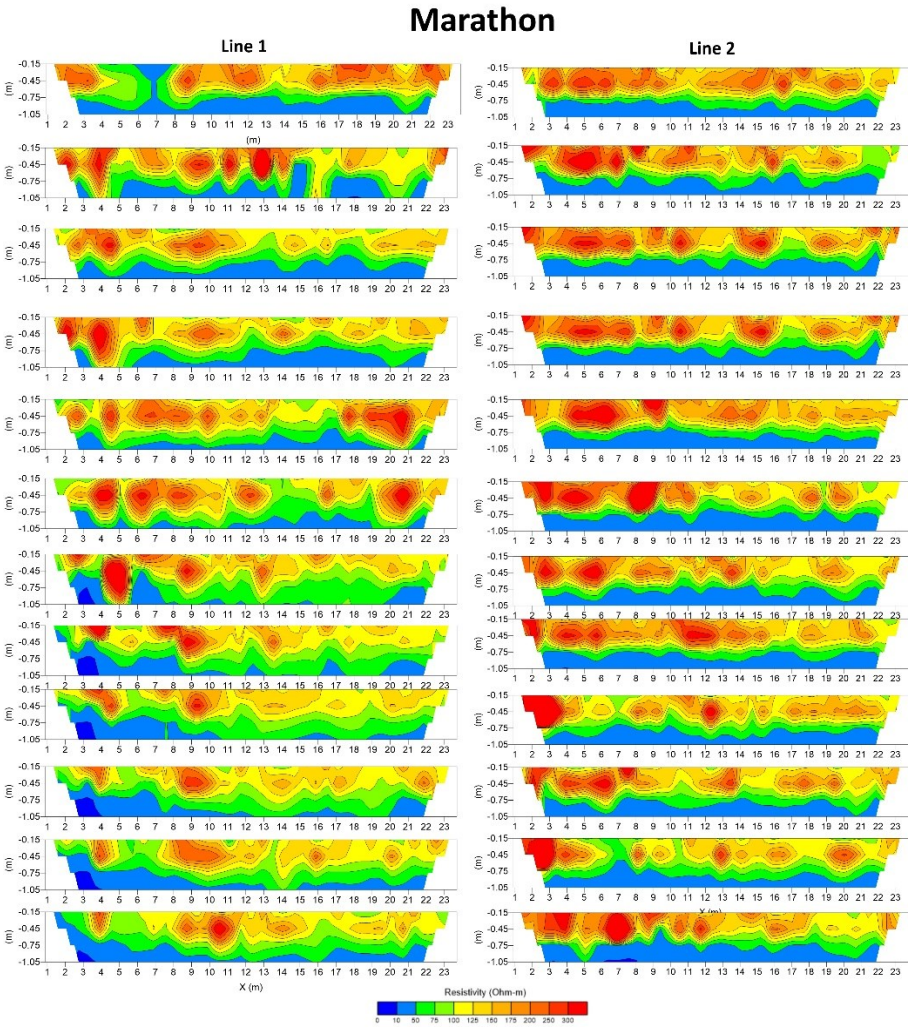

Figure 7.1 ERT measurements in Marathonas basin.

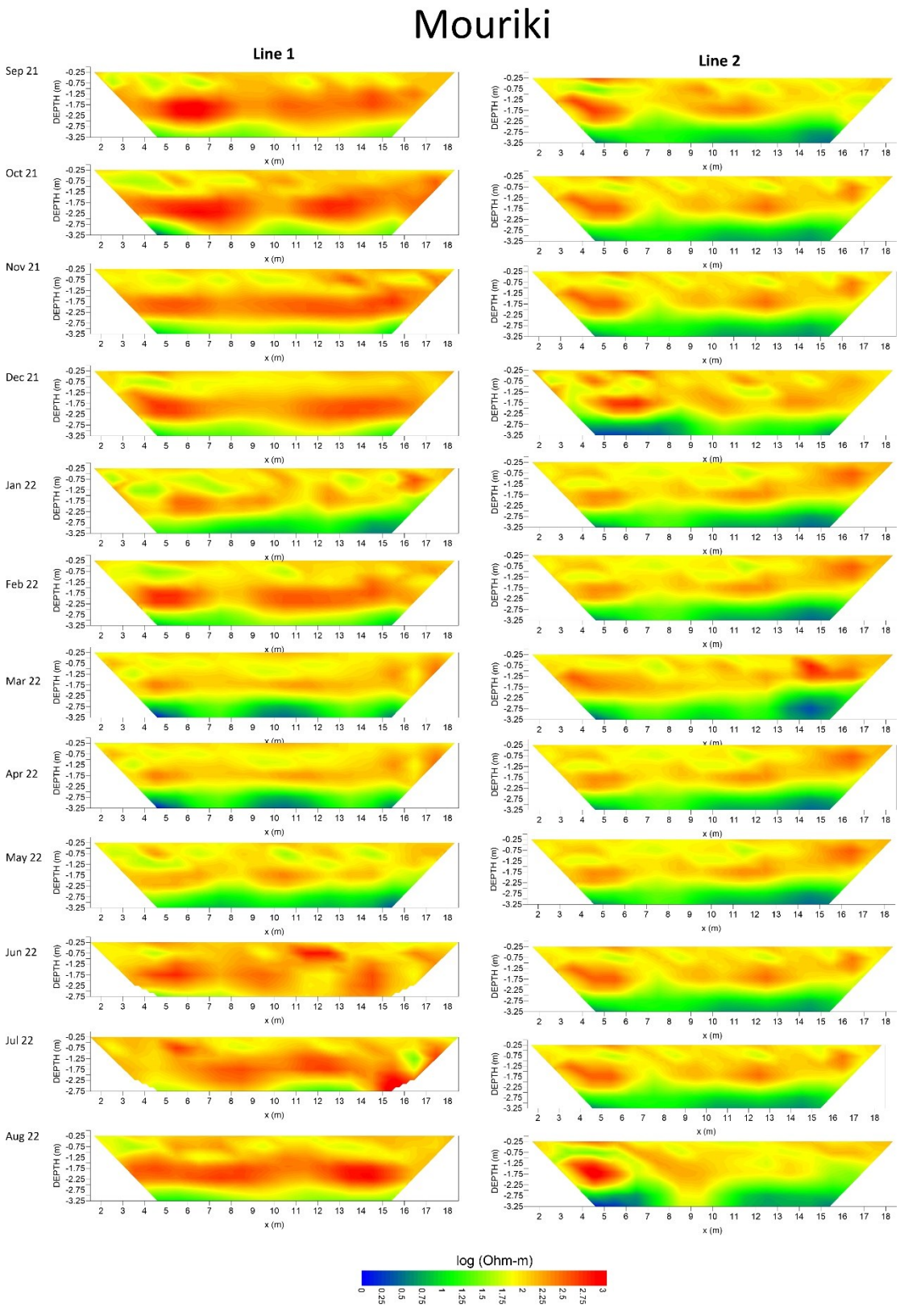

Figure 7.2ERT measurements in Mouriki basin.

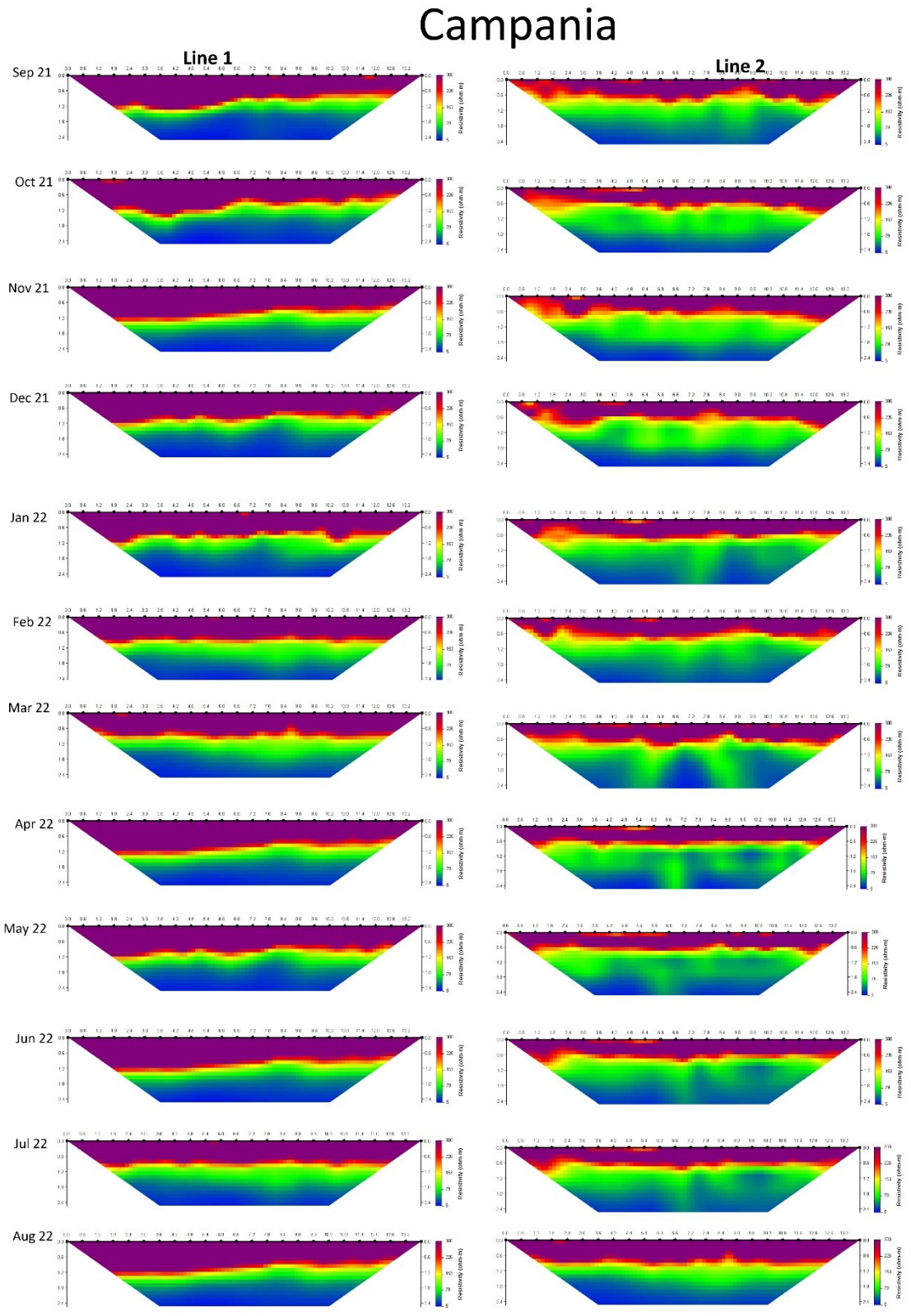

Figure 7.3 ERT measurements in Campania region.

7.1 Site of the monitoring - Instruments

In N.Rysio established the high frequency monitoring station using ERT (Figure 7.4, Figure 7.5). The monitoring instruments consist of a weather station, a soil electrical resistivity measuring device and a probe that measures the temperature and the water content of soil up to -0.85 [m].

The weather sensitive instruments were placed in a waterproof box also called acquisition station. These instruments consist of a) The end of a power supply line from the nearby building to supply power to the instruments, b) The main unit of the weather station, c) A mobile phone device for the supply of internet connection to the weather station and the transition of the weather station collected data to an online data storage provider and d) The electrical resistivity measuring device, the cable multiplexer, a 12v battery and a 12v battery charger. The probe as a weather resistant and self-powered instrument was inserted in the soil in a selected point of the investigation site.

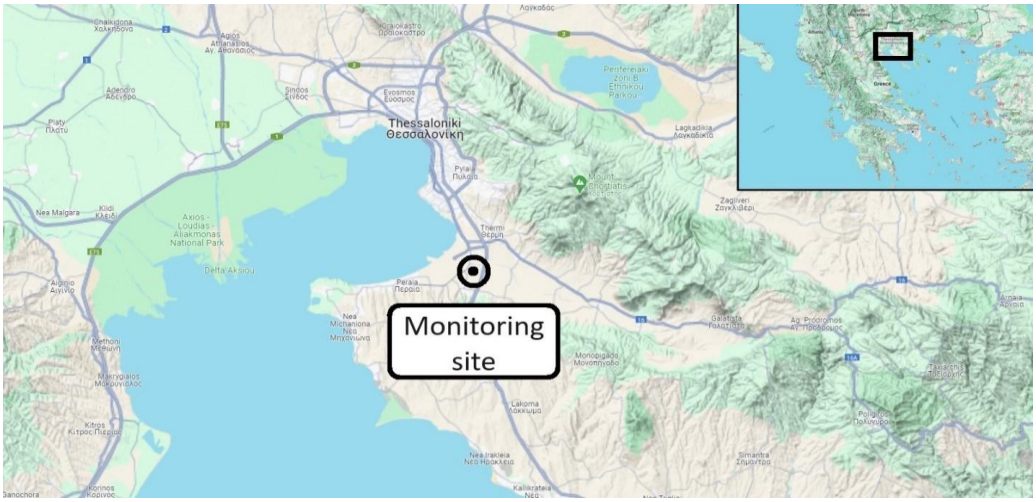

Figure 7.4 Monitoring site.

7.1.1 Weather station and air conditions monitoring

The meteorological and climatological conditions were recorded with the Bresser "Weather Station". The weather station sensors (remotely connected to the main unit in the water resistance box) were placed at the investigation site at a height of +1.6 meters from the ground level and includes sensors for air temperature [°C], precipitation [mm], wind (direction in degrees° and speed [m/s]), air humidity [%] and air pressure [hPa].

7.1.2 Soil electrical resistivity monitoring

For the calculation of the subsurface electrical resistivities [ohm·m] and the supply of electrical resistivity tomographies (ERT) it was necessary the acquisition of soil apparent resistivity measurements [ohm·m]. These measurements were performed with the Lippmann "Earth resistivity meter 4point light 10W" and the placement of 24 electrodes with 0.4 [m] electrode spacing at the site connected by cables with the multiplexer located inside the water resistance box.

7.1.3 Soil temperate and soil water content monitoring probe

The Sentek “Drill & Drop Bluetooth probe” was used to acquire subsurface temperature and water content data, with sensors at -0.05, -0.15, -0.25, -0.35, -0, 45, -0.55, - 0.65, -0.75 and -0.85 [m] depth.

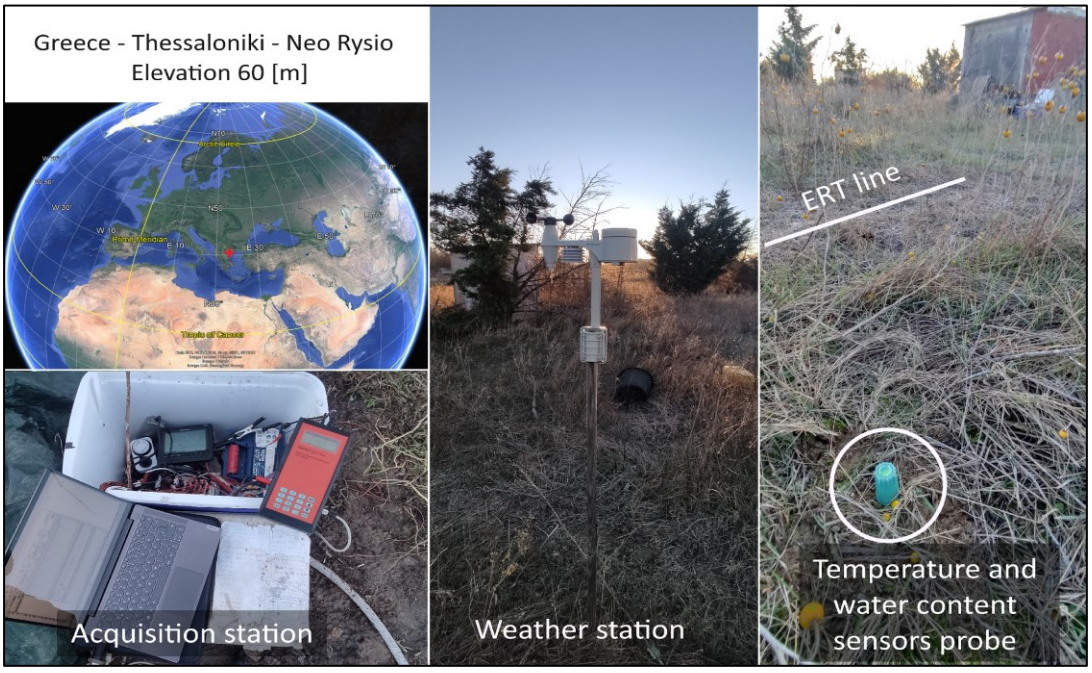

Figure 7.5 Monitoring site data acquisition instruments.

7.2 Methodology

The procedure followed during the monitoring period can be divided into two sections, first the acquisition of the data and second the processing of data. Various codes were written in Matlab environment to manage and process the data. Each of these codes involved processing either each of the recorded variables or was used to jointly process variables that correlated with each other and produced useful results. By the time this report is written, the monitoring is still in process.

7.2.1 Weather station data

The weather station data were collected with 10 minutes' measurement interval. All data were sent to an online data base and then were download and processed. In Figure 7.6 presented below, consist of 116.050 dated data points for each one of the 12 exported data sets of the weather station.

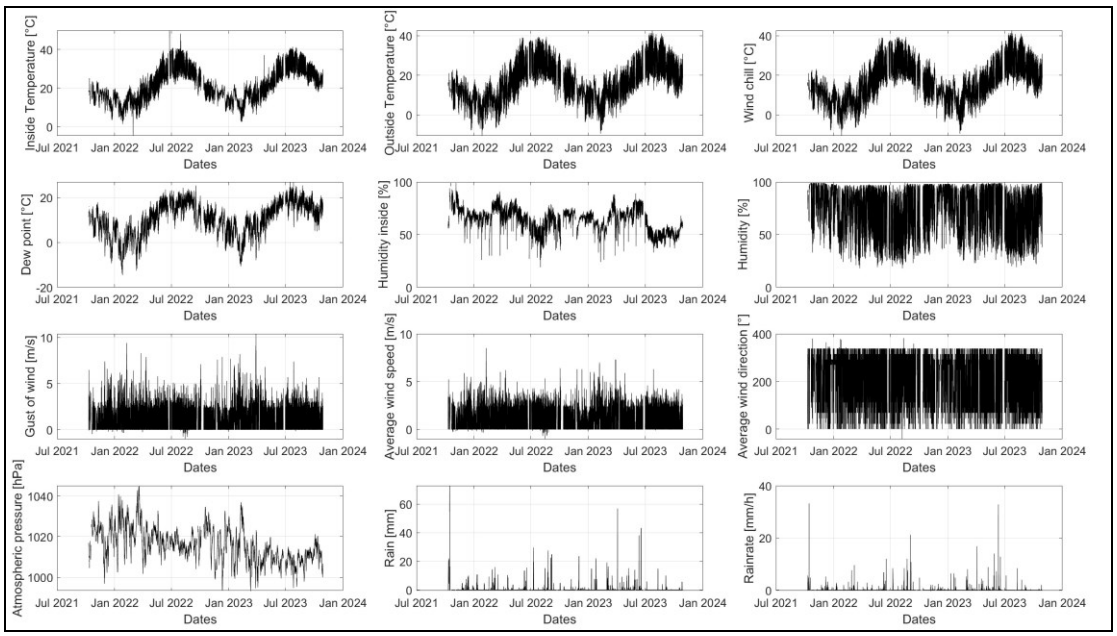

Figure 7.6 Recorded data from the weather station.

7.2.2 Temperature and water content probe

The probe measurements of temperature and water content had a measurement interval of 30 minutes. The data were collected weekly in situ and converted from electrical signals to temperature and soil water content values. Figure 7.7 and Figure 7.8 present the temperature and the water content recorded data. The data consist of 39.024 dated data for each one of the 18 sensor of the probe (700.000 dated data total).

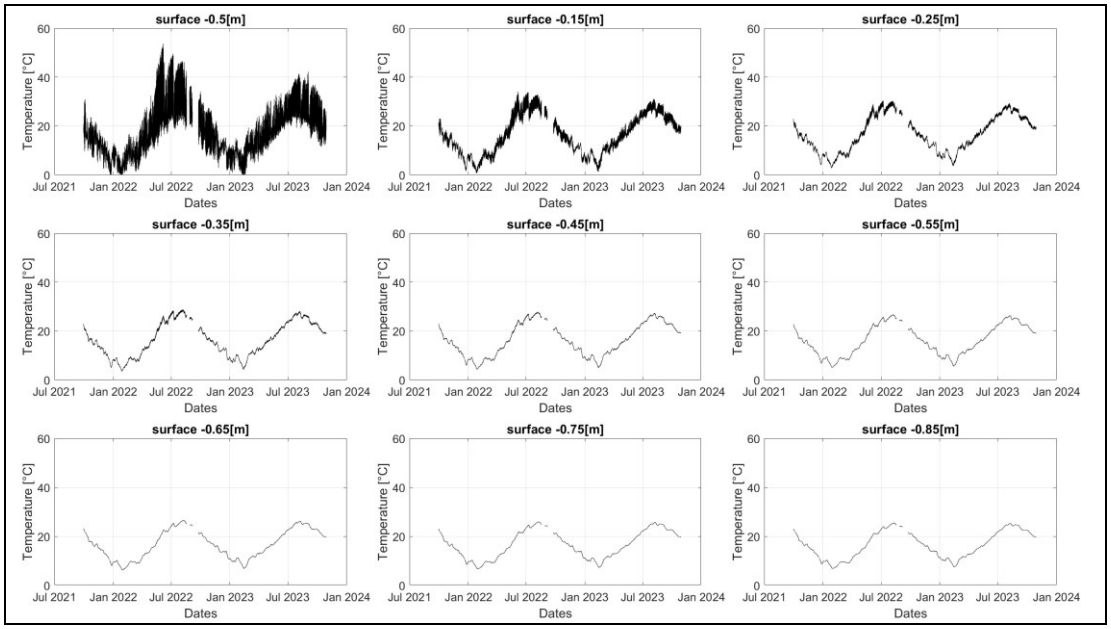

Figure 7.7 Temperature data recorded from the drill and drop probe.

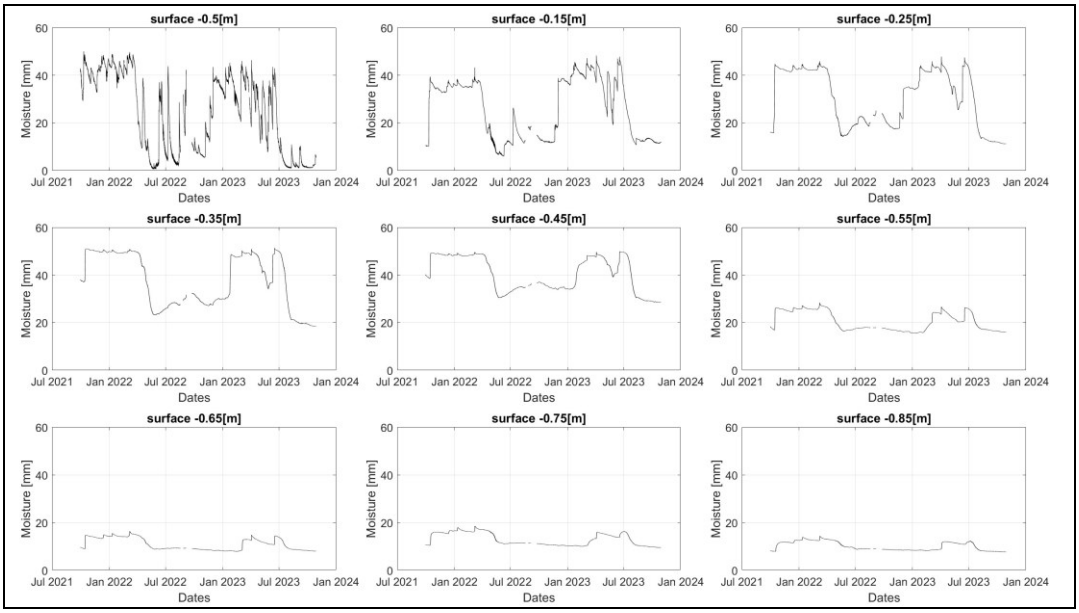

Figure 7.8 Water content data recorded from the drill and drop probe.

7.2.3 Electrical resistivity tomography

Mapping the underground electrical resistivities and producing the electrical resistivity tomographies (ERT) consist of two steps. First step is the acquisition of apparent electrical resistivity data, and the second step is the inversion process that produce the actual electrical resistivity tomographies. The apparent electrical resistivity measurements are based on 4 electrodes (Figure 7.9) which are inserted into the soil surface on a straight line and in specific distances between them and

finally are connected with cables to a resistivity meter device. Two of the electrodes (Figure 7.9, A and B electrodes) are injecting electric current into the ground and create an electric field which is shaped according to the materials of the ground. The other two electrodes (Figure 7.9, M and N electrodes) are recording the potential difference at their positions as a result of the previously produced electric field. The apparent electrical resistivity value for this measurement is then calculated based on the measured potential difference and the geometrical factor (which is calculated based on the distance between electrodes).

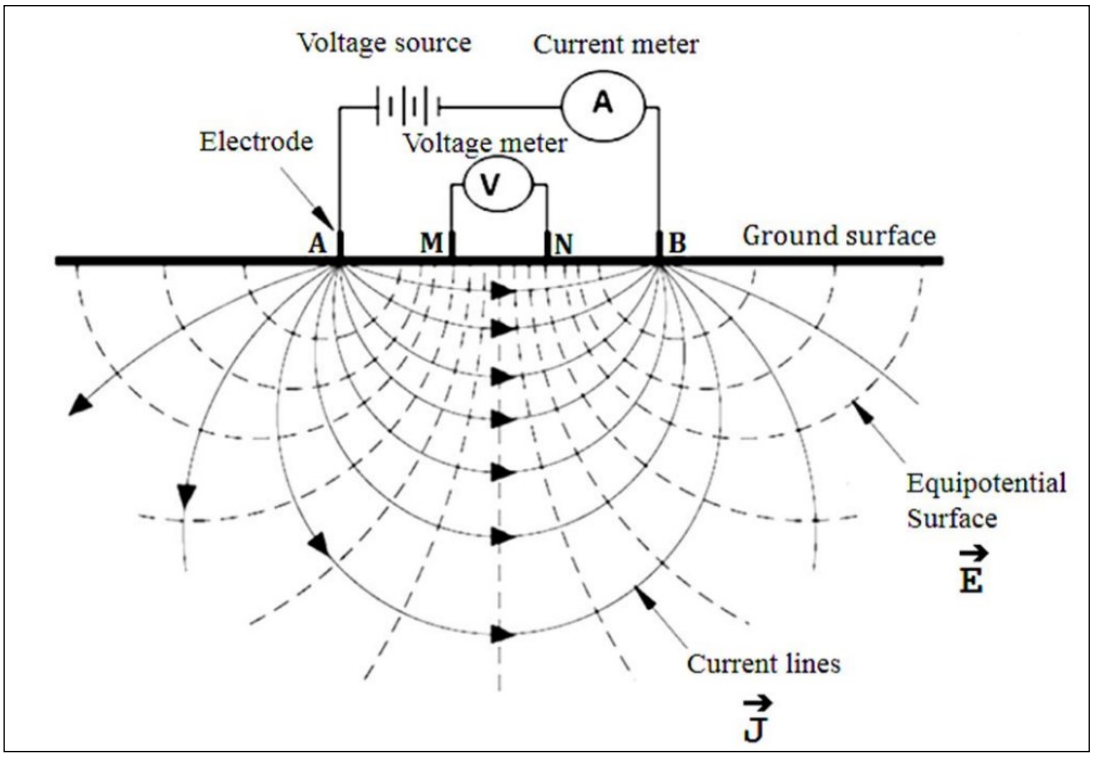

Figure 7.9 ERT method, surface potential difference measurements (from Muchingami et al., 2012).

Automated multielectrode resistivity meter device with more than 4 electrodes placed in line, can produce a large number of measurements based on acquisition protocols that are using different quadrats of electrodes for each measurement. Each of these measurement represents the apparent electrical resistivity of a pseudo position in the ground which is based on the position of the electrodes, as it is presented on the Figure 7.10, with electrode stations 1, 2 and 3 and their corresponding red dots.

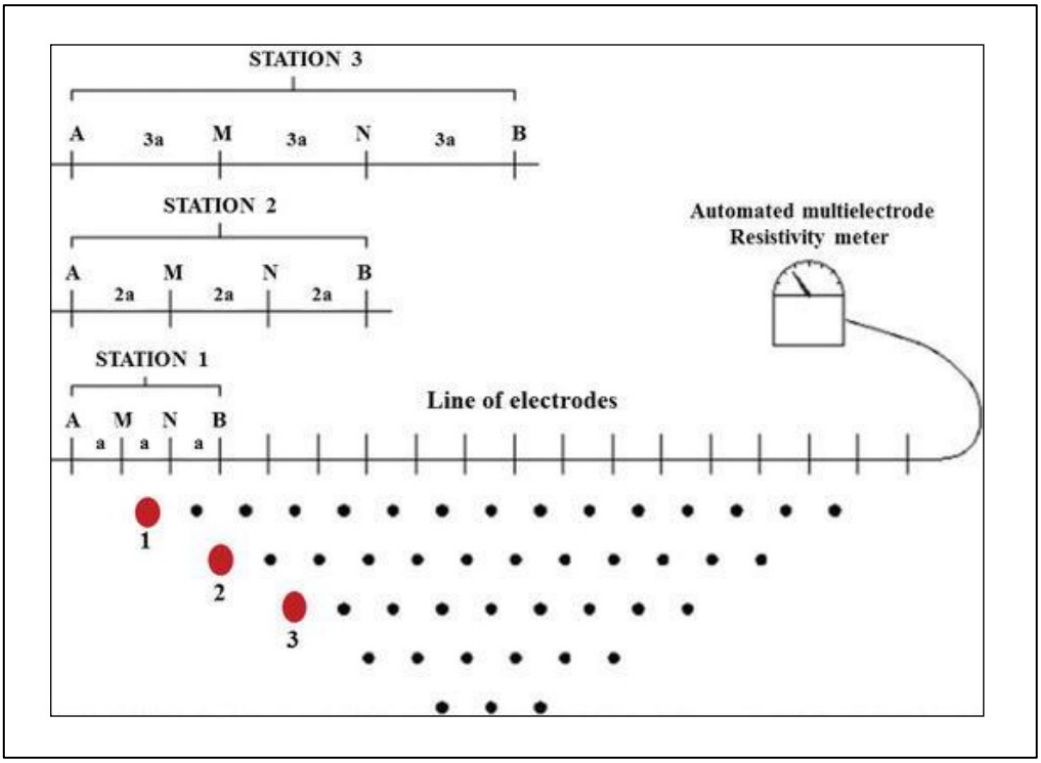

Figure 7.10 ERT measurements (from Matias & Almeida, 2017).

In this study were used 24 electrodes with 2 acquisition protocols, a Dipole-Dipole protocol with 195 stations (measurements) and a Multiple Gradient protocol with 442 stations (measurements). A data set, consisting of those 2 protocols, was measured with 4-hours interval, providing until now 4.757 dated apparent resistivity data sets (more than 3 million measurements). The data were collected weekly in situ, and transferred from the electrical resistivity device to the laptop's memory.

Next, each apparent electrical resistivity measured protocol of the data sets, was inverted with Geotomo "res2Dinv" software using standard constraints on the data and the model. Inversion process is based on a 2.5D finite-element routine that solve the forward resistivity problem with an iterative least-squares algorithm with active constrain balancing for the reconstruction of the actual subsurface resistivity model. The actual subsurface resistivity model (also called the inversion results or the electrical resistivity tomography (ERT)), is a 2-dimension grid of parameters Figure 7.11 A. Each parameter corresponds to a horizontal position in meters, a vertical position in meters and has a value of electrical resistivity in ohm·m. The electrical resistivity values (as a result of the inversion process) are presented in a graph like the one in Figure 7.11 B.

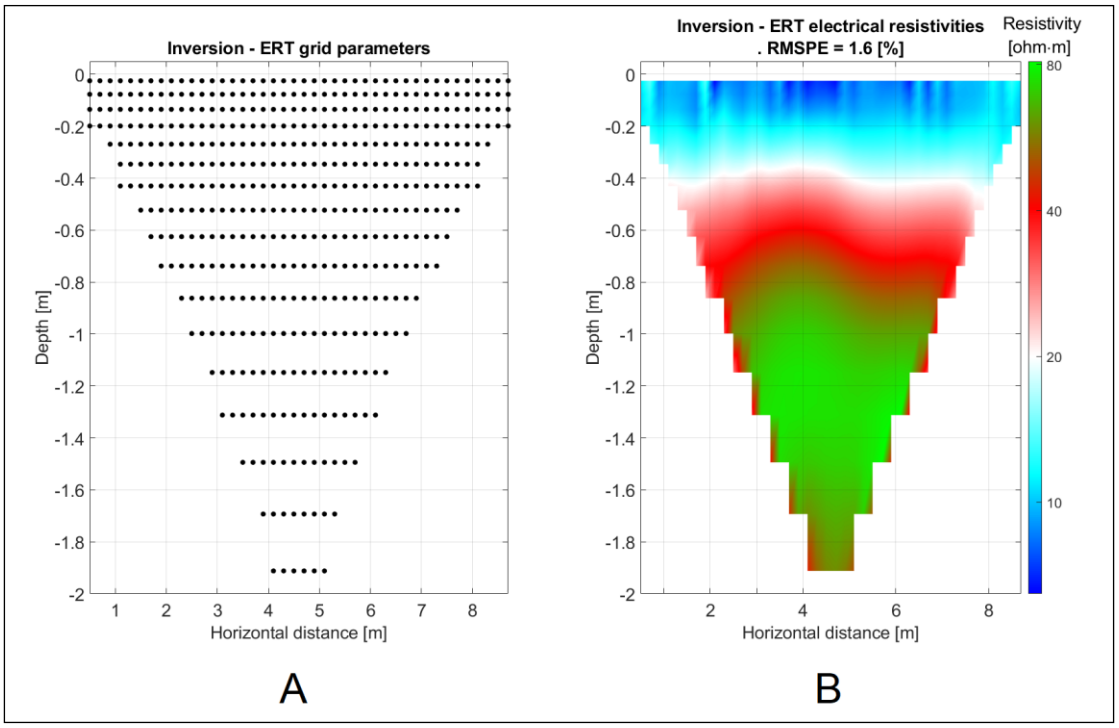

Figure 7.11 ERT grid parameters A, ERT inversion results B.

7.2.4 Temperature compensation of the inversion results

The produced electrical resistivity tomographies were corrected from the effect of the subsurface temperature fluctuations, providing the temperature compensated ERTs. The temperature compensation was based on Keller and Frischknecht temperature compensation model (Ma et al. 2010), as well as the measured soil temperatures that firstly interpolated into the depths of the ERT and then smoothed with a 24 hour moving mean algorithm for the correction of electrical resistivities based on the seasonal and not the diurnal temperature fluctuations. Figure 7.12 present an example for the -0.26 [m] depth, off the initial invention results (with black color), and the temperature compensated inversion results to 25 [°C] (magenta color) that provide the ability to present electrical resistivities as a result of only the effect of water penetration, free from the effect of temperature fluctuations.

In Figure 7.12 it can be seen that any interpretation made on the temperature uncorrected electrical resistivities (black color line) regarding the presence and effect of water would lead to incorrect results. In contrast, the temperature-corrected electrical resistances (magenta color line) closely follow the recorded changes in water content from the probe sensor.

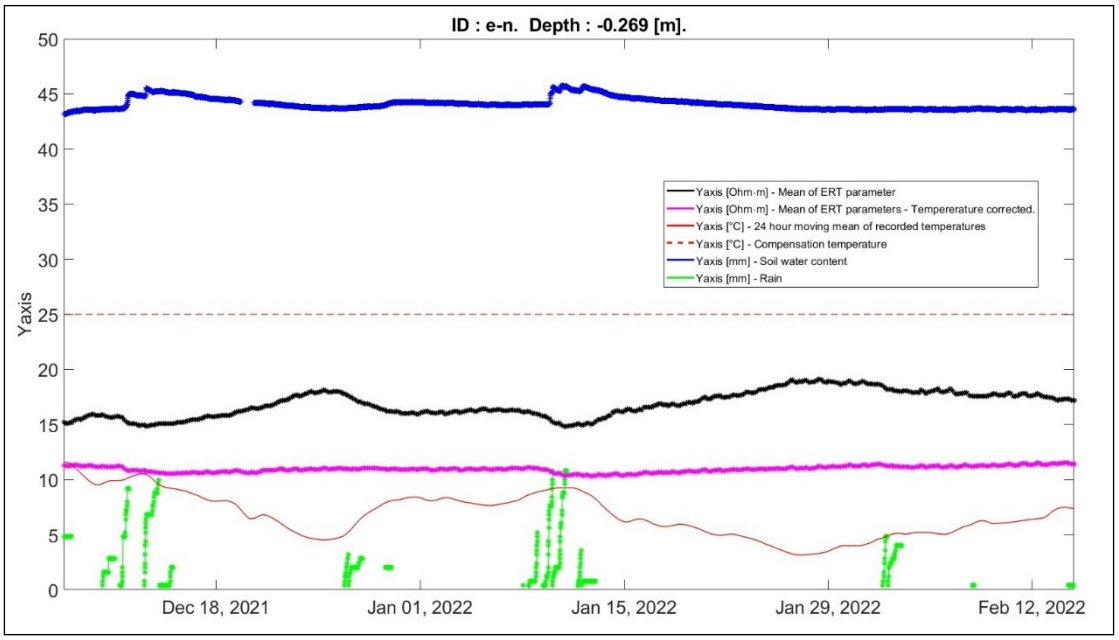

Figure 7.12 Temperature compensation of the ERT.

7.2.5 Time durations estimation of water infiltration

Next, for selected precipitation incidents, the changes over time on the water-content data and the ERT data were investigated. The investigation consists of defining 7 time points (Figure 7.13), TP1) Time point of precipitation start, TP2) Time point where the water content sensor starts to be influenced by the rainwater infiltration, TP3) Time point where the water content sensor reaches the maximum influence by the rainwater infiltration, TP4) Time point where the water content sensor returns to the initial value before the influence by the rainwater infiltration, TP5) Time point where the electrical resistivity values starts to be influenced by the rainwater infiltration, TP6) Time point where the electrical resistivity values reaches the maximum influence by the rainwater infiltration, TP7) Time point where the electrical resistivity values returns to the initial value before the influence by the rainwater infiltration. The selection of each time point was based on the discretion of the analyst and not on any mathematical procedure. Finally, the duration time from TP1 to the other time points was calculated (Table 7.1).

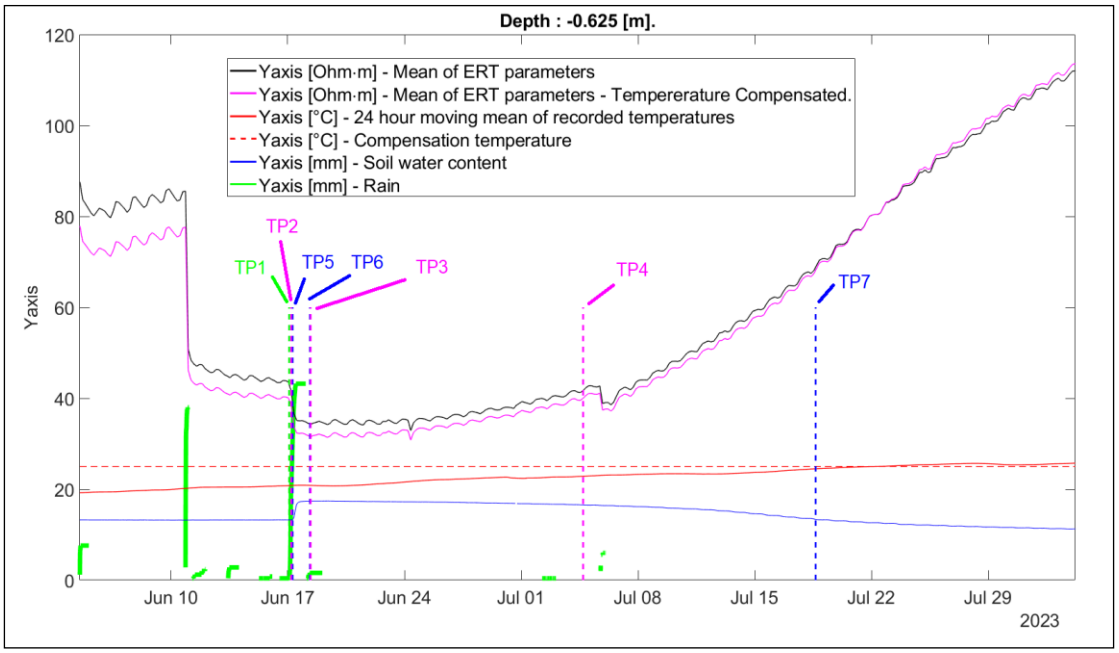

Figure 7.13 Time points selection.

Table 7.1 Example of time durations based on the selected time points.

Depth [m]	TP1 to TP2. Start rain to Start water content sensor effect. [days:hours :min:sec]	TP1 to TP3. Start rain to Max water content sensor effect. [days:hours :min:sec]	TP1 to TP4. Start rain to drainage and finish water content sensor effect. [days:hours :min:sec]	TP1 to TP5. Start rain to Start Resistivity effect. [days:hours :min:sec]	TP1 to TP6. Start rain to Maximum Resistivity effect. [days:hours :min:sec]	TP1 to TP7. Start rain to drainage and finish Resistivity effect. [days:hours :min:sec]
-0.625	00:05:00:00	01:06:00:00	31:13:30:00	00:03:00:00	01:07:00:00	17:15:00:00

The criteria for the investigated rain incidents were that the rainwater infiltration is influencing the deepest water content sensor at -0.85 [m], Figure 7.14. Also, the investigation depths were based on the ERT data inversion grid depths and limited to the depth of the last temperature sensor which is -0.85 [m].

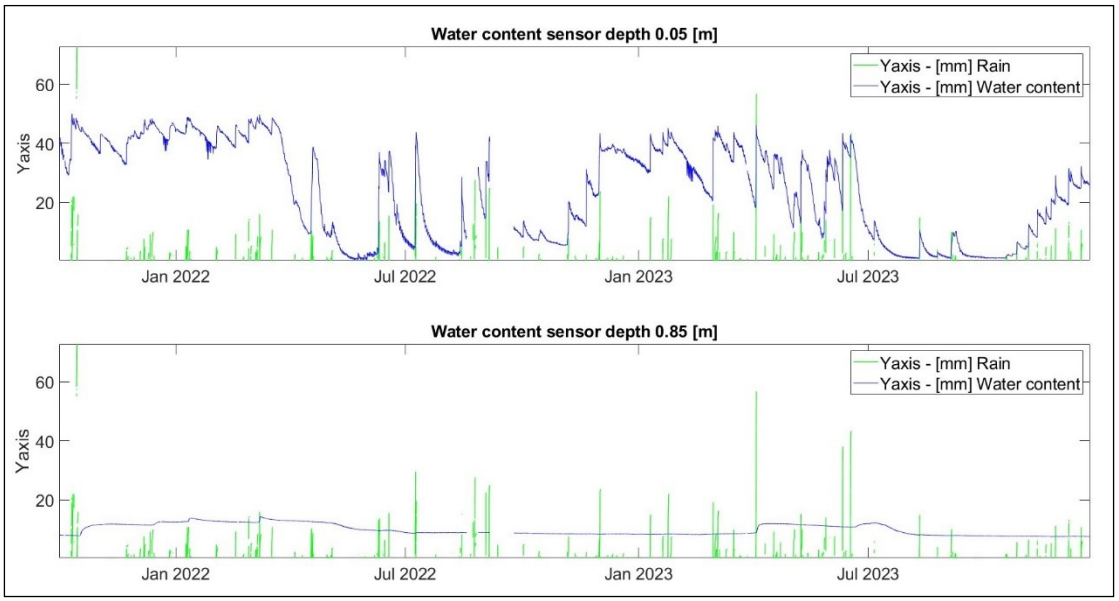

Figure 7.14 Criteria for the investigated rain incidents. Rainwater infiltration is recorded by the deepest water content sensor.

7.2.6 Overview of the collected data and the inversion results

An overview of the collected data is presented in Figure 7.15. The recorded soil water content up to -0.85 [m] for the presented period of the monitoring has a good correlation with the median value of the all the parameters of the ERT for the corresponding dates. The low water content and the high temperatures during summer periods are revealed through the electrical resistivity topographies. Also, the precipitation events during the dry periods of summer have a great impact to the electrical resistivity conditions of the ground. Figure 7.16 and Figure 7.17, present one electrical resistivity tomography from each month of the years 2021, 2022 and 2023, revealing the underground electrical conditions in comparison with above figure.

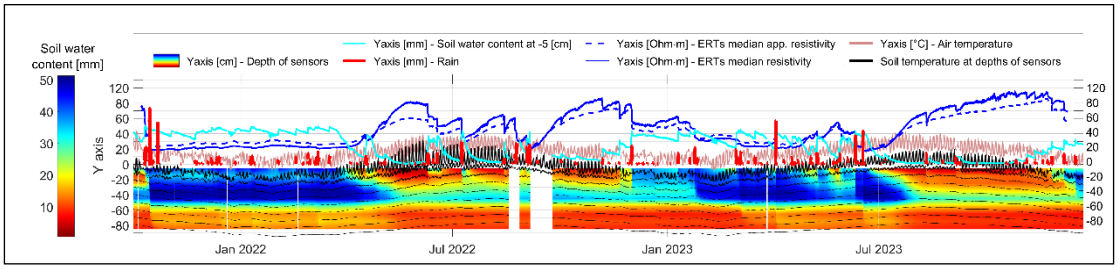

Figure 7.15 Overview of the recorded data.

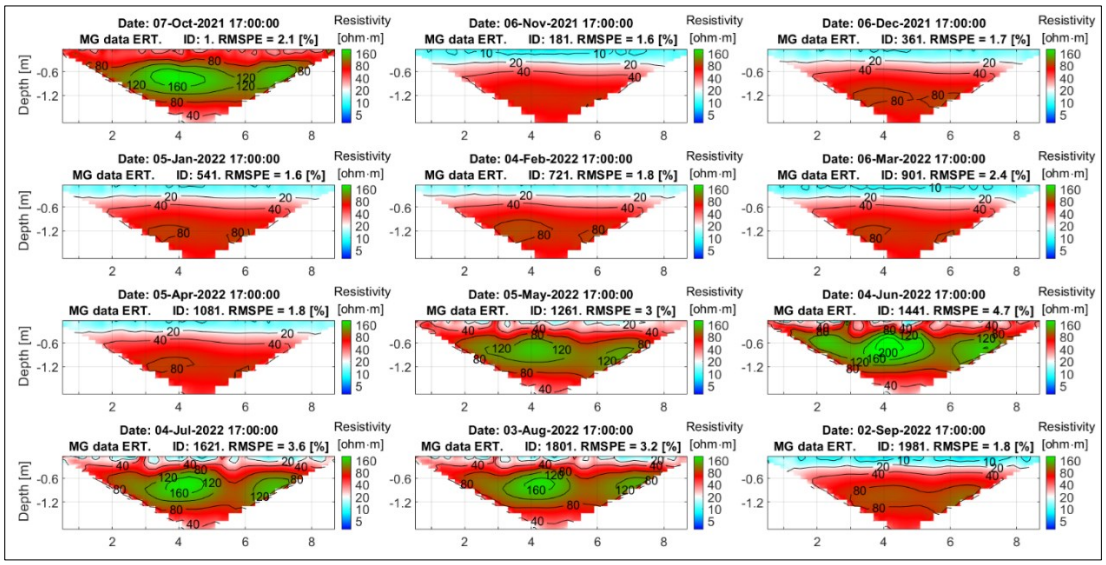

Figure 7.16 One ERT form each month for the period October 2021 - September 2022

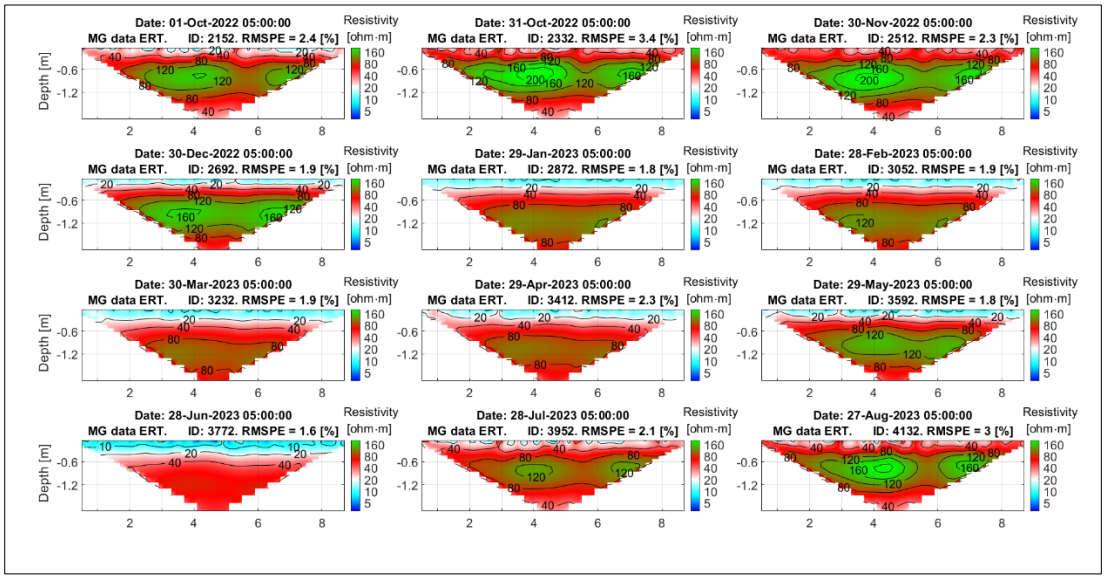

Figure 7.17 One ERT form each month for the period October 2022 - August 2022

In Figure 7.18 is shown a 48 hour period of electrical resistivity tomographies, during a precipitation incident that started at 19:00 of July 8 2022 ant continued for the next two days. The electrical resistivities values gradually decreased as the water concentration increased in the sediment pores.

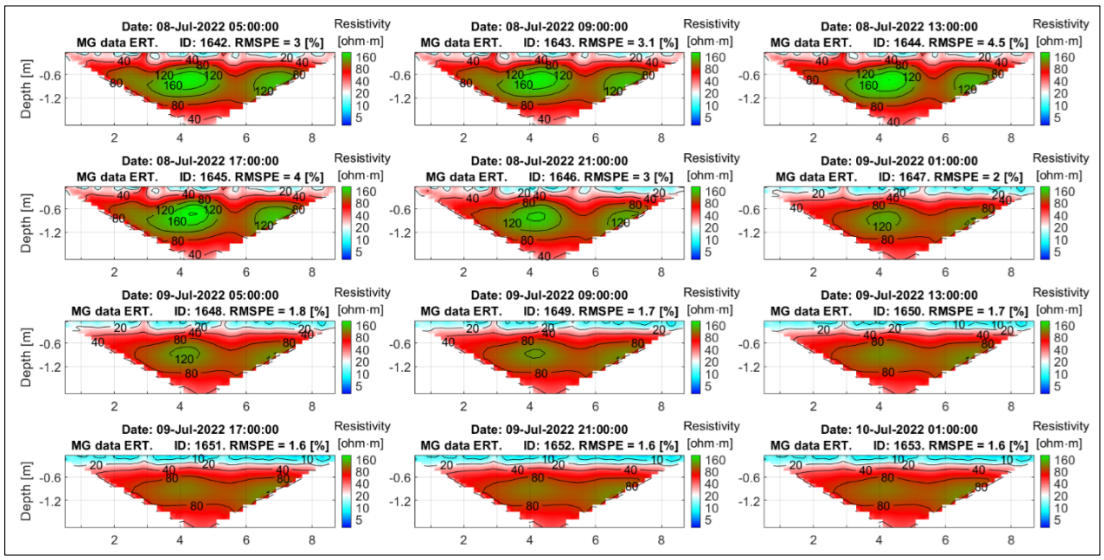

Figure 7.18 Precipitation incident of 8 July as imprinted on the ERT inversion results

In Figure 7.19 is shown the ERT that is the result of the soil texture measurements of each parameter from the 4.757 produced ERTs (based on the multiple gradient data sets). The ERT tomography as well as the soil's layers can be divided into three layers. The upper layer for the surface to -0.6 [m] depth with electrical resistivities up to 60 [ohm·m] consisting mostly of clay and silt sediments. The middle layer form -0.6 [m] up to -1.2 [m] with electrical resistivities between 80 and 120 [ohm·m] consisting mostly of sand sediments. And the lower layer that starts from the -1.2 [m] and continues to greater depths that has similar sediment constituents as the first layer.

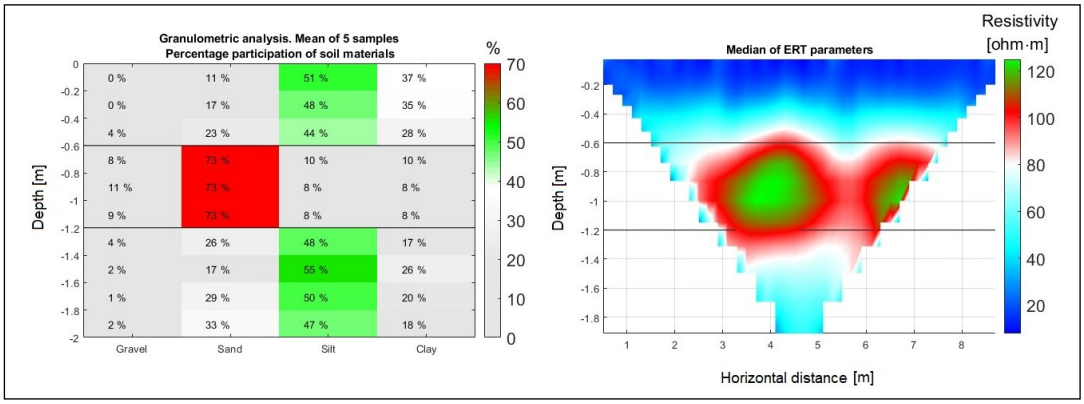

Figure 7.19 Granulometric analysis and media of all inversion results.

7.3 Precipitation incidents

Below are presented 3 cases of precipitation incidents based on the criteria that the rainwater infiltration is influencing has been recorded by the deepest water content

sensor at -0.85 [m]. The selected precipitation incident are A) October 15th 2021, B) March 8 2022 and C) Precipitation incident of June 17th 2023 (Figure 7.20).

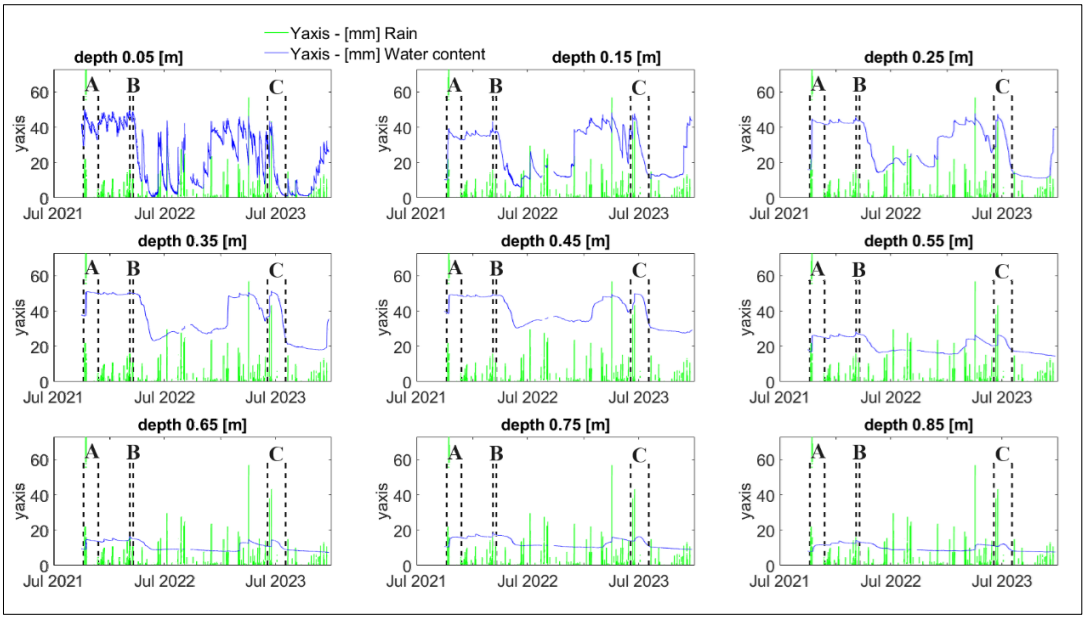

Figure 7.20 Selected precipitation events for analysis of the water infiltration.

The purpose is to investigate how the evolution of the rainwater infiltration is imprinted on the water content sensors as well as on the inversion results. This evolution is expressed is time durations between the start of the rain and other selected time points. These time points as described previously are:

- TP1. Time point of precipitation start.
- TP2. Time point where the water content sensor starts to be influenced by the rainwater infiltration.
- TP3. Time point where the water content sensor reaches the maximum influence by the rainwater infiltration.
- TP4. Time point drainage, where the water content sensor returns to the initial value as before the influence from the rainwater infiltration.
- TP5. Time point where the electrical resistivities starts to be influenced by the rainwater infiltration.
- TP6. Time point where the electrical resistivity values reach the maximum influence by the rainwater infiltration.
- TP7. Time point of drainage, where the electrical resistivities returns to the initial value as before the influence from the rainwater infiltration.

In Table 7.2 are shown the time durations for the precipitation incident. "**NoR TP**" sign in this table means "**Not Recorded Time Point**". There two main reasons for not a recorded time point. The first reason is that very soon after the precipitation incident under investigation a second precipitation event happens and as a result the drainage stage is not happening since new water in imported into the system. NoR TP4 and NoR TP7 signs at the precipitation incident of October 15th 2021 that is presented in below table, is the result of new precipitation very soon after the first one. The second reason (found in ERT measurements of the Precipitation incident example of March 8 2022, designated with NoR TP5 and NoR TP6) is that due to already "water saturated state" of the sediments, the new water that is infiltrating from the precipitation incident has no effect to the electrical resistivities of those sediments as they are already water saturated.

For the small up to -0,269 depths, the low time resolution of the ERT data (4 hours interval) and the long-time distance of the first measured ERT after the precipitation incident of October 15 2021 and June 17 2023, could explain the long delay of electrical tomography to show changes due to infiltrating water effect in comparison with the recorded data of the water content sensor. The same is not applied for the grader depths, as it is presented that the water content sensor is affected by the infiltrating water in slow rater as the depth increases while the effect of the infiltrating water seems that effect instantaneous all the depths on the electrical resistivity results.

Table 7.2 Time durations of water infiltration based on the selected time points. NoR = Not Recorded.

Depth [m]	TP1 to TP2 Duration. Start rain to Start **water content sensor** effect. [days:hours :min:sec]	TP1 to TP3. Duration. Start rain to Max **water content sensor** effect. [days:hours :min:sec]	TP1 to TP4. Duration. Start rain to drainage and finish **water content sensor** effect. [days:hours :min:sec]	TP1 to TP5. Duration. Start rain to Start **Resistivity** effect. [days:hours :min:sec]	TP1 to TP6. Duration. Start rain to Maximum **Resistivity** effect. [days:hours :min:sec]	TP1 to TP7. Duration. Start rain to drainage and finish **Resistivity** effect. [days:hours :min:sec]
Precipitation incident of October 15 2021						
-0.077	00:00:30:00	00:06:30:00	NoR TP4	00:04:00:00	00:16:00:00	NoR TP7
-0.135	00:01:30:00	00:17:00:00	NoR TP4	00:04:00:00	01:04:00:00	NoR TP7
-0.199	00:03:00:00	01:02:30:00	NoR TP4	00:04:00:00	01:04:00:00	NoR TP7

-0.269	00:04:30:00	01:05:00:00	NoR TP4	00:04:00:00	02:00:00:00	NoR TP7
-0.346	00:04:30:00	01:17:30:00	NoR TP4	00:04:00:00	02:08:00:00	NoR TP7
-0.430	00:06:00:00	02:19:00:00	NoR TP4	00:04:00:00	03:08:00:00	NoR TP7
-0.523	00:17:30:00	02:21:00:00	NoR TP4	00:04:00:00	05:00:00:00	NoR TP7
-0.625	00:18:30:00	02:10:00:00	NoR TP4	00:04:00:00	06:12:00:00	NoR TP7
-0.738	00:22:00:00	09:17:00:00	NoR TP4	00:04:00:00	07:04:00:00	NoR TP7
Precipitation incident of March 8 2022						
-0.077	00:00:20:00	00:02:20:00	02:00:50:00	NoR TP5	NoR TP6	NoR TP7
-0.135	00:00:50:00	00:02:20:00	02:02:20:00	NoR TP5	NoR TP6	NoR TP7
-0.199	00:01:20:00	00:02:20:00	02:08:20:00	NoR TP5	NoR TP6	NoR TP7
-0.269	00:01:20:00	00:02:20:00	02:10:20:00	NoR TP5	NoR TP6	NoR TP7
-0.346	00:01:50:00	00:03:20:00	02:06:50:00	NoR TP5	NoR TP6	NoR TP7
-0.430	00:01:50:00	00:03:20:00	02:16:50:00	00:05:20:00	01:13:20:00	NoR TP7
-0.523	00:01:50:00	00:04:20:00	05:00:50:00	00:05:20:00	01:13:20:00	NoR TP7
-0.625	00:01:50:00	00:04:50:00	05:21:50:00	00:05:20:00	01:13:20:00	NoR TP7
-0.738	00:02:20:00	00:10:20:00	NoR TP4	00:05:20:00	01:13:20:00	NoR TP7
Precipitation incident of June 17 2023						
-0.077	00:01:00:00	00:05:00:00	03:22:00:00	00:03:00:00	00:03:00:00	02:07:00:00
-0.135	00:01:30:00	00:05:30:00	04:15:30:00	00:03:00:00	00:03:00:00	05:07:00:00
-0.199	00:02:30:00	00:05:30:00	07:07:30:00	00:03:00:00	00:19:00:00	07:11:00:00
-0.269	00:02:30:00	00:07:00:00	13:10:00:00	00:03:00:00	01:19:00:00	10:03:00:00
-0.346	00:02:30:00	00:07:00:00	20:07:00:00	00:03:00:00	01:07:00:00	12:15:00:00
-0.430	00:02:30:00	00:10:00:00	30:08:00:00	00:03:00:00	01:07:00:00	14:19:00:00
-0.523	00:04:00:00	00:16:00:00	32:12:00:00	00:03:00:00	01:07:00:00	17:15:00:00
-0.625	00:05:00:00	01:06:00:00	31:13:30:00	00:03:00:00	01:07:00:00	17:15:00:00
-0.738	00:07:30:00	13:07:00:00	31:12:30:00	00:03:00:00	01:07:00:00	18:15:00:00

In the next figures, are presented the data used for the determination of the arrival time of the infiltrating water for the precipitation:

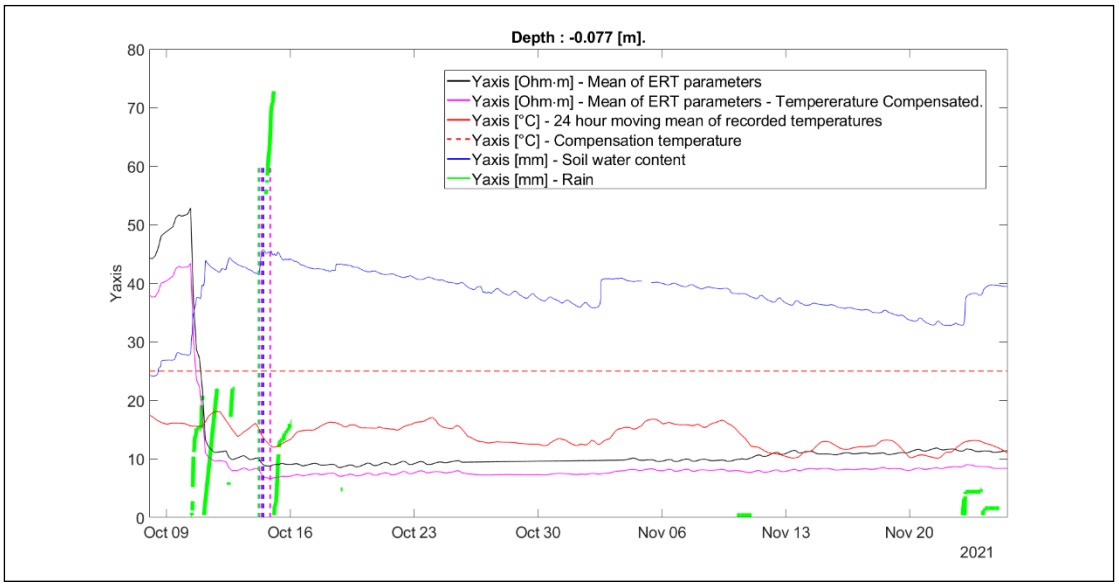

Figure 7.21 Time points selection for precipitation incident of October 15 2021.

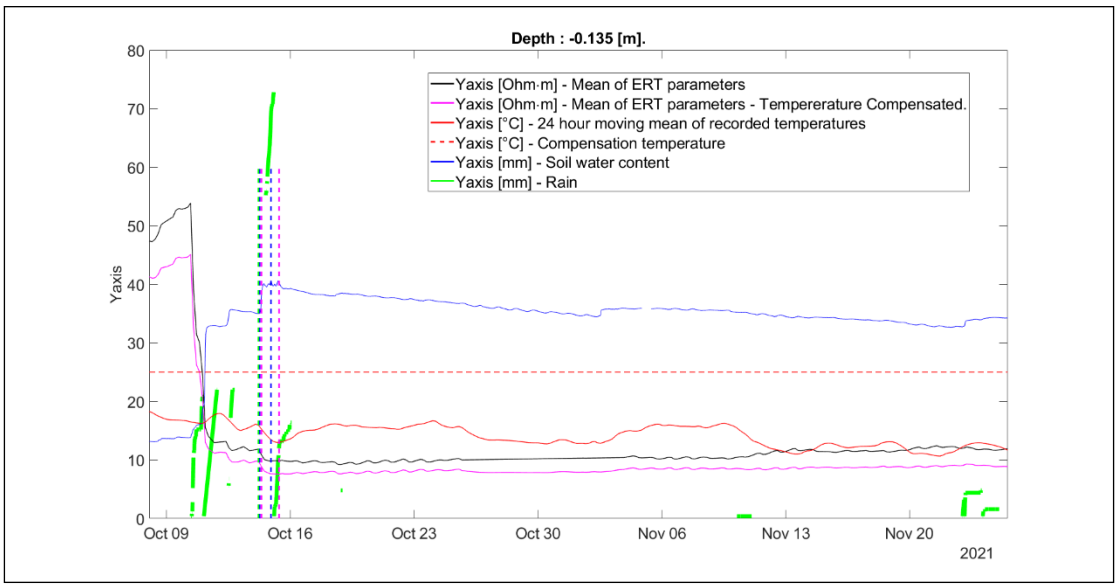

Figure 7.22 Time points selection for precipitation incident of October 15 2021.

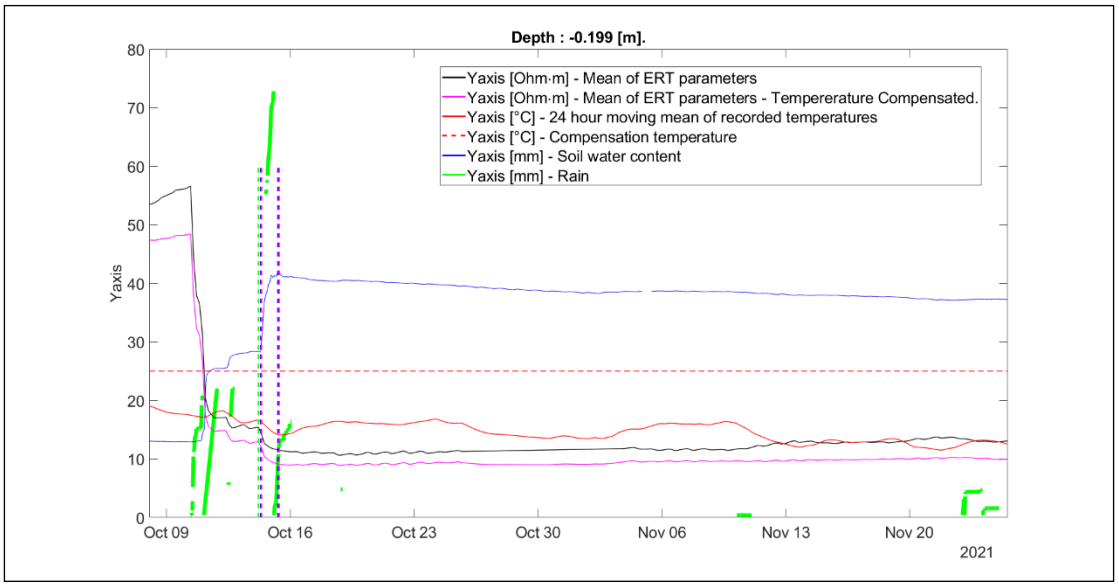

Figure 7.23 Time points selection for precipitation incident of October 15 2021.

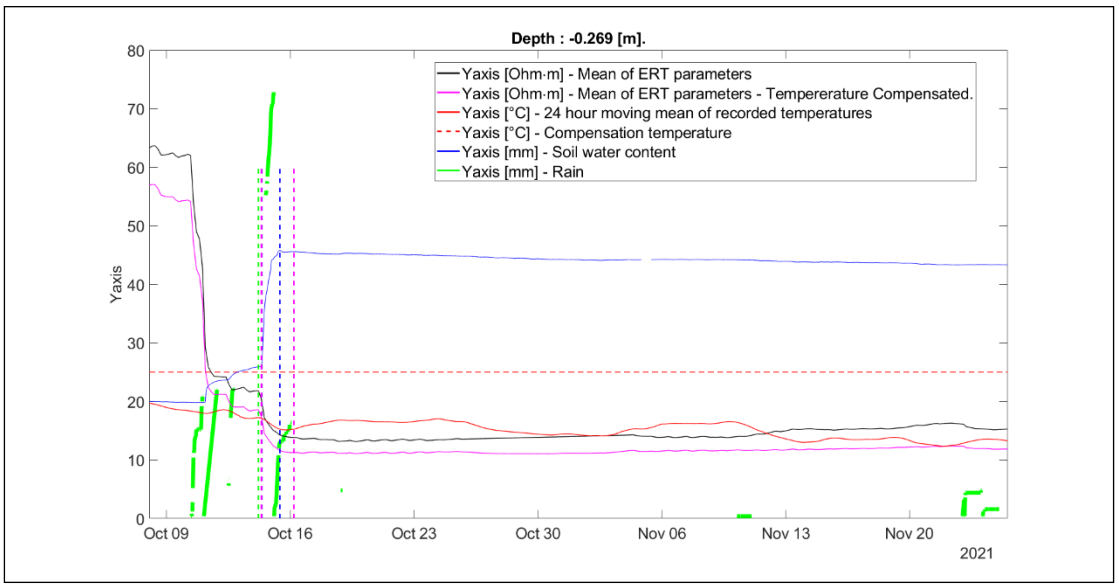

Figure 7.24 Time points selection for precipitation incident of October 15 2021.

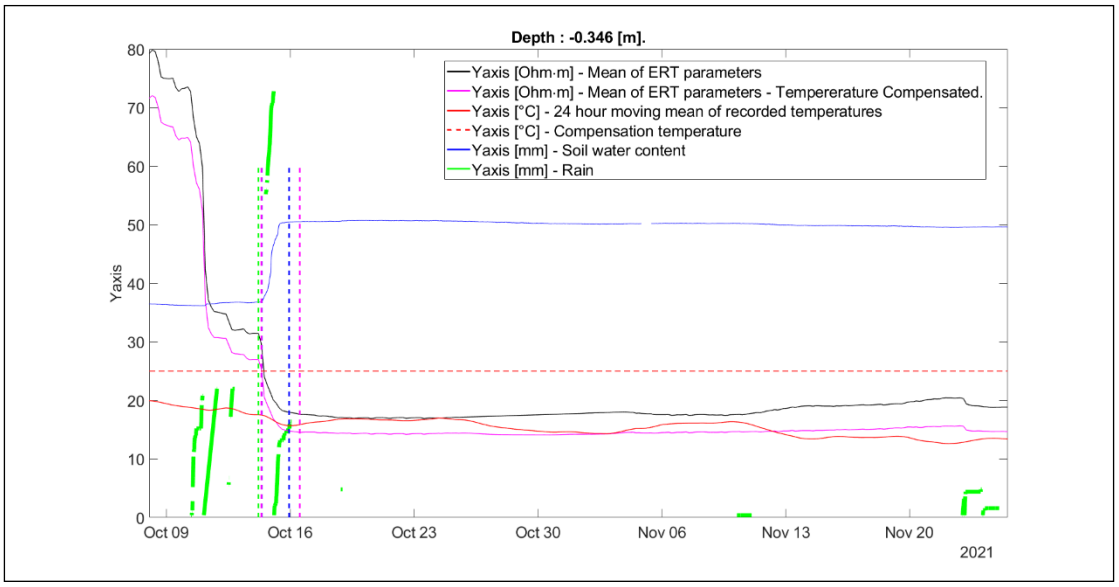

Figure 7.25 Time points selection for precipitation incident of October 15 2021.

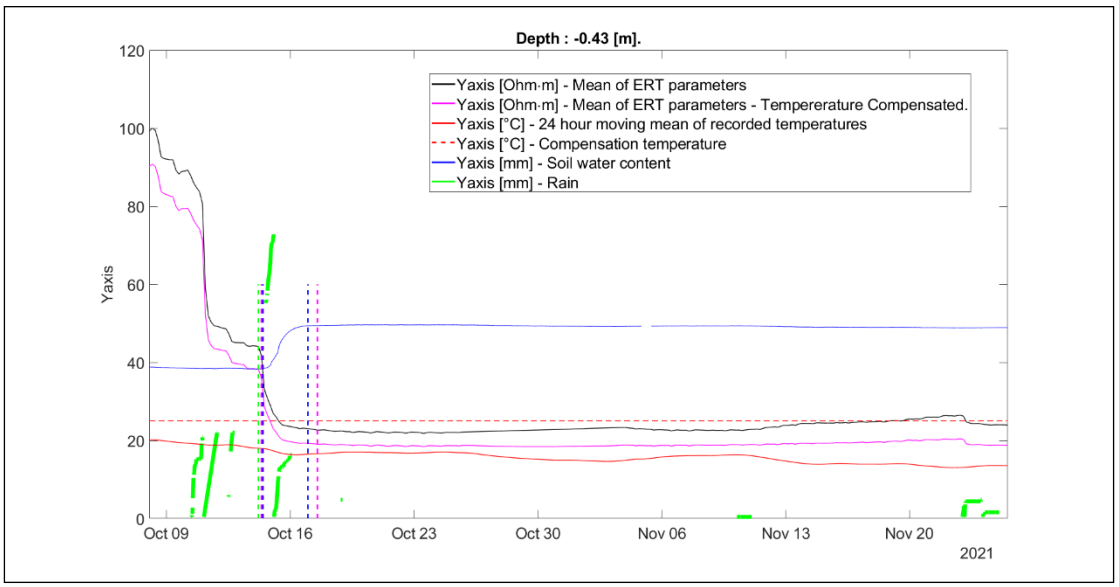

Figure 7.26 Time points selection for precipitation incident of October 15 2021.

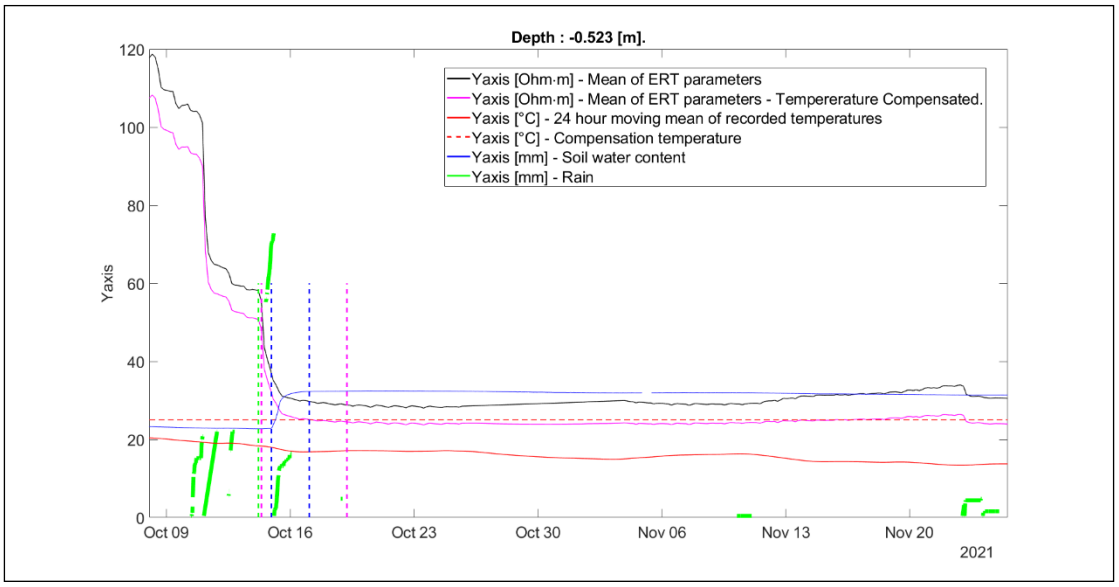

Figure 7.27 Time points selection for precipitation incident of October 15 2021.

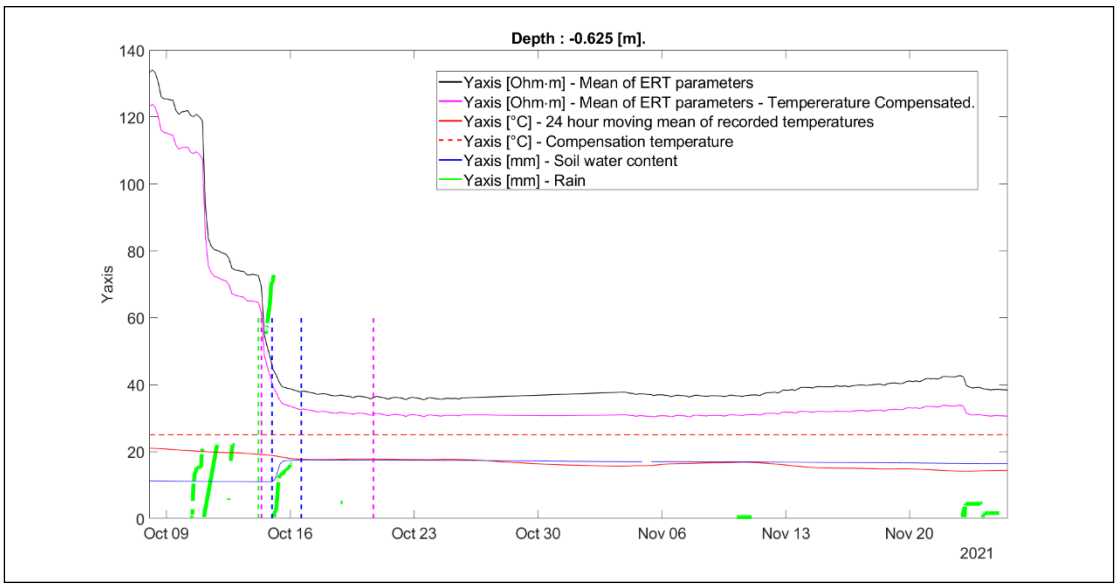

Figure 7.28 Time points selection for precipitation incident of October 15 2021.

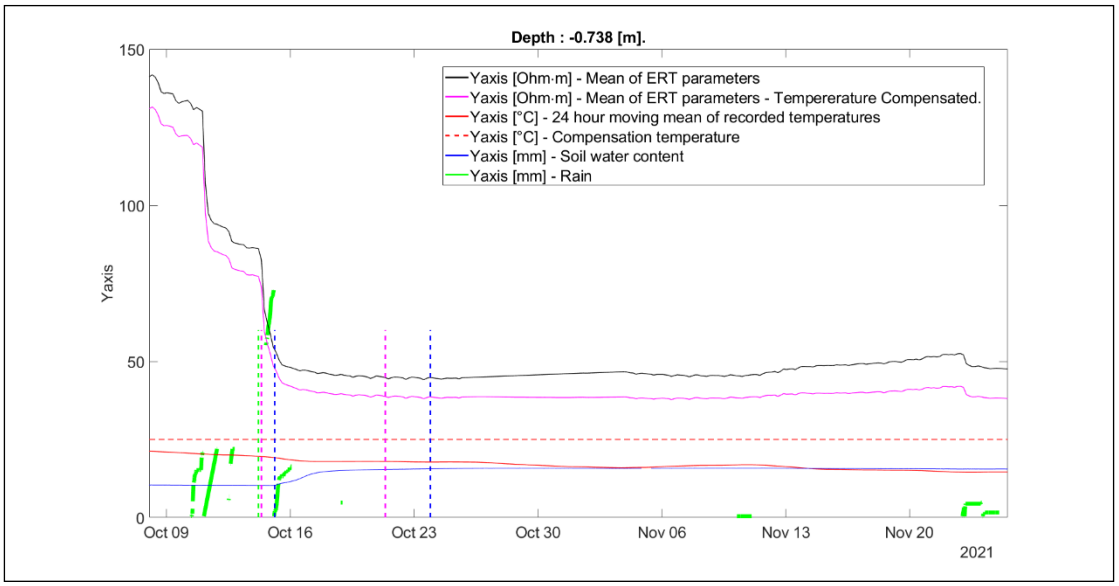

Figure 7.29 Time points selection for precipitation incident of October 15 2021.

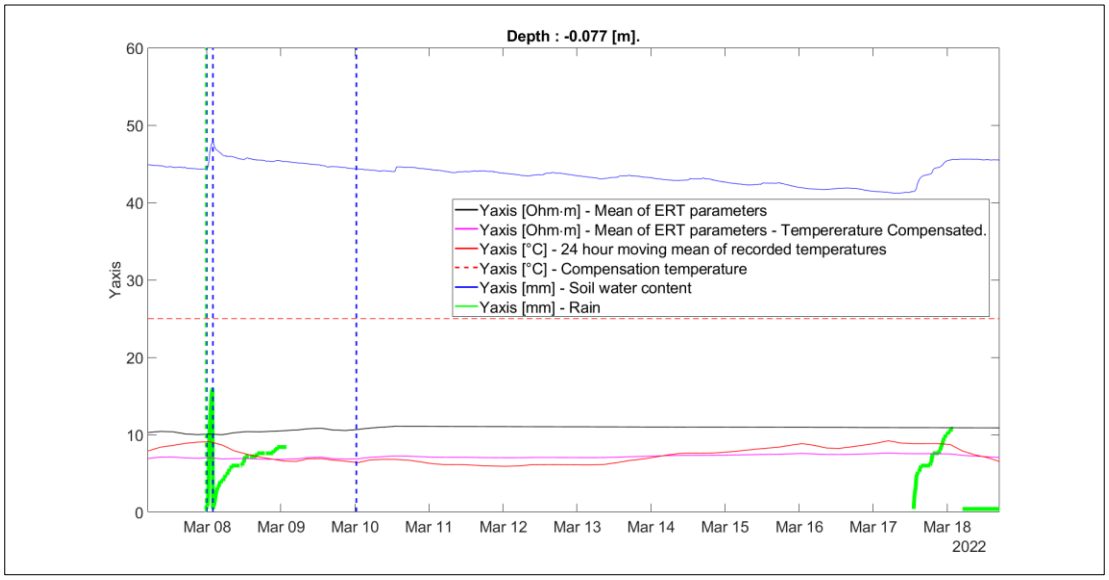

Figure 7.30 Time points selection for precipitation incident of March 8 2022.

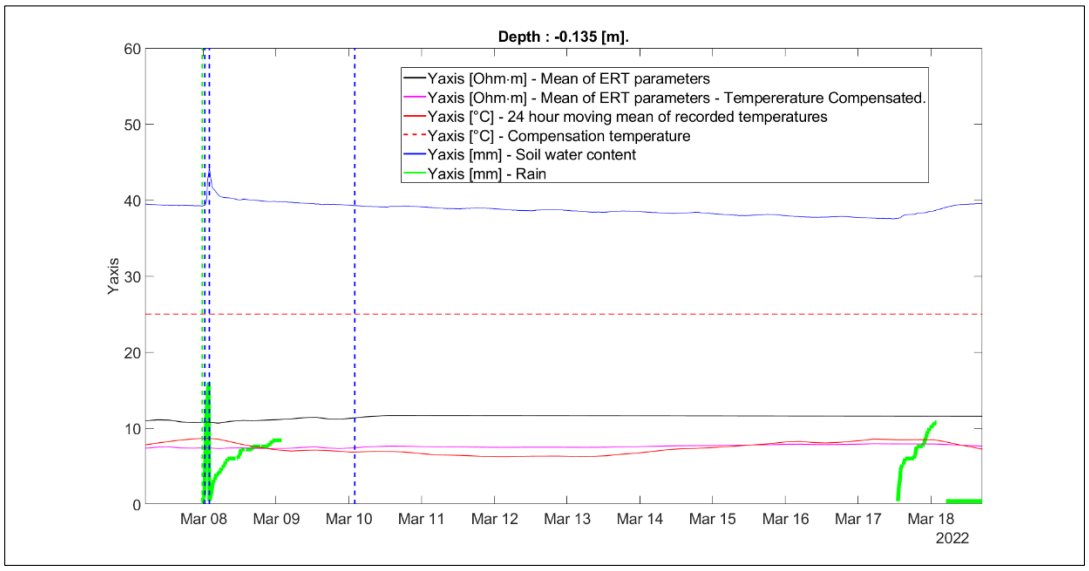

Figure 7.31 Time points selection for precipitation incident of March 8 2022.

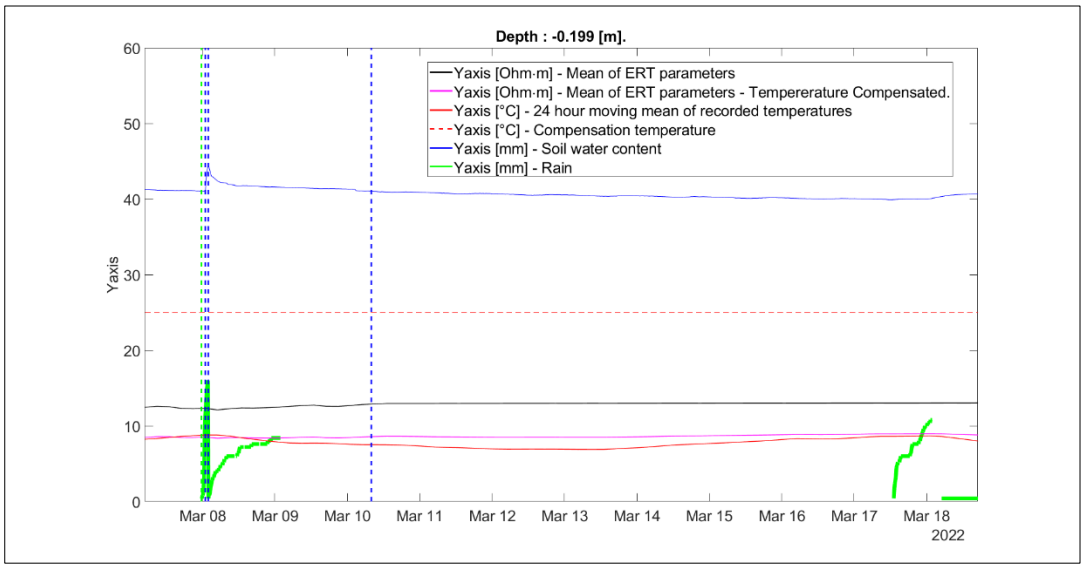

Figure 7.32 Time points selection for precipitation incident of March 8 2022.

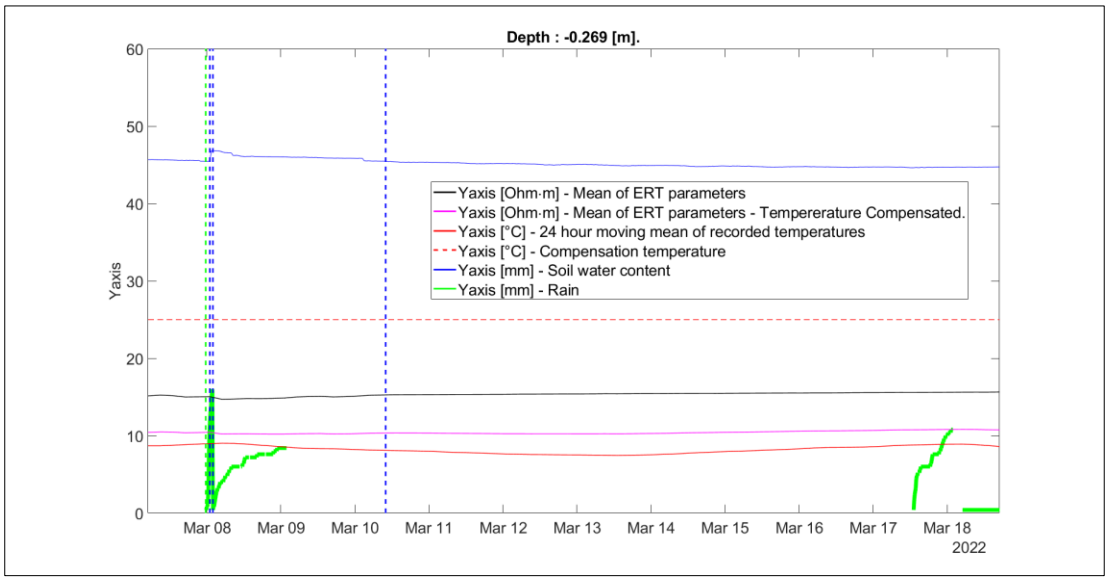

Figure 7.33 Time points selection for precipitation incident of March 8 2022.

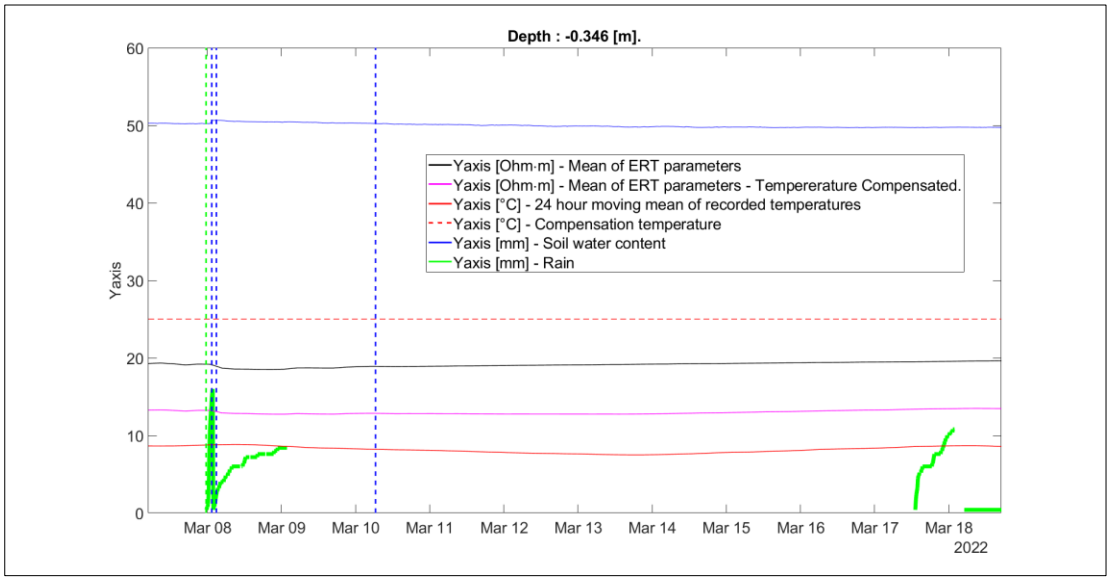

Figure 7.34 Time points selection for precipitation incident of March 8 2022.

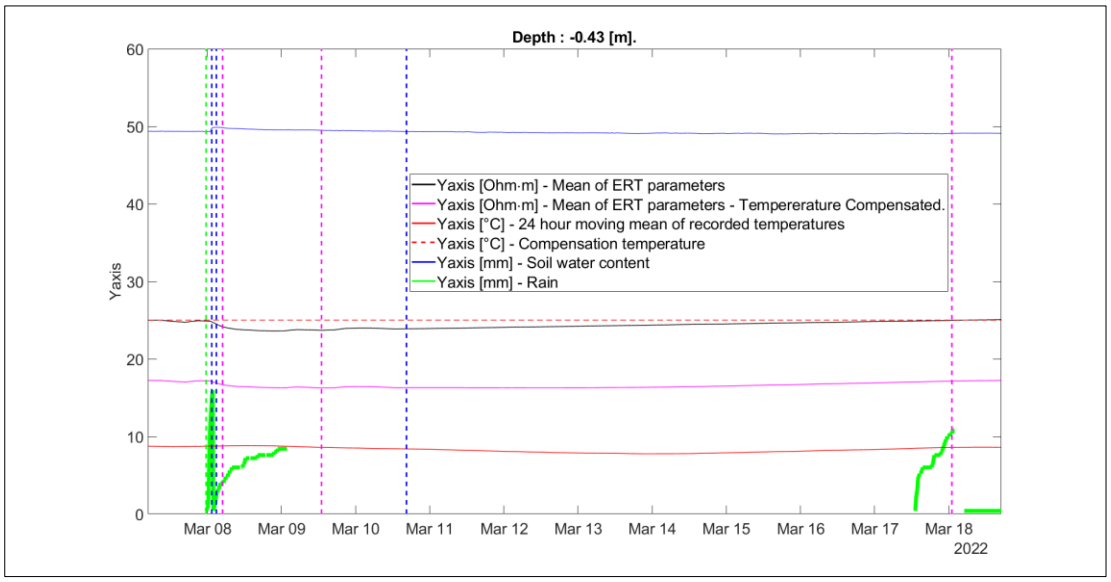

Figure 7.35 Time points selection for precipitation incident of March 8 2022.

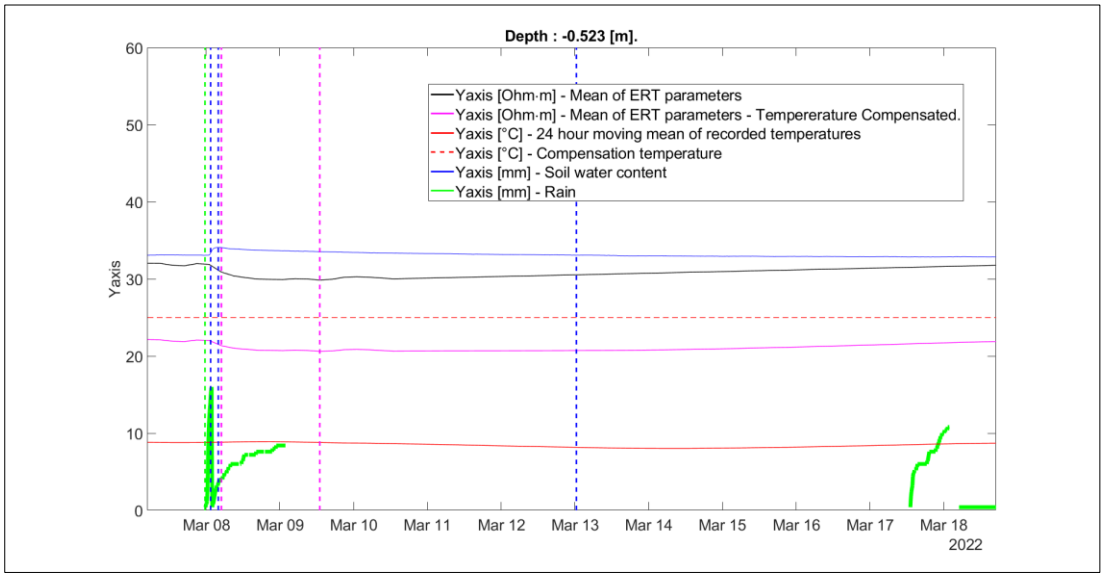

Figure 7.36 Time points selection for precipitation incident of March 8 2022.

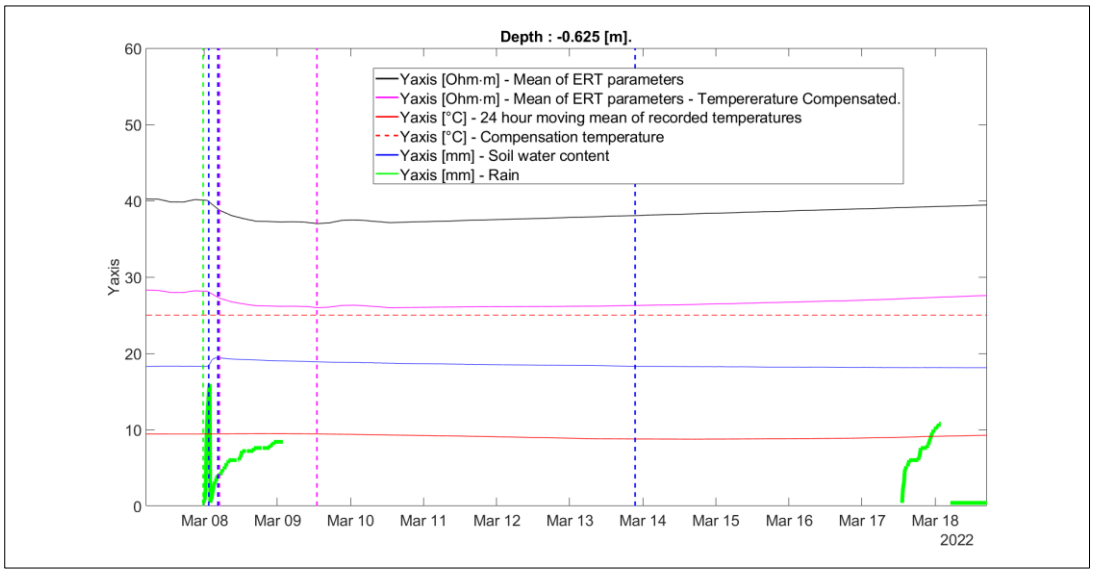

Figure 7.37 Time points selection for precipitation incident of March 8 2022.

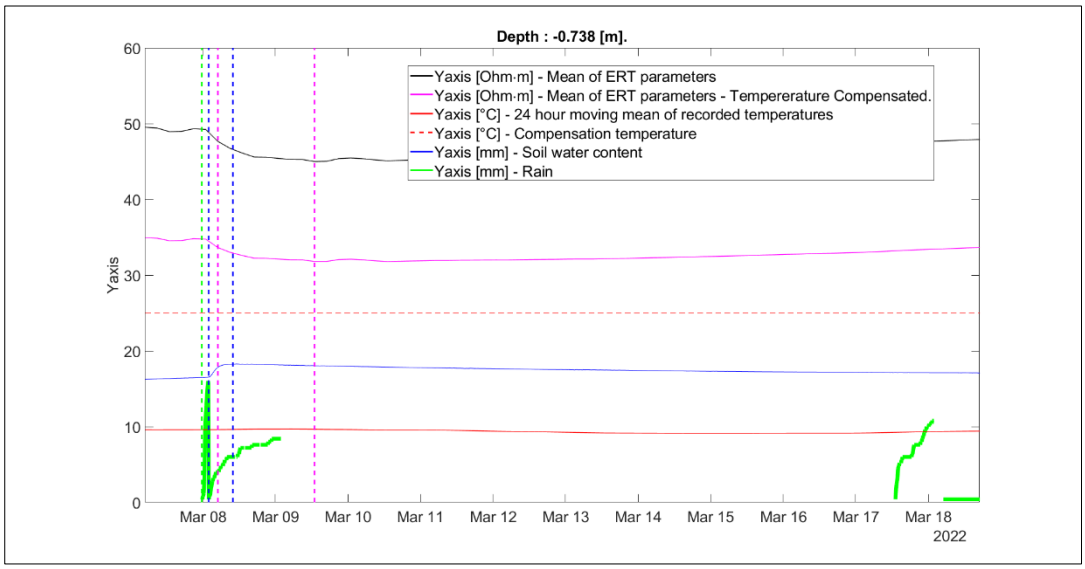

Figure 7.38 Time points selection for precipitation incident of March 8 2022.

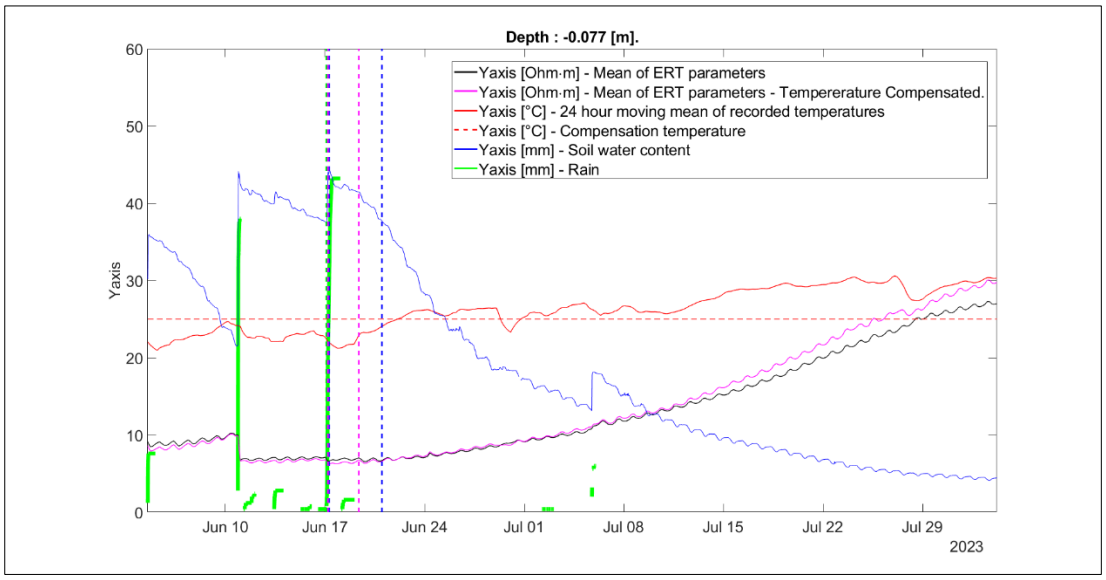

Figure 7.39 Time points selection for precipitation incident of October 17 2023.

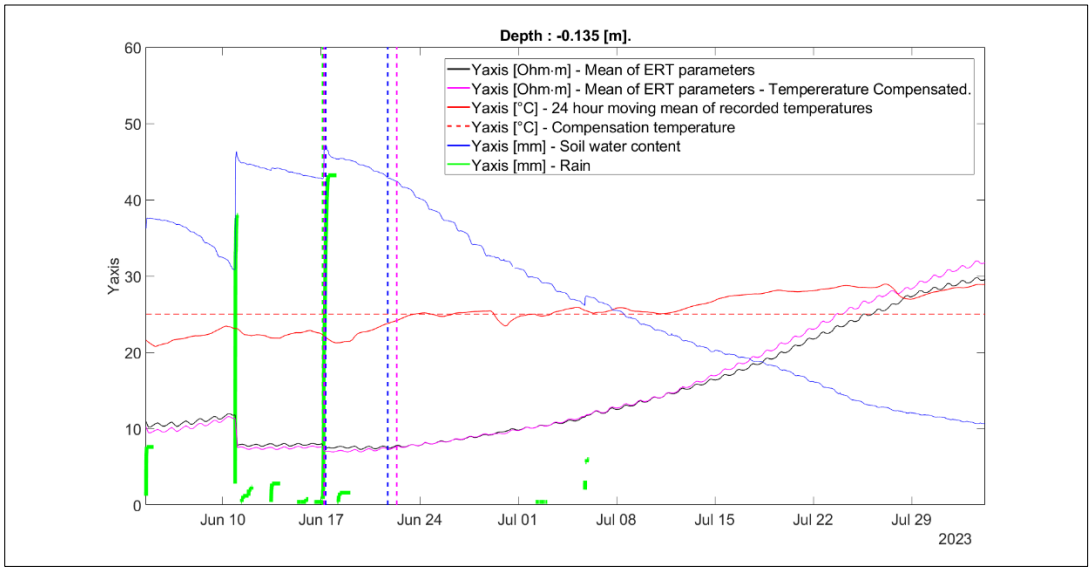

Figure 7.40 Time points selection for precipitation incident of October 17 2023.

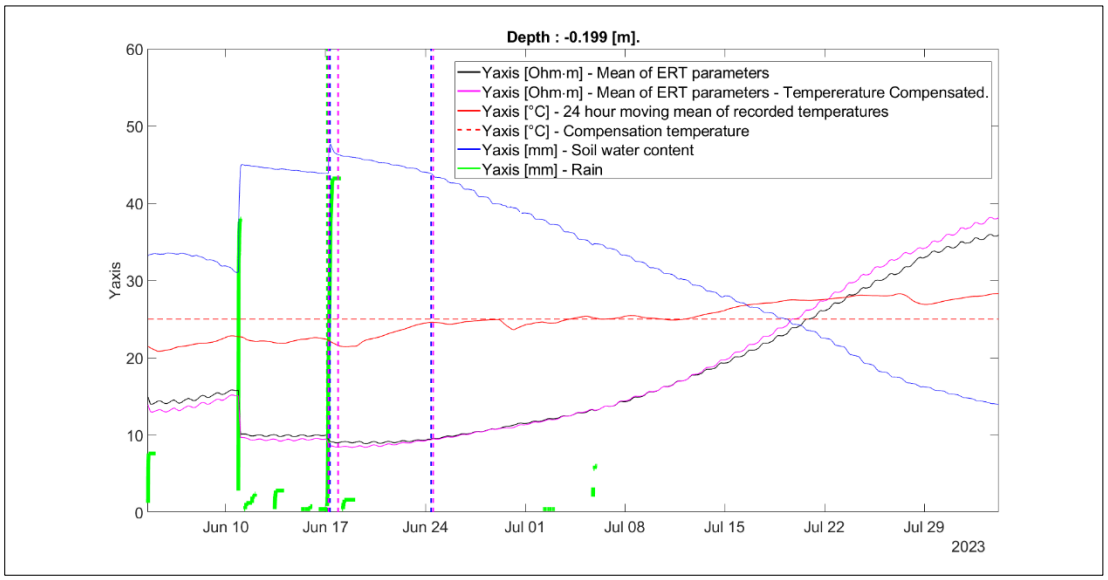

Figure 7.41 Time points selection for precipitation incident of October 17 2023.

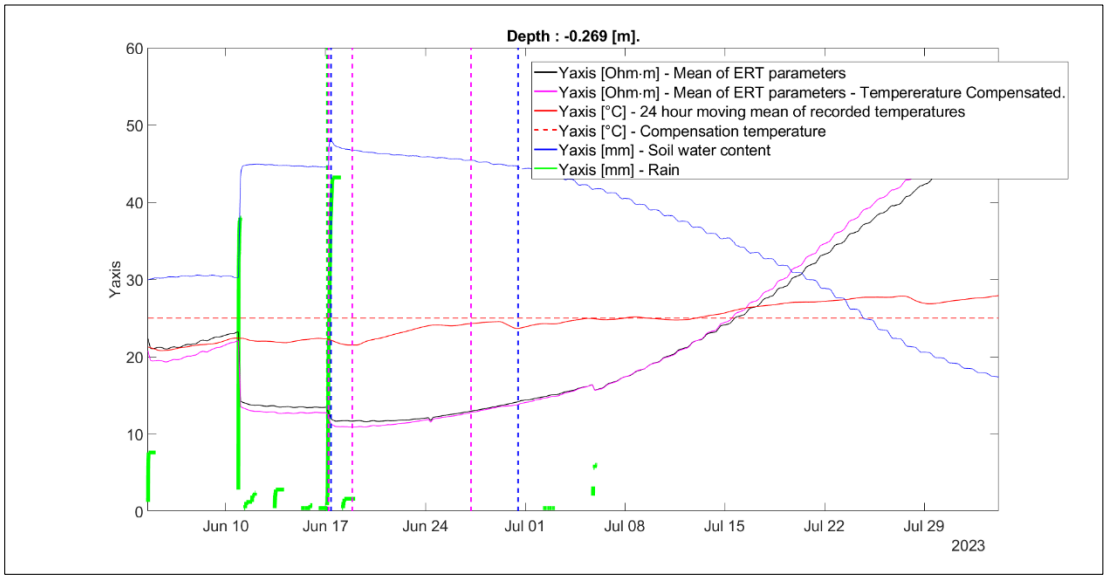

Figure 7.42 Time points selection for precipitation incident of October 17 2023.

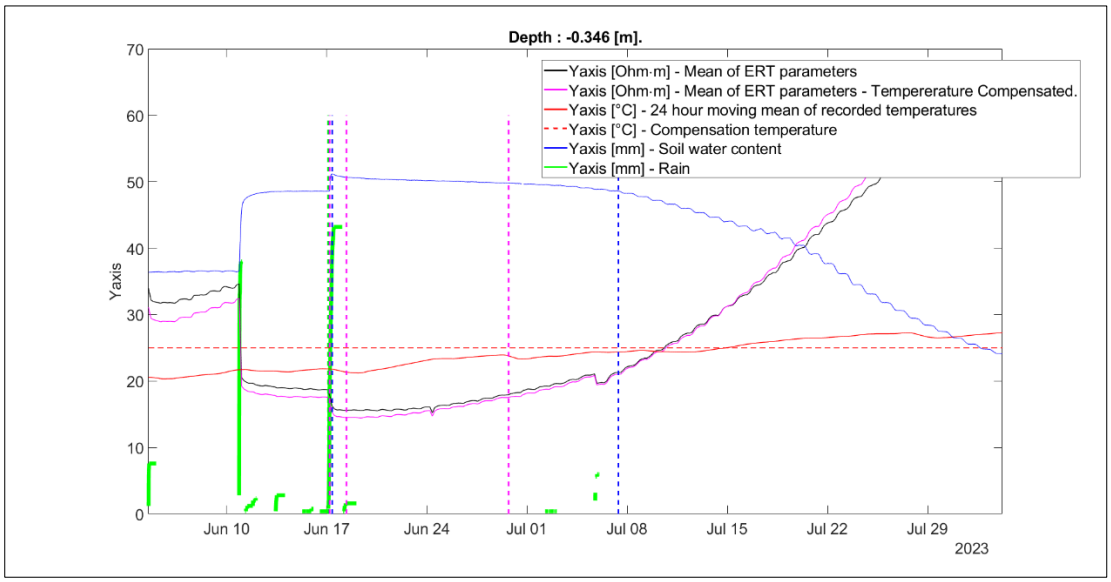

Figure 7.43 Time points selection for precipitation incident of October 17 2023.

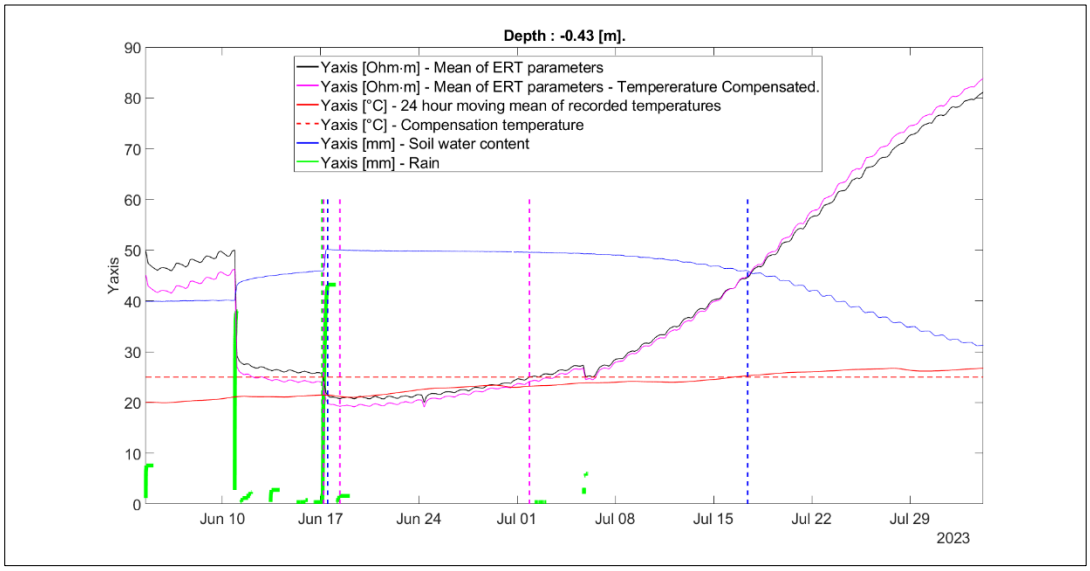

Figure 7.44 Time points selection for precipitation incident of October 17 2023.

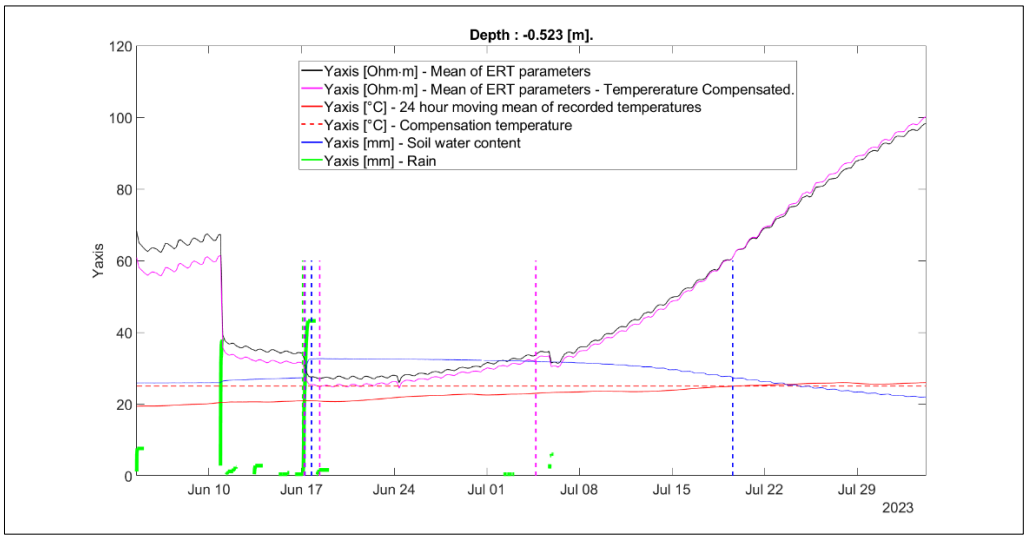

Figure 7.45 Time points selection for precipitation incident of October 17 2023.

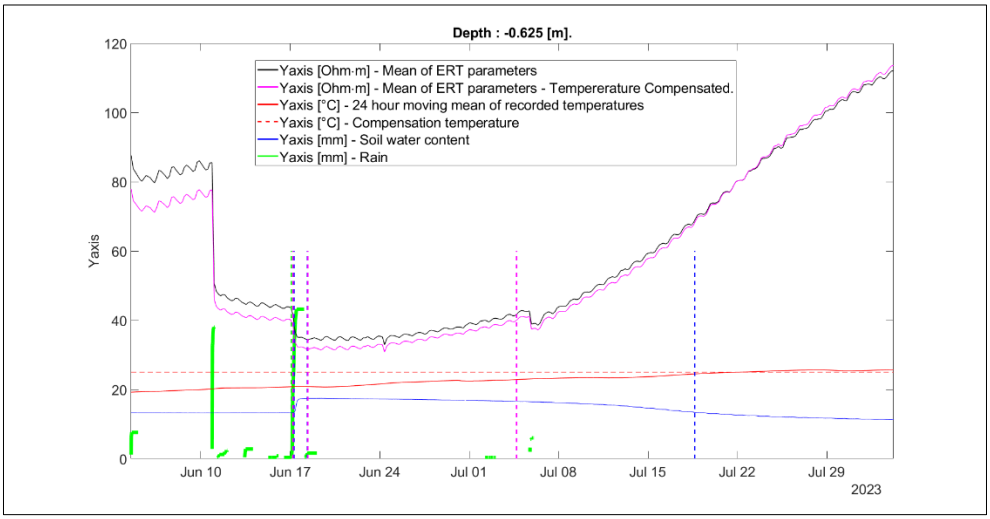

Figure 7.46 Time points selection for precipitation incident of October 17 2023.

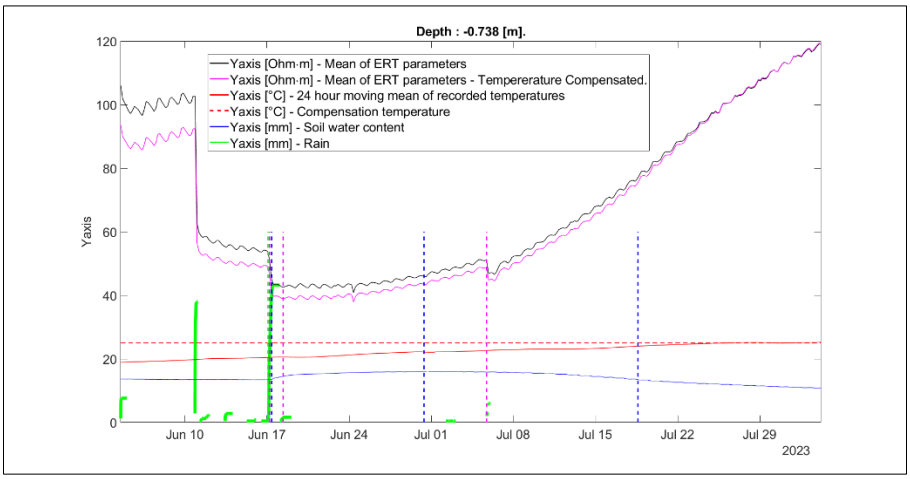

Figure 7.47 Time points selection for precipitation incident of October 17 2023.

The soil texture measurement curves are presented in the following figures:

Figure 7.48 Photo during soil collection in N. Risio site.

Figure 7.49 Photo during soil collection in N. Risio site.

Figure 7.50 Photo during soil collection in N. Risio site.

Table 7.3 Results of the soil texture measurements.

A/A	Sample Code	Percent (%) of Gravel, Sant, Silt, Clay				Percent (%) of Sand, Silt, Clay		
		G	S	Si	C	S	Si	C
1	A 0-0.2	0.1	11.24	49.79	38.86	11.25	49.84	38.90
2	A 0.2-0.4	0.1	11.74	47.81	40.45	11.74	47.81	40.45
3	A 0.4-0.6	18.76	25.48	36.2	19.73	31.30	44.47	24.24
4	A 0.6-0.8	11	74.31	14.69		83.49	16.51	
5	A 0.8-1	16.28	71.72	12		85.67	14.33	
6	A 1-1.2	7.3	60.91	31.79		65.71	34.29	
7	A 1.2 -1.4	9.86	4.43	63.67	22.04	4.91	70.63	24.45
8	A 1.4-1.6	3.94	16.89	54.7	24.47	17.58	56.94	25.47
9	A 1.6-1.8	0.32	17.03	57.95	24.7	17.08	58.14	24.78
10	A 1.8-2	1.08	14.9	67.7	16.31	15.06	68.45	16.49
11	B 0-0.2	0	9.73	52.44	37.82	9.73	52.45	37.82
12	B 0.2-0.4	0	8.96	50.65	40.39	8.96	50.65	40.39
13	B 0.4-0.6	0	22.73	48.07	29.2	22.73	48.07	29.20
14	B 0.6-0.8	8.76	64.8	26.44		71.02	28.98	
15	B 0.8-1	13.18	76.72	10.1		88.37	11.63	
16	B 1-1.2	9.71	82.47	7.82		91.34	8.66	
17	B 1.2-1.4	1.24	28.61	50.65	19.49	28.97	51.29	19.74
18	B 1.4-1.6	5.92	22.55	49.07	22.46	23.97	52.16	23.87
19	B 1.6-1.8	1.09	34	43.02	21.89	34.37	43.49	22.13
20	B 1.8-2	3.58	33.79	42.67	19.95	35.05	44.26	20.69
21	Γ 0-0.2	0	10.04	47.85	42.11	10.04	47.85	42.11
22	Γ 0.2-0.4	0	12.48	52.26	35.26	12.48	52.26	35.26
23	Γ 0.4-0.6	1.43	32.08	39.01	26.89	32.74	39.81	27.44
24	Γ 0.6-0.8	9.58	69.71	20.72		77.09	22.91	

25	Γ 0.8-1	15.92	62.14	21.94		73.91	26.09	
26	Γ 1-1.2	18.07	69.19	12.73		84.46	15.54	
27	Γ 1.2-1.4	4.09	28.52	54.18	13.21	29.74	56.49	13.77
28	Γ 1.4-1.6	0.89	15.98	54.31	28.83	16.12	54.79	29.09
29	Γ 1.6-1.8	0.16	29.7	54.56	15.58	29.75	54.65	15.60
30	Γ 1.8-2	1.62	39	37.49	21.9	39.64	38.10	22.26
31	Δ 0-0.2	0	11.73	55.46	32.81	11.73	55.46	32.81
32	Δ 0.2-0.4	0.12	36.82	40.33	22.73	36.86	40.38	22.76
33	Δ 0.4-0.6	0.26	11.2	51.94	36.6	11.23	52.08	36.70
34	Δ 0.6-0.8	5.38	79.31	15.3		83.83	16.17	
35	Δ 0.8-1	3.06	82.01	14.93		84.60	15.40	
36	Δ 1-1.2	4.41	77.89	17.7		81.48	18.52	
37	Δ 1.2-1.4	0	30.7	55.91	13.39	30.70	55.91	13.39
38	Δ 1.4-1.6	0	17.11	58.61	24.28	17.11	58.61	24.28
39	Δ 1.6-1.8	0.44	34.23	48.71	16.62	34.38	48.93	16.69
40	Δ 1.8-2	0.49	42.26	41.62	15.64	42.46	41.82	15.72
41	E 0-0.2	0	14.42	50.06	35.52	14.42	50.06	35.52
42	E 0.2-0.4	0	16.42	46.65	36.94	16.42	46.65	36.94
43	E 0.4-0.6	0	25.47	46.55	27.98	25.47	46.55	27.98
44	E 0.6-0.8	4.42	74.99	20.59		78.46	21.54	
45	E 0.8-1	4.11	73.47	22.42		76.62	23.38	
46	E 1-1.2	7.31	82.84	9.86		89.36	10.64	
47	E 1.2-1.4	9.04	59.29	31.67		65.18	34.82	
48	E 1.4-1.6	0.3	12.85	57.69	29.17	12.89	57.86	29.25
49	E 1.6-1.8	2.69	30.89	46.16	20.27	31.74	47.43	20.83
50	E 1.8-2	2.46	33.41	46.75	17.38	34.25	47.93	17.82

7.4 Conclusions

The geoelectrical measurement which obtained within the project resulted in the following conclusions:

- During the summer rainfall invents the recharge is insufficient to zero. This evidence supports the argument of the project that small dams constitute a significant solution to supplement the recharge of groundwater using also rainwater during summer periods.
- The study of vadose zone hydrology by using ERT require high frequency data and temperature correction. Even two ERT per month cannot provide valuable information for the recharge rate.

- The recharge rate is strongly dependent by the period of rainfall events and in some zones require more than 7 days to penetrate to deeper zones.

The geoelectrical measurements significantly contributed to the conceptualization of the recharge regime in Anthemountas basin. The geoelectrical data are partly included in a submitted article in the framework of the project, while the elaboration will be continued and will be published even after the end of the project due to the necessity to study more than two hydrological cycles. Obviously the monitoring will be continued after the end of the project (February 2024).

8 Conclusions

Within this report presented the dam optimization code and the corresponding software, as well as the snow code-algorithm. In the next deliverables will be presented the data elaboration with these tools and their results. Additionally, presented the analysis of the meteorological data and the time series of draught indices and rainfall intensity. The draught indices show that after 2020 will occur larger periods of draughts with alternations of wet periods. The most intense problem is in the coastal areas of Greece. In previous years the draught occurs in short periods and alter with wet periods. Rainfall intensity doesn't show a slight which also might influence the recharge of groundwater.

The geoelectrical measurement provided valuable information for the recharge rates in the Eastern Thermaikos Gulf. A valuable conclusion is that only high frequency data of ERT can supplement in the understanding of vadose zone hydrology and recharge regime.

Finally, within this report we provide the measurements of groundwater level which used in the simulation process of groundwater. Further analysis and elaboration of the data is presented in the corresponding publications. The data are available in the web-site of the project. It is necessary the permition of principal investigator if someone want to re-publish this data, while there are not available for commercial reasons.

9 References

Arrey, Ivo & Odiyo, J. & Makungo, Rachel & Kataka, Milton. (2019). Vadose zone infiltration and its implication for groundwater contamination risk assessment in Siloam village, Limpopo province, South Africa. Jàmbá Journal of Disaster Risk Studies. 11. 10.4102/jamba.v11i2.682.

Bakker, M. (2016). FloPy: Python package for creating, running, and post-processing MODFLOW-based models. US Geological Survey, Https://Doi. Org/10.5066/F7BK19FH.Bakker, M., Post, V., Langevin, C. D., Hughes, J. D., White, J. T., Starn, J. J., & Fienen, M. N. (2016). Scripting MODFLOW Model Development Using Python and FloPy. Groundwater, 54(5), 733–739. https://doi.org/10.1111/gwat.12413

Bashiri-Atrabi, H., Qaderi, K., Rheinheimer, D.E. et al. Application of Harmony Search Algorithm to Reservoir Operation Optimization. Water Resour Manage 29, 5729–5748 (2015). https://doi.org/10.1007/s11269-015-1143-3

Brillante, L., Mathieu, O., Bois, B., Van Leeuwen, C., & Lévêque, J. (2015). The use of soil electrical resistivity to monitor plant and soil water relationships in vineyards. Retrieved from https://www.soil-discuss.net/1/C547/2015/soild-1-C547-2015-supplement.pdf

Corwin, D. L., & Lesch, S. M. (2005). Apparent soil electrical conductivity measurements in agriculture. Computers and Electronics in Agriculture, 46, 11–43. https://doi.org/10.1016/j.compag.2004.10.005

CRES. Renewable energy statistics. National report for EUROSTAT; 2006.

Dorigo, M., V. Maniezzo and A. Colorni, "Ant system: optimization by a colony of cooperating agents," in IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), vol. 26, no. 1, pp. 29-41, Feb. 1996, doi: 10.1109/3477.484436.

Droege, P. (Ed.) 100% Renewable: Energy Autonomy in Action; Routledge: London, UK, 2012.

Fayaed, S.S., A. El-Shafie and O. Jaafar, “Reservoir system simulation and optimization techniques,” Stochastic Environmental Research and Risk Assessment, vol. 27, no. 7, pp. 1751-1772, 2013.

Foppa, N. and Seiz, G.: Inter-annual variations of snow days overSwitzerland from 2000–2010 derived from MODIS satellite data,The Cryosphere, 6, 331–342, doi:10.5194/tc-6-331-2012, 2012.

Gafurov, A. and Bárdossy, A.: Cloud removal methodology fromMODIS snow cover product, Hydrol. Earth Syst. Sci., 13, 1361–1373, doi:10.5194/hess-13-1361-2009, 2009.

Geem Z.W. (2007) Optimal Scheduling of Multiple Dam System Using Harmony Search Algorithm. In: Sandoval F., Prieto A., Cabestany J., Graña M. (eds) Computational and Ambient Intelligence. IWANN 2007. Lecture Notes in Computer Science, vol 4507. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-73007-1_39

Geem, Z.W., J.H. Kim & G.V. Loganathan (2002) Harmony Search Optimization: Application to Pipe Network Design, International Journal of Modelling and Simulation, 22:2, 125-133, DOI: 10.1080/02286203.2002.11442233

Geem, Z.W.; Kim, J.H.; Loganathan, G.V. A New Heuristic Optimization Algorithm: Harmony Search. Simulation 2001, 76, 60–68.

Geem, Z.W.; Kim, J.H.; Loganathan, G.V. Harmony Search Optimization: Application to Pipe Network Design. Int. J. Model. Simul. 2002, 22, 125–133.

Harbaugh, A.W., Langevin, C.D., Hughes, J.D., Niswonger, R.N., and Konikow, L. F., 2017, MODFLOW-2005 version 1.12.00, the U.S. Geological Survey modular groundwater model: U.S. Geological Survey Software Release, 03 February 2017, http://dx.doi.org/10.5066/F7RF5S7G

Hasan torabi poudeh Maryam Mirbeyk Sabzevari, Application of Harmony Search Algorithm in Optimization of the Dez Dam Reservoir Operation for Long Period, Water and Soil Science Volume 29, Issue 4Winter 2020 Pages 109-120

Hassanvand, MR, Karami, H, Mousavi, S-F. Use of multi-criteria decision-making for selecting spillway type and optimizing dimensions by applying the harmony search algorithm: Qeshlagh Dam Case Study. Lakes & Reserv. 2019; 24: 66– 75. https://doi.org/10.1111/lre.12250

Holland, John H."Genetic Algorithms and the Optimal Allocation of Trials." SIAM J. Comput. 2 (1973): 88-105.http://dx.doi.org/10.1109/ICNN.1995.488968

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

Hüsler, F., Jonas, T., Riffler, M., Musial, J. P., and Wunderle, S.: A satellite-based snow cover climatology (1985–2011) for the European Alps derived from AVHRR data, The Cryosphere, 8, 73–90, https://doi.org/10.5194/tc-8-73-2014, 2014.

JANATROSTAMI S.*, KHOLGHI M., BOZORG HADDAD OMID,Management of reservoir operation system using improved harmony search algorithm, WATER AND SOIL SCIENCE (AGRICULTURAL SCIENCE) FALL 2010 , Volume 20.1 , Number 3; Page(s) 61 To 72.

Kazakis, Voudouris, Vargemezis, & Pavlou (2013). Hydrogeological regime and groundwater occurence in the Anthemountas River Basin. Bulletin of the Geological Society of Greece, 47, 711-720. https://www.researchgate.net/publication/313229655_Hydrogeological_regime_and_groundwater_occurence_in_the_Anthemountas_River_Basin [accessed Feb 01 2024].

Kaldellis JK (2007). The contribution of small hydro power stations to the electricity generation in Greece: Technical and economic considerations. Energy policy, 35(4}, 2187-2196.

Kaldellis, J.K., Katsirou, V., Kondili, E., Korbakis, G., 2006. Optimal sizing of small hydro power plants for maximum energy production. In: Eigth National Conference on the Soft Energy Resources, Thessaloniki, Greece, 2006.

Katsifarakis, Konstantinos. (2008). Groundwater Pumping Cost Minimization – an Analytical Approach. Water Resources Management. 22. 1089-1099. 10.1007/s11269-007-9212-x.

Kennedy, J. and Eberhart, R. (1995) Particle Swarm Optimization. Proceedings of the IEEE International Conference on Neural Networks, 4, 1942-1948.

Kirkpatrick, S., Gelatt, J.C.D. and Vecchi, M.P. (1983) Optimization by Simulated Annealing. Science, 220, 671-680. http://dx.doi.org/10.1126/science.2204598.671

Kougias I, Ho V, Kim J. A Hybrid Harmony Search Optimization Algorithm: Application on Hydropower in Central Vietnam. In Conference Proceedings: Proceedings of OPT-i Conference on Engineering and Applied Sciences Optimization. National Technical University of Athens (NTUA); 2014. p. 1046-1054. JRC88446.

Kougias I; Karakatsanis D; Malatras A; Monforti-Ferrario F; Theodossiou N. Renewable energy production management with a new harmony search optimization toolkit. CLEAN TECHNOLOGIES AND ENVIRONMENTAL POLICY 18 (8); 2016. p. 2603–2612. JRC97549

Kougias, Ioannis and Nikolaos Theodosiou. "A NEW MUSIC-INSPIRED HARMONY BASED OPTIMIZATION ALGORITHM . THEORY AND APPLICATIONS." (2010).

Leaf, A. T., & Fienen, M. N. (2022, November 1). Flopy: The Python Interface for MODFLOW. Groundwater. John Wiley and Sons Inc. https://doi.org/10.1111/gwat.13259

Lie, B., & Haugen, F. (2015). Scripting Modelica Models using Python. SNE Simulation Notes Europe, 23(3–4). https://doi.org/10.11128/sne.23.tn.10212

Lundh, F. (1999). An introduction to tkinter. URL: Www. Pythonware. Com/Library/Tkinter/Introduction/Index. Htm.

Ma, R., McBratney, A., Whelan, B., Minasny, B., & Short, M. (2011). Comparing temperature correction models for soil electrical conductivity measurement. Precision Agriculture, 12(1), 55–66. https://doi.org/10.1007/s11119-009-9156-7

Marty, C. and Meister, R.: Long-term snow and weather observa-tions at Weissfluhjoch and its relation to other high-altitude ob-servatories in the Alps, Theor. Appl. Climatol., 110, 573–583,doi:10.1007/s00704-012-0584-3, 2012.

Matias, M., & Almeida, F. (2017). ERT and the Location of Mining Cavities in Anisotropic Media: A Field Example. In Cave Investigation. InTech. https://doi.org/10.5772/intechopen.68475

McKee, T.B., Doesken, N.J., Kleist, J., 1993. The relationship of drought frequency and duration to time scales. In Proceedings of the Eight Conference on Applied Climatoligy, Anaheim, CA, USA, 17–22.

McKee, T.B., N.J. Doesken, J. Kleist, 1993. The relationship of drought frequency and duration of time scales. Eighth Conference on Applied Climatology, American Meteorological Society, Jan17-23, 1993, Anaheim CA, pp.179-186.

Meløysund, Vivian &Leira, Bernt&Høiseth, Karl &Lisø, Kim, Predicting snow density using meteorological data. Meteorological Applications, 14, p. 413 – 423, 2007.

Michot, D., Benderitter, Y., Dorigny, A., Nicoullaud, B., King, D., & Tabbagh, A. (2003). Spatial and temporal monitoring of soil water content with an irrigated corn crop cover using surface electrical resistivity tomography. Water Resources Research, 39(5), 1–20. https://doi.org/10.1029/2002WR001581

Milan Cisty and Veronika Soldanova (December 20th 2017). Ensemble Prediction of Stream Flows Enhanced by Harmony Search Optimization, Time Series Analysis and Applications, Nawaz Mohamudally, IntechOpen, DOI: 10.5772/intechopen.71192. Available from: https://www.intechopen.com/books/time-series-analysis-and-applications/ensemble-prediction-of-stream-flows-enhanced-by-harmony-search-optimization

Mohamed SHAMS, Ahmed EL-BANBI, Helmy SAYYOUH,Harmony search optimization applied to reservoir engineering assisted history matching,Petroleum Exploration and Development,Volume 47, Issue 1,2020,Pages 154-160,ISSN 1876-3804, https://doi.org/10.1016/S1876-3804(20)60014-3

Mohammad Hadi AfsharMohamad AzizipourMohamad AzizipourB. OghbaeeaJoong Hoon KimJoong Hoon Kim, Exploring the Efficiency of Harmony Search Algorithm for Hydropower Operation of Multi-reservoir Systems: A Hybrid Cellular Automat-Harmony Search Approach, International Conference on Harmony Search Algorithm, February 2017.

Mousavi, S. J., Nakhaei, P., Sadollah, A., & Kim, J. H. (2017). Optimization of hydropower storage projects using harmony search algorithm. In J. Del Ser (Ed.), Harmony Search Algorithm - Proceedings of the 3rd International Conference on Harmony Search Algorithm (ICHSA 2017) (pp. 261-270). (Advances in Intelligent Systems and Computing; Vol. 514). Springer Verlag. https://doi.org/10.1007/978-981-10-3728-3_26

Muchingami, I., Hlatywayo, D. J., Nel, J. M., & Chuma, C. (2012). Electrical resistivity survey for groundwater investigations and shallow subsurface evaluation of the basaltic-greenstone formation of the urban Bulawayo aquifer. Physics and Chemistry of the Earth, 50–52, 44–51. https://doi.org/10.1016/j.pce.2012.08.014

Nay Myo Lin, Martine Rutten, Optimal Operation of a Network of Multi-purpose Reservoir: A Review,Procedia Engineering,Volume 154,2016,Pages 1376-1384,ISSN 1877-7058, https://doi.org/10.1016/j.proeng.2016.07.504.

Patsialis, T.; Kougias, I.; Kazakis, N.; Theodossiou, N.; and Droege, P. Supporting Renewables' Penetration in Remote Areas through the Transformation of Non-Powered Dams. Energies, 2016, 9(12), p.1054.

PipeFlow. "Darcy-Weisbach Formula Flow of Fluid through a Pipe." PipeFlow (2023): n. pag. PipeFlow. Web.

Pistocchi, Alberto & Bagli, Stefano &Callegari, Mattia &Notarnicola, C. &Mazzoli, Paolo., On the Direct Calculation of Snow Water Balances Using Snow Cover Information Water, 9, p. 848, 2017.

RAE - Regulatory Authority for Energy (www.rae.gr)

Rodell, M., Houser, P. R., Jambor, U., Gottschalck, J., Mitchell, K., Meng, C.-J., Arsenault, K., Cosgrove, B., Radakovich, J., Bosilovich, M., Entin, J. K., Walker, J. P., Lohmann, D., & Toll, D., The Global Land Data Assimilation System, Bulletin of the American Meteorological Society, 85(3), 381-394. Retrieved Jan 28, 2021, from https://journals.ametsoc.org/view/journals/bams/85/3/bams-85-3-381.xml, 2004.

Sturm, M., Taras, B., Liston, G. E., Derksen, C., Jonas, T., & Lea, J. (2010). Estimating Snow Water Equivalent Using Snow Depth Data and Climate Classes, *Journal of Hydrometeorology*, *11*(6), 1380-1394. Retrieved Apr 6, 2022, from https://journals.ametsoc.org/view/journals/hydr/11/6/2010jhm1202_1.xml

Subyani, A.M.; Sen, Z. Refined chloride mass-balance method and its application in Saudi Arabia. Hydrol. Process. **2006**, 20, 4373–4380.

Theis, C.V. (1935) The Relation between the Lowering of the Piezometic Surface and the Rate and Duration of Discharge of a Well Using Groundwater Storage. Transactions on American Geophysical Union, Washington DC, 518-524.

Van Rossum, G. (2020). The Python Library Reference, release 3.8.2. Python Software Foundation.

Vicente-Serrano, S.M., Beguería, S., López-Moreno, J.I., 2010. A Multi-scalar drought index sensitive to global warming: The Standardized Precipitation Evapotranspiration Index – SPEI. Journal of Climate, 23, 1696-1718.

Voudouri K.A, Ntona M.M and Kazakis N., Investigating the snow water equivalent in Greece, Proceedings of the 15th International Conference on Meteorology, Climatology and Atmospheric Physics COMECAP 2021, Ioannina, Greece, 26–29 September 2021; Bartzokas, A., Nastos, P., Eds.; Hellenic Meteorological Society: Ioannina, Greece, 2021.

Wang, X., Xie, H., Liang, T., and Huang, X.: Comparison and validation of MODIS standard and new combination of Terra and Aqua snow cover products in northern Xinjiang , China, Hydrol. Process., 429, 419–429, doi:10.1002/hyp.7151, 2009.

Wesemann, J., Herrnegger, M., & Schulz, K., Hydrological modelling in the anthroposphere: Predicting local runoff in a heavily modified high-alpine catchment. Journal of Mountain Science, 15(5), 921–938. https://doi.org/10.1007/s11629-017-4587-5, 2018.

Xu,W.; Meng, L.; Liu, P.; Dong, K. Use of a modified chloride mass balance technique to assess the factors that influence groundwater recharge rates in a semi-arid agricultural region in China. Environ. Earth Sci. 2019, 78, 241.

YPEKA - Ministry of Environment & Energy (www.ypeka.gr)

Zong Woo Geem, Joong Hoon Kim, Loganathan GV. A New Heuristic Optimization Algorithm: Harmony Search. SIMULATION. 2001;76(2):60-68. doi:10.1177/003754970107600201