The return period of rainfall-induced static liquefaction of tailings dams

A modelling study

J.M.G. van der Zon

The return period of rainfall-induced static liquefaction of tailings dams

A modelling study

by

J.M.G. van der Zon

to obtain the degree of Master of Science
at the Delft University of Technology,

Student number:	4342593	
Thesis committee:	Dr. Sc. A. Askarinejad,	TU Delft, chairman
	Dr. R.C. Lanzafame,	TU Delft
	Dr. P.J. Vardon	TU Delft
	Ir. J. de Greef	Witteveen+Bos, daily supervisor

Preface

This thesis is the final part of my MSc. programme at TU Delft. Around February 2020, I was invited by the company of Witteveen+Bos to graduate on the topic of the liquefaction of tailings dams. Upon reading on some of the material available on this phenomenon, I quickly became intrigued by its complexity. It also felt like a perfect topic to combine the two MSc. programmes that I am following: Applied Earth Sciences and Civil Engineering. Both with a focus on the interaction between soil, structures and water.

Working on this thesis has been a tough but satisfying job. The pandemic has posed an additional challenge as I have not been able to come to the office of W+B often because of it. However, I have always felt welcome at the company. Through here, I sincerely want to thank Jos de Greef and Robert de Heij for offering me this opportunity and the guidance during the past months. Furthermore, I have enjoyed the freedom that was given to me to work on this issue. It allowed me to fully make the work my own. Furthermore, I would like to express my gratitude to the chair of my graduation committee, Amin Askarinejad, for his support and critical view on my work. I would like to thank Robert Lanzafame and Phil Vardon as well for assessing my thesis and providing with directions and tips during the meetings that we had.

Last, but certainly not least, I would like to thank my family, friends and girlfriend for their support during the past 10 months especially and also for the amazing moments during the past years in Delft.

Jelle van der Zon,
Lisse, February 2021

Summary

Tailings are the rock flour which result from the process of mineral extraction at a mine. This waste material is stored in tailings storage facilities, also known as tailings impoundments. These impoundments gradually grow in size during operation as more tailings are produced and stored over time. These structures are however also prone to failure. Well known failures in the past years are that of the Feijão and Fundão dams in the Minais Gerais state in Brazil. Static liquefaction of the dams has caused mudstreams in which many people were killed and the environment became heavily polluted. This study researched the probability of rainfall-induced static liquefaction as meteorological causes have often been named the trigger of failure. The study was limited to upstream raised decommissioned dams in the Minas Gerais state in Brazil.

An analysis methodology was established in which the needed rainfall intensity to failure was calculated for different durations of rainfall. In a geohydrological analysis, the Richards equation for unsaturated flow was solved, to model rainfall infiltration. A quasi-2D procedure was implemented to allow for a computationally efficient solution. Following the results of the geohydrological analysis, a limit equilibrium method was employed to calculate the factor of safety against liquefaction of the dam. In this case, the undrained strength analysis was applied together with an effective stress analysis to model both liquefiable tailings and non-liquefiable tailings in the tailings storage facility. If the factor of safety was equal to a predefined factor of safety associated with failure, the failure intensity was found.

The failure intensity was converted to a return period of failure by the application of the Gumbel distributions that are part of the intensity-duration-frequency curve of rainfall. These IDF-curves describe for all possible combinations of intensity and duration the probability of occurrence. Integration of the IDF-curve for the failure intensities yields the total return period of failure as a single performance indicator. The analysis was restricted to a period of 5 days of continuous rainfall (storm events) as evaporation effects were not considered and no longer rainfall events were observed in the precipitation data used.

A series of sensitivity studies was performed to observe how rainfall influences the stability of a tailings dam. It was found that failure is attributed to a loss of suction in the unsaturated zone upon rainfall infiltration as well as a loss of strength due to mounding of the water table. Because of the low strengths that liquefiable tailings have, there was the tendency to develop deep slip surfaces below the phreatic surface, causing only marginally stable dams to be prone to failure because of rainfall infiltration. The factor of safety upon rainfall is reduced in the order of 2 to 5 % depending on the parameters used. The strength parameters were of the largest influence. Increasing the friction angle of the non-liquefiable tailings by 10 % increases the return period of failure by approximately 400 years. The strength ratios used to model the liquefiable tailings are of even larger influence. Adjusting that by only 2 % leads to a jump in the return period of failure of around 1000 years. The chosen boundary conditions for the phreatic surface in the impoundment were also of great influence. A difference of 0.3 meters for the water table for a dam of 20 meters high brings the dam to marginal stability.

Given the little sensitivity to storm events, it is recommended to shift focus towards the behaviour of the phreatic surface over longer simulation periods to model the effect of e.g. extremely wet seasons. In addition, construction defects should be considered in which perched water tables could arise, severely decreasing the stability of a tailings dam.

Contents

List of Figures

List of Tables

CHAPTER 1

Introduction

1.1. Background

On the 25th of January 2019, one of the tailings impoundments of the Feijão iron ore mine near Brumadinho, Brazil, liquefied. The event triggered a mud flow of many millions of cubic meters of tailings material. Apart from the large environmental consequences that the failure had, 259 people were confirmed to have lost their lives with 11 people still missing (Noguiera and Plumb, 2020). An expert panel was assembled to investigate the causes of this failure (Robertson et al., 2019). It was found that a loss of suction due to intensive rainfall in combination with a phenomenon known as 'creep rupture', in which the tailings material loses strength upon deviatoric creep deformation, caused collapse of the impoundment. The study has also shown the complexity of the failure of tailings dams as the prerequisites for failure were set long before. Amongst others, these conditions were: a steep upstream slope, poor water management with weak tailings near the crest, lack of drainage, a setback over finer tailings and the fact that the tailings had a high brittleness index due to oxidation bonding.

The failure of the Feijão dam I was not the first case of a tailings dam failure in Brazil. Another large failure occurred in 2015 in the Fundão mine, killing 19 people and causing a major ecological disaster with tailings travelling hundreds of kilometres. This failure was also caused by liquefaction of the tailings impoundment (Oliviera and Kerbany, 2016; Queiroz et al., 2018). In the investigation, it was concluded that construction history had been a vital aspect in the collapse (Morgenstern et al., 2016). Construction problems lead to: increased saturation of the dam due to damage to the starter dam, deposition of slimes near the dam and structural problems with a concrete conduit so the upstream dams had to be built on top of the slimes. In the period after these structural issues, the slimes began to laterally extrude under the weight of the overlying tailings. This caused a stress state inside the liquefiable tailings for which only a minor seismic event was needed to trigger the liquefaction event.

These failures in Brazil are not standalone when considering the worldwide occurrences of tailings dams collapse. There are an estimated 3500 tailings dams worldwide. Over the past few decades, on average 2 to 5 tailings dams have failed per year. This equates to a probability of failure of around 1:750 (Davies, 2002). Morgenstern (2018) mentioned in his Victor de Mello lecture that in the worldwide context of these failures, 50% of them could be attributed to: 'the inadequate understanding of undrained failure mechanisms leading to static liquefaction with extreme consequences'. Also with respect to this, the failures of the Brazilian dams are not unique.

There are over 750 other tailings dams present in Brazil (Palu and Julien, 2019). These geotechnical structures pose an immediate risk to their surroundings (Owen et al., 2020). After the failures, the Brazilian government has decided to decommission all of the tailings dams which are of the same type as the Feijão and Fundão dam: the upstream type. It is unsure whether stopping the operational phase of tailings impoundments significantly decreases the risk as the lifetime of tailings impoundments is eternity. They may be decommissioned but cannot be removed. Loads may intensify over time and the material may degrade. The latter was the case in the failure of the Feijão dam which was out of operation for several years already before failure. In addition, the government is taking measures which are of gargantuan proportions by building secondary dams which should hold back the liquid flow if one of the tailings dams fail (DRR Team Brazil, 2019). It is mentioned by DRR Team Brazil (2019) that: ‘The fact that the failing dam in the Brumadinho disaster had an emergency level 1 [ed. lowest risk class] introduces a lack of confidence in the assessments of the current safety levels’. This issue raises the question whether the measures taken at present are effective and decrease the risk of dams where it is most needed. In other words, are tailings dam engineers able to adequately define the risks that these structures pose?

1.2. Problem statement and objective

At present there is no probabilistic framework available for the design and assessment of tailings dams. Design is often based on the well-known factor of safety (FoS) as a performance indicator. The Brazilian guidelines for tailings dam design mention the FoS to be met in a safety assessment but do not clearly specify with which analysis methods this factor must be calculated (Associacão Brasileira de Normas Téchnicas, 2017; DRR Team Brazil, 2019). Furthermore, the FoS tends to create binary results. It provides with an image of the dam being safe/unsafe. Also the design guidelines such as used in Australia are not providing all of the necessary rules. Brett et al. (2012) describe the FoS to be determined for various stability analyses but only briefly treat the use of reliability analyses or the consideration of probabilistically determined loads. This is especially the case for static liquefaction mechanisms, i.e. failure mechanisms of the undrained type. Here the choice of the strength to test against is vital for the outcome as the strength of the soil varies considerably over the stress path imposed. At this point this mechanism is either neglected or fed with a lot of conservatism (Martin and McRoberts, 1999).

A start has to be made with a better consideration of the failure modes of tailings impoundments to appraise the risk that these dams hold as well as to provide with adequate measures to assure a desired safety level for the future. The objective of this thesis therefore is to construct and implement such a framework. The failure mechanism for which this will be done is static liquefaction as it is most relevant and holds the largest consequences. However, the failure mode of static liquefaction in itself may have various causes and it is necessary to specify which cause is researched. For this thesis, the focus will be with static liquefaction induced by rainfall infiltration. Rainfall is a load that will need to be endured by the tailings impoundment during the entire lifetime. Especially with the idea of climate change and potentially intensifying rainfall storms, this is of special interest. Also in a historical perspective, meteorological loads account for a large part of the tailings dams failures. Rico et al. (2008) note that ‘unusual rain’ has been the cause of 25% of the worldwide impoundment failures and of 35% in European context.

1.3. Main research question

Following this objective, the main research question can be formulated. This research question will be answered throughout this thesis but will be addressed specifically in the conclusion. The main research question reads the following:

”How does the probability of static liquefaction of a tailings dam relate to extreme rainfall events?”

Unlike many other MSc. theses it has been decided not to document with the use of sub-questions. Although at the start of this thesis sub-research questions have been formulated, they became obsolete quickly and at this point the author believes they will confuse the reader rather than to provide with structure. Section 1.4 will elaborate further on the structure of this thesis.

1.4. Document structure

In Chapter 2, a better technical scope will be provided together with some restraints on the methodology to be applied. This will reduce the number of ways in which the main research question may be answered. Chapter 3 provides with a more detailed literature study by discussing observed relations between static liquefaction, rainfall and tailings dams. From the literature study a detailed methodology for the analysis of rainfall-induced static liquefaction is derived which is provided in Chapter 4. Chapter 5 treats the various parameters needed in analysis and the values that these parameters may take. Chapter 6 elaborates on the results from the models implemented and simulations performed. It will discuss the various observations that were done and provide with an explanation of the results. Chapter 7 will conclude on the observed matter and answer the main research question. Chapter 8 provides with recommendations on the model and the proposed framework for future research into (rainfall-induced) static liquefaction of tailings dams.

CHAPTER 2

Scope

In this chapter the scope and restraints of this thesis are further addressed. This will aid the reader in understanding why certain modelling decisions were made and allows for a critical reflection on these decisions. The restraints are formulated on the basis of data availability and the methodology which may be applied. This is done in Section 2.1. After that, in Section 2.2, the failure mechanism will more clearly be defined. Finally, Section 2.3 provides with the choices made on tailings dam configuration and the tailings material.

2.1. Data availability and applied methodology

The amount of methods that are available for the analysis of geotechnical failure mechanisms is extensive. The methodology which may be applied is however restricted by the availability of data, previous research and the right software. For this thesis, the availability of data has been governing in the choice of the methodology. Little data is publicly available on tailings dams as much data belongs to the mining companies and is not shared. Tailings dams are also structures which are not monitored as extensively as other geotechnical works. Moreover, there exists a large variety of tailings types such as.: gold, iron, copper and rare earth metal tailings. Each of these different types show different soil mechanical behaviour and it is unsure how different parameters are related between them.

This thesis will therefore be a modelling research project. This means that a large sensitivity study is adopted in which no specific data from a single site is used. Parameter estimates will be based on a set of data that was documented in literature. Any conclusions within this thesis are therefore based on the model results and may well be considered to be ‘model hypotheses’. Extensive effort has been put into verification of these results with case studies which have been treated in literature.

2.2. Static liquefaction as a failure mechanism

Static liquefaction has been marked as the failure mechanism of interest for this study. There are however many ways in which static liquefaction may be triggered. Static liquefaction is usually represented as an instability envelope. The liquefaction instability envelope presents an area within the stress space of a soil element for which the soil may liquefy (Lade, 1992). There are however various stress paths possible which lead to the instability envelope. See e.g. the work of Olson (2001), of which an important figure is shown in Figure 2.1.

Previously, the event of precipitation was mentioned as the trigger of interest. In essence, rainfall infiltration due to precipitation causes an increase in the phreatic surface. In general, it may therefore be assumed that the deviatoric stress of a soil element within the tailings dam will remain the same whilst the mean effective stress will go down. This is consistent with the stress path shown as A-D in Figure 2.1, and is presented as a horizontal line. The initial distance between the soil stress state and the instability line may be expressed by a factor of safety (FoS). Additional notes on the assumption of a horizontal stress path will also be made in Chapter 3 as the representation of a horizontal stress path upon rainfall infiltration might not be fully accurate.

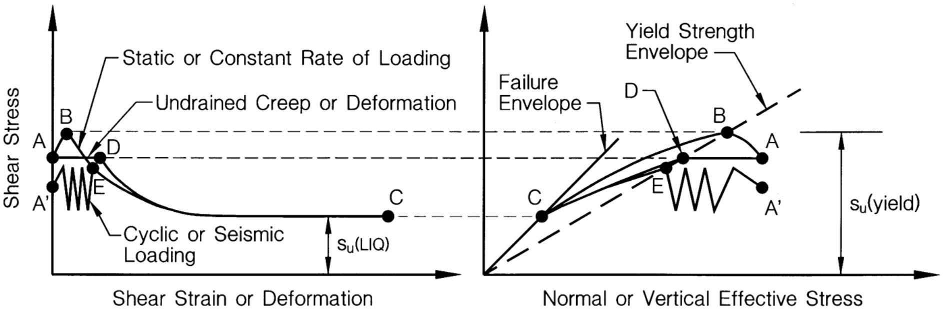

Shear Stress
Static or Constant Rate of Loading
Undrained Creep or Deformation
Cyclic or Seismic Loading
A
B
D
E
A'
C
s_u(LIQ)
Shear Strain or Deformation
Yield Strength Envelope
Failure Envelope
D
B
A
E
A'
C
s_u(yield)
Normal or Vertical Effective Stress

Figure 2.1: The various possible stress paths leading to the instability envelope. The stress path A to B describes shearing loading. A to D follows from rainfall infiltration and A' to E concerns cyclic loading upon an earthquake excitation. Figure from: (Olson, 2001).

A different cause of static liquefaction due to rainfall is overtopping of the impoundment. This will erode the downstream slope of the dam, cause oversteepening and an increase in deviatoric stress whereas the mean effective stress remains constant. The stress state therefore follows a vertical stress path to failure in the right graph in Figure 2.1. This stress path is not provided in this figure. Overtopping will not be treated in this study because of the different modelling implications.

2.3. Tailings dams and tailings material

This thesis will focus on dams of the upstream type as they have the largest record of failures. At this moment these dams are taken out of operation. Therefore, this study will pin itself down to decommissioned dams. Concretely, that means some assumptions are made which are:

- the tailings material is fully consolidated;
- shallow slip surfaces are unlikely because of vegetation that has grown on the surface;
- the tailings pond may have evaporated over time and is therefore no longer a correct measure of the boundary condition to the phreatic surface inside the dam.

In addition to that, no construction defects in the dam will be considered. The Brazilian dam collapses have shown that construction history is often an important prerequisite for failure. However, there are many possible ways in which construction may go wrong. Also as there is no framework yet available to assess the standard cases, the defects will not be considered.

With regards to the tailings type, the study will focus on iron ore tailings because of the high frequency of iron ore mines in Brazil. Especially with regards to the Minas Gerais state in which both failures, as mentioned in Chapter 1, have occurred (Clout and Manuel, 2015). The precipitation data used in the analysis will also be from the Minais Gerais state.

2.4. Conclusion on the scope

Reflecting on the aforementioned, the main research question as provided in Chapter 1 is further narrowed down by assuming only static liquefaction events due to rainfall infiltration and to consider decommissioned iron ore upstream tailings dams, with special reference to those in Brazil. The main research question will be answered through a modelling approach and this thesis is not intended to be performing extensive data analysis of existing tailings dams.

CHAPTER 3

Tailings dams, static liquefaction and rainfall: the relations

In Chapter 1, it was mentioned that the static liquefaction of tailings dams is a frequently occurring failure mechanism. Additionally, rainfall was named as a common cause of the static liquefaction of tailings dams. The purpose of this chapter is to identify the main relations between static liquefaction, tailings dams and rainfall to be able to set up an analysis methodology for the rainfall-induced static liquefaction of tailings dams. The following sections will describe the relations between: static liquefaction and tailings dams (Section 3.1); static liquefaction and rainfall (Section 3.2); and tailings dams and rainfall (Section 3.3). In Section 3.4 case studies of the static liquefaction of tailings dams are described where rainfall played the most important role. Some prior knowledge of the concepts of static liquefaction, tailings dams and unsaturated flow is assumed for conciseness of the material.

3.1. Static liquefaction and tailings dams

Davies (2002) estimated, based on some simple database analysis, that the annual probability of failure of tailings impoundments is between 1:750 and 1:1750. A large part of the failures in this database involved static liquefaction of the dams. Generally, common water retention dams and other geotechnical structures have far lower probabilities of failure. Therefore the question arises as to why tailings dams are relatively prone to failure, and especially static liquefaction. This section addresses a few frequently named causes in literature:

- the unique typing of tailings material;
- methods of depositioning and resulting void ratios;
- the configuration of upstream tailings dams.

3.1.1. Tailings materials and its typing

In general, a soil is liquefiable if: it contains large amounts of finer material, it is non-cemented, it is non-cohesive and has a high void ratio. The large amount of fines reduces permeability upon loading, leading to excess pore pressures. Furthermore, Lade and Yamamuro (2011) argue that for a sufficiently small particle size gravity is no longer the governing force in terms of packing of the soil upon deposition. Van der Waal attraction and ‘double-layer repulsion’ instead cause high void ratios to occur. These high void ratios increase the potential of the soil to liquefy.

Tailings are the left-over rock flour after the process of mineral extraction at a mine. The term ‘slimes’ or ‘slimes tailings’ is used to describe the portion of the tailings smaller than the US sieve no. 200 which is equivalent to a grain size of 0.074 mm (Vick, 1983). The grains with a grain size larger than 0.074 mm are called ‘sands’ or ‘sand tailings’. As argued by Shuttle and Cumming (2007), especially the slimes are prone to liquefaction. This is indirectly also addressed by Lade and Yamamuro (2011) who mention that the US sieve 200 is the boundary from which the earlier mentioned attraction bonds start to occur. Also Vick (1992) argues that the slimes tailings are the liquefiable material in the tailings impoundment whereas the sand tailings will form a non-liquefiable outer shell upon depositioning.

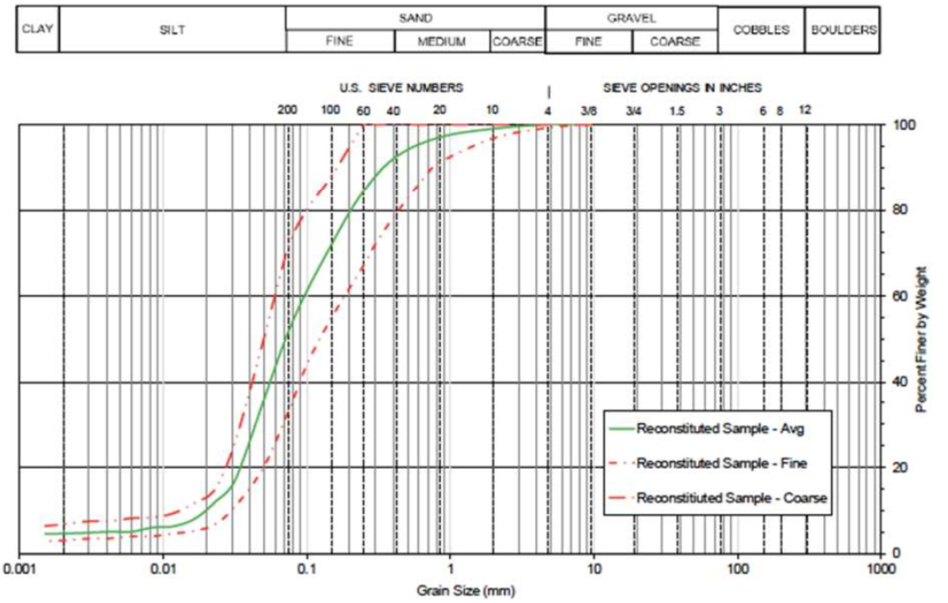


Figure 3.1: A sieve curve of the tailings of the Feijao dam. Showing the weight percentage passing on the vertical axis. Figure from: Robertson et al. (2019).

In other literature however, the definition of sand and slimes tailings with their potential to liquefy is not consistent. In both failure reports of the Brazilian dams, the slimes are seen as plastic/cohesive tailings which will not liquefy and liquefaction is attributed to the sand tailings (Robertson et al., 2019; Morgenstern et al., 2016). The 'berm' is the term used for the zone of tailings near the dam face which have been compacted and will show dilative behaviour upon shearing. To overcome this confusing use of terminology, the terms adopted in this study will be 'liquefiable tailings' and 'non-liquefiable tailings'. The liquefiable tailings are those with a grain size smaller than the US Sieve 200, a limited amount of plastic minerals and a high enough void ratio. The non-liquefiable tailings are those that have a too large grain size, contain high amounts of plastic minerals or are compacted and will show dilative behaviour upon shearing. As addressed earlier, to determine the liquefiability of tailings, it has to be tested whether tailings will have a large percentage of fines, are non-cohesive and non-plastic.

With regards to the grain size, a sieve curve of the tailings of the Feijão dam used in laboratory experiments is visible in Figure 3.1. In this figure, it is shown that the average sieve curve holds around 50% of the tailings smaller than the US 200 sieve. Of iron ore tailings no other reliable sieve curves have been found. In general, it may be noted that the amount of fines is variable as was also shown in Qiu and Sego (2001). Gold, coal and copper tailings showed varying fines percentages between roughly 20 and 80 %. Cambridge et al. (2018) mention that the particle size grading depends on fineness of milling, mineralogy, weathering, separation process etc.

In addition, deposits need to be non-cemented and non-cohesive in order to be able to liquefy. Tailings deposits are of young geological age and will mostly be non-cemented. In the Minas Gerais state the only plastic mineral that is present in tailings is kaolinite with a weight percentage of, on average, 2.1 % (Clout and Manuel, 2015). Robertson et al. (2019) reports different mineral values for the tailings of the failed dam, but differentiates between slimes, fine tailings and coarse tailings. The slimes hold around 6-9 % kaolinite, whereas this is only 0.6% for the coarse tailings. From this, it may be assumed that the amount of plastic minerals increases with increasing distance from the dam in upstream direction. It may also be concluded from this failure that the amount of plastic minerals was too low to prevent liquefaction.

Key argument to take away from this section is that due to: the tailings' small grain size, large amount of non-plastic fines and the fact that the deposit is of young geological age, ideal circumstances are created for deposits to form which are prone to liquefaction.

3.1.2. Methods of depositioning and occurring void ratios

Tailings dams are geotechnical structures which grow over time. Whilst more minerals are being mined, more tailings are produced and these are deposited in the same or in several tailings storage facilities. Spigotting is the most frequently applied deposition method for tailings. With spigotting, the tailings material is released in slurry form along the top of the tailings dam after which it flows towards the tailings pond away from the crest. The spigotting method is schematically drawn in Figure 3.2.

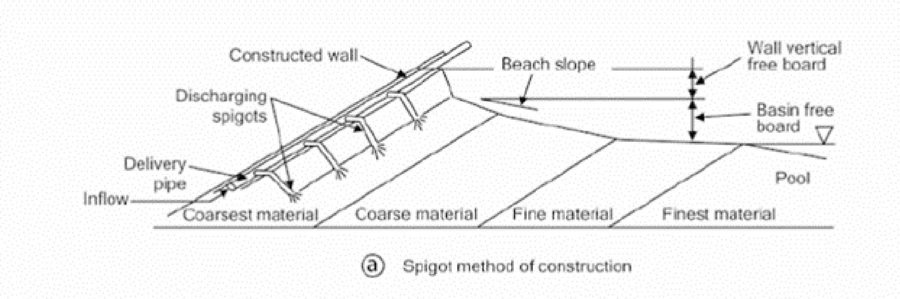


Figure 3.2: Tailings being spigotted along roughly equally distributed outlets along the dam crest of a tailings pond. The material flows upstream towards the tailings pond. The finer material settles near the pond. The coarser material will deposit onto the tailings beach. Figure from: Blight (2010).

During the spigotting process, part of the grains in the slurry will deposit onto the 'tailings beach'. The tailings beach is that part of the tailings impoundment which is not covered by the tailings pond. The fraction of the tailings that will deposit onto the beach depends on several factors, including: total discharge, slurry density, grain sizes in the tailings slurry and the slope angle of the tailings beach. In general, the coarser fraction of the tailings are deposited near the dam crest whereas the finer fraction will settle in the tailings pond as the flow has lost most of its turbulence there. The grain size therefore decreases with upstream distance from the dam. This process is called upstream fining and it is often represented by the master profile of a tailings dam (Blight, 1994). The master profile describes distributions of grain size, shear strength, permeability, and other soil properties within the tailings dam as a decreasing hyperbolic function with increasing distance from the dam crest. An example profile can be seen in Figure 3.3 where the shear strength is highest near the dam face because of the coarser material deposited there. It was addressed earlier that non-liquefiable coarse tailings are positioned near the dam face. It is thus expected that there is a non-liquefiable shell which provides stability to the weaker tailings. Furthermore, upstream fining causes the permeability in this zone to be higher than further upstream. This lowers the phreatic surface near the dam face, increasing the stability in that zone (Robertson, 1987).

However, the spigotting method may also be disadvantageous to the stability of the dam. First, depositioning of tailings in slurry form results in a loose structure of the material upon settling. The tailings material is usually not compacted, increasing the potential for static liquefaction mechanisms to develop. Also, there will be some immanent horizontal layering of the soil deposit. This layering is disadvantageous when considering that the tailings deposited are a direct reflection of the material that is mined. Having e.g. a large clay-infilled joint in the rock will cause clay minerals to be present in the tailings. In its turn, an impermeable layer will develop within the dam. This leads to perched water tables upon rainfall and deposition. Blight (2010) also highlights the effect of layering: 'It is suspected that liquefaction failures of hydraulic fill tailings storages usually initiate in relatively thin, high void ratio layers that have formed during periods when the relative density of the tailings slurry

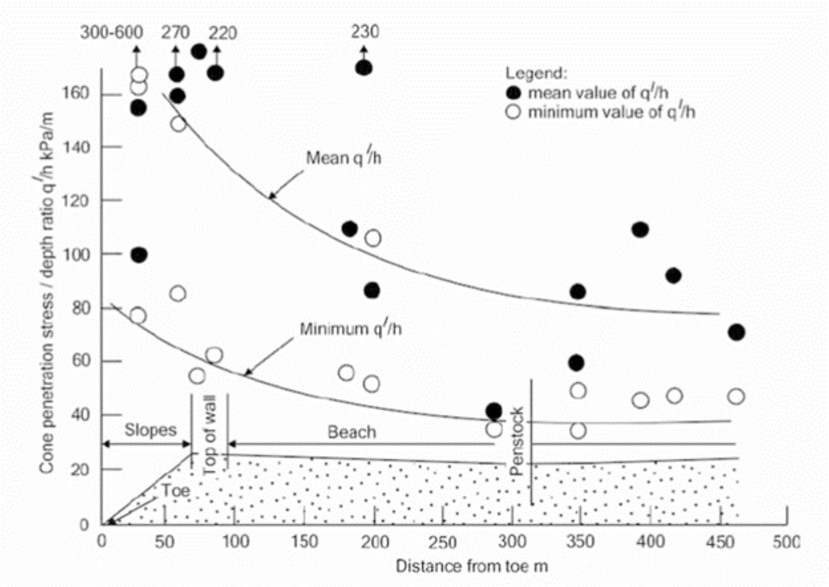


Figure 3.3: Visualisation of possible strength deterioration functions with the distance from the toe. Projected are values of the cone penetration resistance. Figure from: (Blight, 2010).

was unusually low or the rate of rise was unusually high'. Such a high rate of rise may not only be disadvantageous to the realised void ratios within the impoundment but additionally causes increased excess pore pressures inside the material.

The continuous deposition of material in a tailings impoundment causes the consolidation of tailings dams to converge from the consolidation of regular geotechnical earthworks following a phased construction. The continual deposition results in an increase in the load as well as an increase in the drainage length over time. Gibson (1958) presented some relations for consolidation of deposits with increasing thickness. The consolidation ratio in these impoundments are lower than would occur applying a phased construction. The increased excess pore pressure because of deposition reduces stability of the tailings impoundment.

A way to promote consolidation is by increasing time between depositioning phases. Spigotting is usually not applied along the entire dam boundary simultaneously but will be alternated along stretches of the tailings dam. In this way, the tailings are allowed to dry and infiltrated water in the upper zone is allowed to evaporate. Drying reduces water content and, because of the coupled influence of suction on the soil skeleton, the void ratios therefore decrease. Blight (2010) notes that by allowing the tailings to dry, upstream constructed dams are more stable and less likely to fail. Upstream raised dams have therefore successfully been built in water deficit areas. In the Minais Gerais state, precipitation rates are approximately equal to the evaporation rates of on average 1400 mm/a (de Assis Diniz et al., 2018; Giongo et al., 2018). This effect might therefore not be achieved due to the additional process water added to the impoundment.

In conclusion, spigotting is the most widely applied form of depositioning of tailings in the impoundment. Spigotting results in upstream fining of the material which promotes stability as, near the dam face, strength is increased and the phreatic surface is lowered. Yet, spigotting will also lead to unfavourable layering and excess pore pressures within the deposit. The latter effect may be alleviated by increasing time between depositioning phases, allowing the tailings to dry. This effect might not be achieved in Brazil, where precipitation rates equal evaporation rates.

3.1.3. The configuration of upstream tailings dams

Upstream raised tailings dams are, among centred and downstream dams, one of the three common configurations of tailings impoundments. Because of its economic efficiency, the upstream configuration is the most frequently applied one of the three. Within the failure history of tailings dams especially the upstream ones are abundantly present and this section will elaborate on why that is the case.

Around the 1990's there is a clear consideration of static liquefaction mechanisms in the design of upstream tailings dams. Examples of this are the article by Martin and McRoberts (1999) and the Karl Terzaghi Lecture by Ladd (1991). The research work at that time has manifested itself into a few important design guidelines to ensure stability of an upstream tailings dam. From Martin and McRoberts (1999) the following guidelines are cited to be key in having a stable upstream dam:

1. 'spigotting of a wide, sand (drained) tailings beach from the embankment crest';
2. 'avoiding situations whereby the dam slope is underlain by fine tailings (slimes) deposited within the water pond';
3. 'prevention of seepage emerging on the dam face';
4. 'having a well-drained foundation'.

Other design principles are also given, but seem to be alternative formulations of the above four principles. The design principles aim to avoid high saturation of the impoundment which is directly linked to possible liquefaction mechanisms. Also high ratios of deviatoric stress over confining stress are to be avoided as the distance from the stress point of a soil element to the instability envelope is decreased (see Chapter 2). In addition, the formation of a wide beach is argued to be necessary. The beach widths is in this case directly related to the zone of the non-liquefiable tailings near the dam face. Figure 3.4 shows two cross-sections of upstream raised tailings dams with different beach widths. Note that here the sand tailings are associated with non-liquefiable material.

Many occurrences of upstream tailings dams failures may be attributed to not conforming to one or more of these design principles. A perfect example of liquefaction of a tailings dam in which these rules were violated is the failure of the Fundão dam as introduced in Chapter 1. The investigation report by Morgenstern et al. (2016) names the following incidents that led to failure:

1. 'damage to the original starter dam that resulted in increased saturation';
2. 'deposition of slimes in areas where this was not intended';
3. 'structural problems with a concrete conduit that caused the dam to be raised over the slimes'.

In fact all of the design principles have been violated. The construction incidents either caused the degree of saturation of the embankment to increase or to make sure that slimes and finer deposits were situated near the dam or even under it, increasing the ratio of deviatoric stress over confining stress in these weaker deposits. It is therefore stressed that the design principles should be adhered to in the design of an upstream tailings dam. Furthermore, these guidelines could serve as the basis for a safety assessment in which it is (quantitatively) verified whether the guidelines are met.

3.1.4. Conclusions on the relations between tailings dams and static liquefaction

Based on a literature study, various aspects of tailings dams have been identified which makes them prone to static liquefaction. The first is the nature of tailings being a fine, non-cohesive material. Second, the common method of depositioning, spigotting, allows for loose configurations of the material. This is often combined with high excess pore pressures due to a slow consolidation process. Finally,

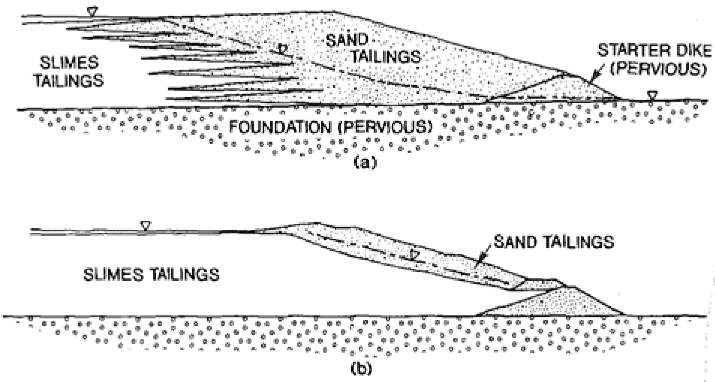


Figure 3.4: Two tailings dams cross-sections with different sand tailings beach widths that may in this case be associated with the width of the zone of non-liquefiable materials. The upper cross-section shows a good configuration of the beach. The lower cross-section is poor practice of tailings dam construction. Figure from Vick (1992).

for upstream raised dams it is difficult to avoid high saturation of the material as well as high ratios of deviatoric stress over confining stress near the dam face. Both will reduce stability of the dam.

To quantitatively judge whether the deposited tailings are able to liquefy, in-situ or laboratory testing may be employed. Laboratory testing provides more reliable results in the determination of the instability envelope in stress space. However, soil disturbance during sampling is a large issue as the soil is non-cohesive and has high void ratios (Shuttle and Cumming, 2007). With regards to in-situ tests, examples of measured relations between normalised cone penetration factors and the liquefiability of a soil deposit can be found in: Olson and Stark (2003), Jefferies and Been (2016) and Robertson (2010). Determination is therefore only possible through correlations which is argued to be a weakness of in-situ testing.

3.2. Static liquefaction and rainfall

The effect of rainfall on the stability of slopes has long been recognised. Considering static liquefaction, some overview is provided by Chu et al. (2003) in which it is described that rainfall infiltration in slopes causes reduction of the mean effective stress while the deviatoric stress is maintained. In several laboratory tests in which this stress path was imposed, it was observed that until the rate of deformation started to increase, indicating instability, no excess pore pressures developed. This proves that the stress path may be fully drained until the moment of failure. An example of such a failure is the Aberfan coal tip disaster as also described by Chu et al. (2003). This failure behaviour was additionally experimentally proven in the renowned paper by Eckersley (1990). Rainfall is therefore strictly speaking not the trigger. Instead, it moves the stress state from being outside the instability envelope into the instability envelope. The final trigger of instability failure is in that sense not considered which is also mentioned by Lade (1992) who notes that, as long as the soil behaves in a drained manner, a stable stress state can also exist within this instability envelope. It is however assumed at this point that, if a stress state occurs within the instability envelope, the runaway failure will occur.

Chu et al. (2003) mention that the stress path of reducing mean effective stress and constant deviatoric stress has ‘seldom been examined’ in liquefaction analysis. This is also true for larger scale events such as the static liquefaction of a tailings dam. In this section, a hypothesis is formulated on how a slope may liquefy upon rainfall by first establishing the relation between rainfall infiltration and soil strength. Followed by a description of the liquefaction of unsaturated soils. Also the study of stability of natural slopes upon rainfall and similarities and differences with liquefaction of tailings dams are discussed.

3.2.1. Rainfall infiltration and the effect on soil strength

In the unsaturated zone of a soil the continuum consists of three phases: solid grains, liquid water and gaseous air. With increasing distance from the phreatic surface, the water content inside the pores decreases. Pore water is replaced with pore air. As a result, water lenses will form in between the grains because of the adhesive properties between water and solids. These water lenses are able to resist tensile forces and may well be considered a form of cohesion. This cohesion is however not a strength property of the soil. Wetting of these pores upon rainfall infiltration will saturate the pores, meaning that the earlier mentioned cohesion bond is lost. Unsaturated soils are therefore ‘stronger’ than saturated soils. This effect was captured into e.g. Bishop’s equation for effective stress (Blight, 2013):

$$\sigma' = (\sigma - u_a) + \chi\,(u_a - u_w) \tag{3.1}$$

with u_w the pore water pressure [kPa] and u_a the pore air pressure [kPa]. If it is assumed that air is infinitely mobile, the pore air pressure is consistent everywhere throughout the unsaturated soil and equals the atmospheric pressure (Kim et al., 2012; Ng and Shi, 1998). This means that Equation 3.1, reduces to:

$$\sigma' = \sigma + \chi\,(-u_w) \tag{3.2}$$

The stress parameter χ is commonly set equal to the relative/effective degree of saturation: S_r (Bolzon et al., 1996; Kim et al., 2012).

Bishops’ equation for effective stress does not only describe strength loss due to a loss of suction, but also includes any loss of strength due to mounding of the water table. When rainfall infiltration water leads to a rise of the phreatic surface, the pore water pressures in the soil below that will go up. In its turn, the effective stress decreases, reducing the soil strength below the water table.

The process of rainfall infiltration may be described using simple wetting front models such as that of Green and Ampt (1911). More common now is the use of the convection-diffusion equation for flow through unsaturated soils: the Richards equation. This equation will be treated in more detail in Chapter 4.

3.2.2. Static liquefaction of unsaturated deposits

With increasing suction, the water content inside the soil decreases. It may be expected that when the degree of saturation is sufficiently low, no excess pore water pressures are generated upon shearing as first the pore air will dissipate, leading to volume change upon shearing. The liquefaction of unsaturated soils has been studied within the context of methane gas in deltaic deposits. Grozic et al. (1999) mention that: ‘The minimum (or “cut-off”) initial degree of saturation that samples must be at to strain soften is approximately 90%’. The strain softening mechanism being liquefaction of the soil here. Although it is unsure how the behaviour of these gassy sands compare to that of tailings, it provides with a rough idea of what degree of saturation is needed before liquefaction of the soil may occur.

3.2.3. Stability of slopes upon rainfall infiltration

Contemplating on how suction influences strength and how the degree of saturation affects the potential of a soil to liquefy, allows to zoom out to the issue of liquefaction of unsaturated slopes. Static liquefaction is an undrained mechanism where pore pressures upon loading are not able to dissipate quickly enough. Studies of undrained failure mechanisms in unsaturated slopes are uncommon. There is a however a wide variety of literature available on the stability of unsaturated slopes where the material does not generate excess pore pressures upon shearing, assuming drained failure. Some main conclusions in this field of research will be presented first.

Rahardjo et al. (2007) performed a large sensitivity analysis on the drained stability of unsaturated slopes. Parameters included in the sensitivity analysis were the: saturated coefficient of permeability, rainfall intensity, rainfall duration, slope angle, slope height and depth of the initial water table. Although not specifically mentioned, the value of the saturated permeability dictates the failure mechanism. For homogeneous slopes this is either failure due to a loss of suction or due to mounding of the water table. For low permeability soils, which tailings generally are, the failure mechanism was a shallow slip surface due to the loss of suction above the water table. Short term rainfall, under 24 hours, did not contribute much to lowering of the factor of safety as the low permeability of the soil results in relatively high run-off and little infiltration. The effect of slope geometry, slope angle and the initial water table determine the initial factor of safety and, in this way, the safety margin to failure by rainfall infiltration. High slopes with large slope angles and a low depth of the initial water table are most susceptible to failure by rainfall infiltration.

Ng and Shi (1998) emphasise that the failure mechanism is a function of the imposed water table. When the water table is situated closer to the surface level, mounding of the water table contributes more significantly in the failure. In addition, perched water tables may severely decrease strength. This is a valuable remark with reference to the possibilities of the same perched water tables as highlighted in section 3.1.

Although little research is available on the rainfall-induced liquefaction of unsaturated slopes, the previous observations allow for the formulation of a hypothesis on the stability of a liquefiable tailings dam. The low permeability of the tailings will cause loss of suction to be the governing mechanism in a shearing failure, creating a tendency for a shallow failure. However, due to the low strength of saturated liquefiable tailings, it is expected that there is the tendency to have a deep failure surface as the liquefiable tailings material is situated near and below the phreatic surface. In this way, mounding of the water table might sort a large effect. An increase of the water table will lead to higher saturation of the material and also a reduction of the vertical effective stress below the phreatic surface. Both will reduce the strength of the slope. Possibly a mixed mechanism of loss of suction and mounding of the water table is therefore governing in failure.

The horizontal stress path as depicted in Chapter 2 and highlighted at the start of this section are therefore not fully correct. The unsaturated material will have some relatively high shear strength that is further improved by suction. When this suction is lost, strength is lowered above the phreatic surface. Redistribution is therefore needed along the entire failure plane. This redistribution will cause shear stresses to increase in the potentially liquefiable material. Furthermore, mounding of the water table reduces the mean effective stress of the saturated liquefiable soil. Therefore the stress path following rainfall infiltration will likely not be fully horizontal but will be following a slight increase of the shear stress.

3.2.4. Conclusion to the relation between static liquefaction and rainfall

In conclusion, if considering a tailings dam or slope, the liquefaction model can be applied to the tailings that are below the line which marks a degree of saturation of 90%. Above this line, the soil is too unsaturated to generate excess pore pressures upon shearing and a drained analysis is more appropriate. In ordinary slope stability analysis a drained material is applied homogeneously over the slope which causes a loss of suction to be the governing failure mechanism and the tendency for shallow slip surfaces to form. As the liquefiable material is weaker than material in drained response, it is expected that this might not be true for this study of tailings dams and a mixed mechanism of the increase of the phreatic surface as well as a loss of suction could be the governing failure mechanism. This mixed mechanism causes a relatively unknown stress path which will be relatively difficult to model and should be considered with caution.

3.3. Tailings dams and rainfall

This section elaborates on the relation between tailings dams and rainfall or, more fundamentally, on the water cycle of tailings dams and the position that rainfall takes within this cycle. First, the global water cycle will be elaborated upon, focusing mostly on decommissioned dams. This is followed by the effect of rainfall on the water cycle.

3.3.1. The global water cycle of tailings dams

With slurry deposition of tailings, a large amount of water is brought to the top of the tailings dam. Part of this water will infiltrate into the soil. However, most of it will run-off into the tailings pond, meaning that the amount of deposition water exceeds the infiltration capacity of the soil.

The level of the tailings pond together with the infiltration of the process water will shape the phreatic surface within the dam. The water in the pond evaporates or is removed with a decant system. The water inside the tailings dam seeps through the foundation of the dam or exits at the dam toe. This makes the basic water cycle of a tailings dam. These elements are also illustrated in Figure 3.5.

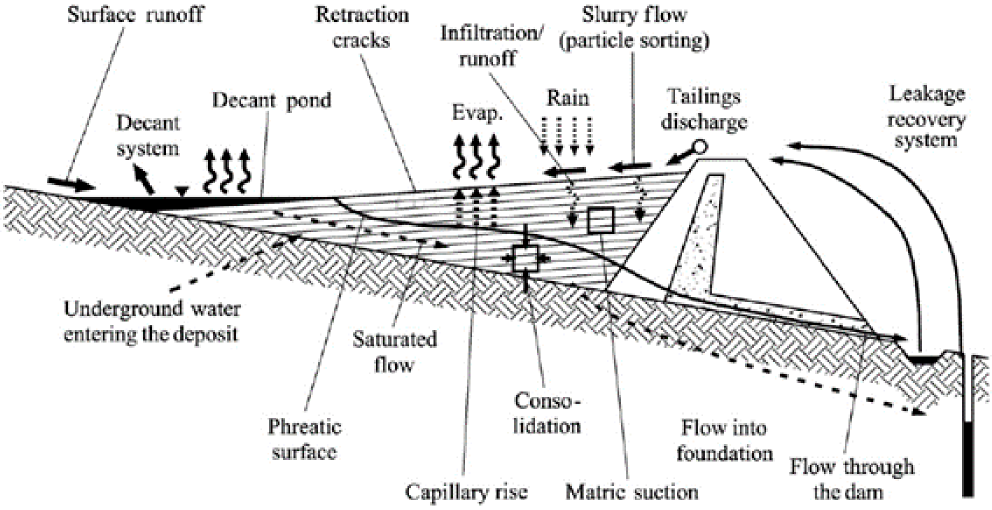


Figure 3.5: The global water cycle of a tailings dam. Figure from: Zandarín et al. (2009).

The water cycle shown belongs to an operational tailings dam. Blight (2010) notes that for decommissioned tailings dams the flow of water into the system consists of rainfall infiltration and possibly irrigation water for surface vegetation. The outflow consists of surface run-off, evapotranspiration and seepage. It is noted that when operations cease the phreatic surface will subside but, depending on various properties, it may 'last several years and possibly as long as two or more decades' to establish steady-state conditions. In addition to this low rate of phreatic subsidence, there is the possible high air entry value (AEV) of the tailings which causes the tailings to be nearly fully saturated, even under high suction values. Capillary rise may be as much as 30 meters for some kinds of tailings (Blight, 2013). Blight (2010) reports on the Simmergo dam failure where failure occurred after a quickly rising phreatic surface. The dam had been decommissioned but was taken back in operation after a period of 20 years. The high AEV value caused the phreatic surface to rise with 20 meters in a single month, eventually leading to failure. Also Zandarín et al. (2009) stress the importance of capillary rise in the stability of tailings dams.

3.3.2. The influence of rainfall on the water cycle

Tailings deposition by spigotting can be seen as a very intensive rainstorm which causes ponding. One could argue that the spigotting that has been performed at the end of the operational period of the dam is essentially the survived design load and therefore the decommissioned dam could potentially withstand extreme rainfall loads.

However, spigotting is usually intermittently applied along stretches of the dam. Other parts of the dam may have a reinforcing effect and cause global stability. This is not the case for rainfall as precipitation occurs everywhere over the impoundment. Additionally, some tailings dams are located in natural depressions like valleys or along hillsides. Any runoff in this catchment area is able to reach the tailings impoundment and is therefore run-on for the tailings dam. Furthermore, the infiltration in the catchment area will influence the groundwater head that is present on the sides of the tailings impoundment. This acts as a boundary condition for the head inside the facility as also applied by Robertson et al. (2019).

3.4. Cases studies of rainfall-induced liquefaction of tailings dams

Considering the coincidence of the effects mentioned, Rico et al. (2008) provides a statistical analysis of tailings dam failures. In worldwide context, 25% of the tailings dams failures were related to extreme meteorological events. It is not discussed how many of these dam failures have been static liquefaction.

The ICOLD Committee on Tailings Dams and Waste Lagoons (1995-2001) (2001) also reports on a number of tailings dam failures by rainfall. It is however hard to isolate those events that have been caused solely by rainfall infiltration mechanisms as usually failure is a combination of various issues. Of special significance are the following failures in the bulletin:

- No. 220, Sgurigad - Western Bulgaria: ‘a rise in pond level after three days of heavy rainfall caused a loss of stability and flow slides of the tailings, although the dam was not initially breached.’
- No. 2, Kennecott - USA: ‘Breach of the embankment triggered a tailings flowslide. Rainfall may have increased dike saturation and minor shearing may have initiated the failure.’
- No. 113, Southwest - USA: ‘After an unusually heavy rainfall, an embankment slope failure breached the dam, no slimes were released.’
- No. 145, UK: ‘A slide occurred in the downstream slope after a period of heavy rain and one week following widening of the dam crest by dumping of uncompacted mine waste fill.’

The failure numbers were directly retrieved from the report to provide proper reference. These were failures in which rainfall was the governing trigger rather than part of a combination of various causes. Liquefaction failures due to rainfall are the result of a complex interaction mechanism between the tailings and the embankment that encapsulates it. Possible loss of suction or increase in the phreatic surface within this embankment may cause it to breach as was indeed hypothesised in the study of the relation between static liquefaction and rainfall.

Reflecting on the various relations established, it has become clear how to assess if a tailings dam will liquefy upon rainfall infiltration. Essentially, the following questions will need to be answered:

1. Are the tailings potentially liquefiable and where are these liquefiable tailings located in the impoundment?
2. What are the in-situ stresses acting on the tailings in the impoundment and are they critical with respect to potential instability?

3. Where is the phreatic surface located and where are saturated tailings to be found?
4. If a rainfall shower is imposed on the tailings is the loss of suction and/or the rise of the phreatic surface large enough to trigger failure?

The first two questions relate to a correct spatial characterisation of the impoundment as well as a proper consolidation analysis to determine in-situ stresses and material parameters. The third question (and part of the fourth) are answered by an implementation of a geohydrological analysis in which the position of phreatic surface as well as the effect of the infiltration of rainfall water are determined. Finally, a stability/liquefaction analysis shows whether the imposed conditions are sufficient to trigger static liquefaction. Summarising, four types of analyses are needed to correctly determine if an impoundment will fail or not. These different parts have been cast into an analysis methodology which is treated in chapter 4.

CHAPTER 4

Analysis methodology for rainfall-induced static liquefaction of tailings dams

In Chapter 3 it was proposed to use four analysis types to determine if rainfall may trigger a static liquefaction event. These analysis types are: spatial characterisation, consolidation analysis, geohydrological analysis and the liquefaction analysis. Combined, these analyses may provide with the margin to failure of a tailings dam. Reflecting on the main research question, from this margin to failure, a probability of failure must be estimated. In this chapter, a novel approach will be proposed which allows for the calculation of the probability of rainfall-induced static liquefaction. The probability of failure may be translated to a return period of failure which is a better parameter for demonstration purposes and is therefore applied in this study. The methodology used is visualised in the flowchart of Figure 4.1.

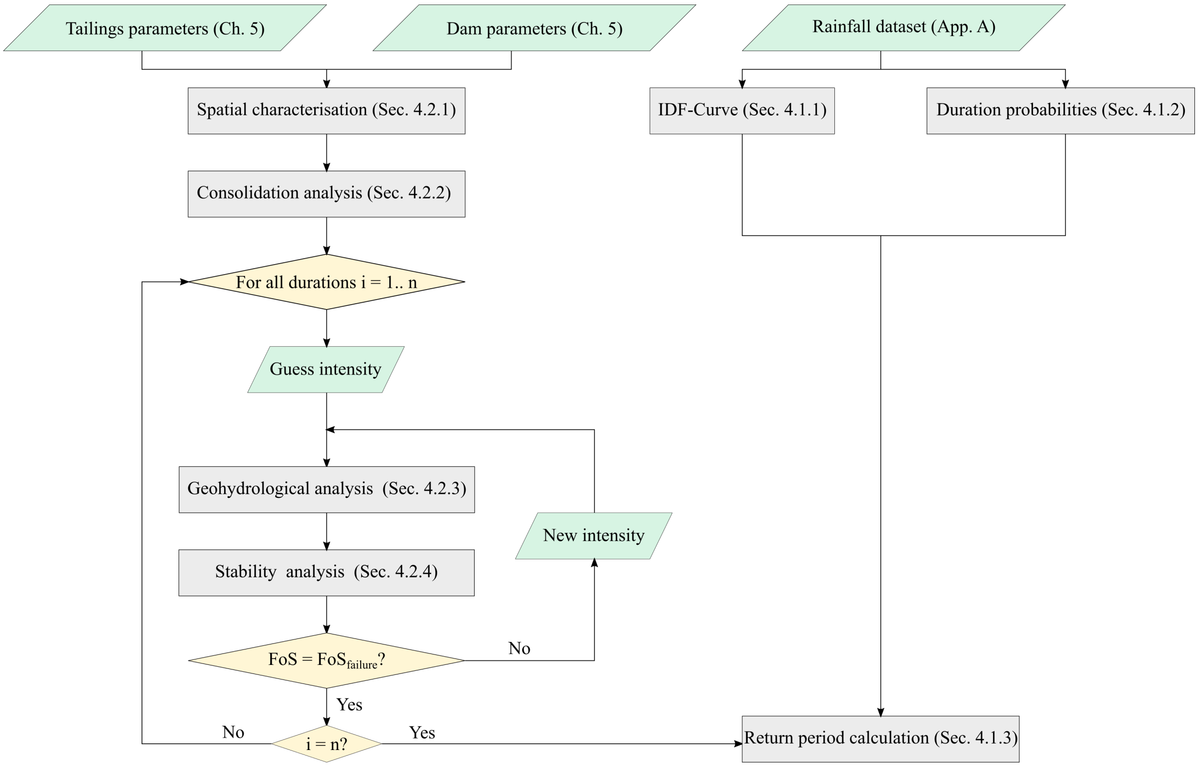

Tailings parameters (Ch. 5)
Dam parameters (Ch. 5)
Rainfall dataset (App. A)
Spatial characterisation (Sec. 4.2.1)
IDF-Curve (Sec. 4.1.1)
Duration probabilities (Sec. 4.1.2)
Consolidation analysis (Sec. 4.2.2)
For all durations i = 1.. n
Guess intensity
Geohydrological analysis (Sec. 4.2.3)
New intensity
Stability analysis (Sec. 4.2.4)
FoS = FoS$_{failure}$?
No
Yes
No
i = n?
Yes
Return period calculation (Sec. 4.1.3)

Figure 4.1: The flow chart of the final analysis methodology.

The flowchart shows two branches in the analysis. The left branch concerns the calculation of the needed intensity to failure for various durations of rainfall. At first, the geometry of the tailings impoundment and the needed input parameters (e.g. strength and hydraulic conductivity) are initialised in the spatial characterisation. After that, a consolidation analysis is performed for an accurate resemblance of the construction history. Then, for each duration, a guess rainfall intensity is applied in the geohydrological analysis where the pressure head and relative degree of saturation (after Equation 3.1) are calculated.

These results are then applied as input to the stability analysis. The stability analysis will present a factor of safety to instability of the tailings dam. It is then evaluated whether this factor of safety equals the failure factor of safety. If not, a new intensity is guessed based on an iterative algorithm and the geohydrological and stability analysis are performed again. If the intensity has been found that provides a factor of safety equal to the failure factor of safety, then the procedure is performed again for a new duration of rainfall. If the failure intensities for all durations have been found, those intensities are used in the return period calculation.

In the right branch, the rainfall spectrum is probabilistically characterised. This allows for the calculation of the return period of failure given the failure intensities for each duration. First the intensity-duration-frequency (IDF) curve is formed based on the rainfall dataset. From this the probability of occurrence of the failure intensity for each duration can be calculated. Furthermore, the probability of occurrence of each duration of rainfall is calculated. These probabilities together with the failure intensities allow for the calculation of the total return period of failure.

The proposed methodology in this chapter has been the result of an elaborate literature study on the various possible methods of analysis available. This has lead to a general framework of analysis which is treated in Appendix B. In addition some earlier phases of analysis are also treated in this appendix. In this chapter, the various parts of the final methodology will be applied in more detail. These parts and the respective sections have been indicated in Figure 4.1. Additionally, a validation of the geohydrological and stability analysis is performed in Section 4.3.

4.1. Probabilistic definition of the rainfall spectrum

4.1.1. IDF-curve: Probability of each intensity occurring conditional to the duration

Rainfall may be described by an intensity and the duration for which that intensity lasts. Each combination of intensity and duration will have a certain frequency of occurrence. In hydrology this is often expressed as the probability or return period of occurrence. The IDF-curve is a contour plot, describing this return period for all possible combinations of intensity and duration. In the assembly of the IDF-curve, first for each duration, the return period of each intensity occurring is represented by a probability distribution. Within rainfall hydrology, these probability distributions are usually Gumbel or Generalised Extreme Value (GEV) distributions. Both distributions are extreme value distributions as the intensities of interest are often associated with high return periods of occurrence. Therefore, extrapolation on the basis of the extreme values is necessary. When the distributions have been fitted, the IDF-curve then forms contours along the various durations of rainfall for the intensities with equal return periods of occurrence.

The cumulative distribution function of the Gumbel distribution is given by:

$$F(x) = \exp(-\exp[-(x-u)/\alpha]) \tag{4.1}$$

where u is the expected value or location parameter and may be associated with the intensity that has a return period of 1 year. α refers to the scale parameter and it determines how much the intensity increases with increasing return period. The GEV distribution adds on the Gumbel distribution by the application of the shape parameter ξ:

$$F(x) = \exp\left[(-(x-u)/\alpha)^{-\xi}\right]. \tag{4.2}$$

The GEV distribution is equal to the Gumbel distribution when this shape parameter equals zero.

The distributions represent the extremes of a dataset of realised precipitation intensities. In this study, daily precipitation depth data from 45 weather stations in the Minas Gerais state, Brazil, during a period from 1979 to 2014 was extracted from the database of NCEP (2020). The dataset provides with a total of 1620 station years of precipitation data. The acquired data in this method was assumed to be stationary and spatially homogeneous. The validation of these assumptions is treated in Appendix A. For each duration of rainfall: 1,2,3,... days, the data was resampled to provide with the average precipitation depth in meters per day. This is then a measure of the rainfall intensity in meters per day. For each duration, a new dataset was formed containing the annual maxima as the extremes for extrapolation. The probabilistic distribution is then fitted to these annual maxima.

In this fit, for each duration, the values of the different parameters for the distributions (u, α, ξ) have to be estimated. For the Gumbel distribution, both the method of moments and the maximum likelihood (ML) estimation were applied for the parameter estimation. The method of moments uses the following equations for the parameter estimation:

$$u = \hat{\mu} - \gamma\alpha = \hat{\mu} - 0.5572\alpha \tag{4.3}$$

$$\alpha = \sqrt{6}\hat{\sigma}/\pi \tag{4.4}$$

with N the sample size and $\hat{\mu}$ is the sample mean which equates to:

$$\hat{\mu} = 1/N \sum_{i=1}^{N} x_i. \tag{4.5}$$

$\hat{\sigma}$ is the sample standard deviation following the unbiased estimator:

$$\hat{\sigma} = \left[\frac{1}{N-1} \sum_{i=1}^{N} (x_i - \hat{\mu})^2 \right]^{1/2}. \tag{4.6}$$

With regards to the maximum likelihood estimation, the log-likelihood function is formulated. Taking the partial derivatives of this function for the parameters and equating to zero provides with equations that must simultaneously be solved. The log-likelihood function of the Gumbel distribution is given by:

$$LL = -\sum_{i=1}^{N} \frac{x_i - u}{\alpha} - \sum_{i-1}^{n} exp\left(-\frac{x_i - u}{\alpha}\right) - N\ln(\alpha) \tag{4.7}$$

Taking the partial derivatives and equating them to zero hands:

$$\begin{aligned} N - \sum_{i=1}^{N} \exp\left(-\frac{x_i - u}{\alpha}\right) &= 0 \\ N - \sum_{i=1}^{N} \frac{x_i - u}{\alpha} + \sum_{i=1}^{N} \sum_{i=1}^{N} \exp\left(-\frac{x_i - u}{\alpha}\right) &= 0 \end{aligned} . \tag{4.8}$$

which must simultaneously be solved to arrive at estimates for u and α. For the GEV distribution, only the maximum likelihood method was used for the parameter estimation. For conciseness, the likelihood function of the GEV distribution will not be presented here.

In summary, for each duration three probability distribution estimates are obtained to represent the annual maxima of precipitation data. These estimates are the Gumbel distribution estimated by the method of moments, the Gumbel distribution estimated by the maximum likelihood method and the

GEV distribution estimated by the maximum likelihood method. The Kolmogorov-Smirnov test was used to determine which estimate would be used in the IDF-curve following de Carvalho et al. (2014). The Kolmogorov-Smirnov test is a measure of the maximum vertical distance between the empirical and fitted distribution functions:

$$D = \frac{max}{1 \leq i \leq N} \left(F'(x_i) - \frac{i-1}{N}, \frac{i}{N} - F(x_i) \right) \tag{4.9}$$

with $F'(x_i)$ the empirical cumulative distribution function and $F(x_i)$ the CDF based on the earlier estimates. The final parameter estimates and resulting IDF-curve are treated in Section 6.2.1 (analysis RP01).

It is highlighted that, although the load is determined following a probabilistic approach, the load used is still deterministic. A full probabilistic approach would concern uncertainty around the fit of the distribution. More on the confidence limits for Gumbel/GEV fits may e.g. be found in Maity (2018) and Overeem et al. (2008). This is not applied in this study.

4.1.2. Probability of each duration of rainfall

For further definition of the rainfall spectrum, the probability of each duration of rainfall occurring must be calculated. To this use, the dataset was transformed to states of dry and rainy days. The threshold between these two states was set at a precipitation depth of 2.5 mm in 24 hours. If the rainy days have state '1' and the dry days have state '0', a sequence such as shown in Figure 4.2 may be obtained.

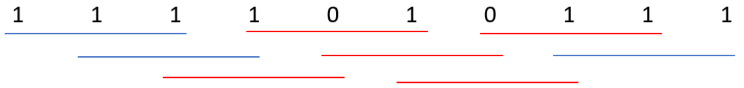


Figure 4.2: An example sequence of dry (state 0) and rainy days (state 1), to determine the probability of occurrence of a certain duration of rainfall.

The blue bars indicate that there have been three days of consecutive rainfall. Whereas, the red bars show that in a block of three days there was at least one dry day. The probability of three consecutive days of rainfall is in this case 3/8. The probabilities for each duration of rainfall, as calculated in this way, can be seen in Table 4.1. Increments of 1 day are used as the data represents daily precipitation depths. No events were registered for which rainfall occurred on 6 consecutive days. In the analysis, no periods are considered for which there is no rainfall on each day. This with respect to some assumptions done later in the geohydrological analysis where evaporation effects are not considered.

Table 4.1
The probability of occurrence of the duration of rainfall for 1 to 6 days.

Length [days]	$P(D = D_i)$ [-]
1	0.3676
2	0.2607
3	0.1930
4	0.1486
5	0.1176
6	0
Total	**1.0875**

The total probability of these events is larger than 1 as these different events are subsets of each other. Represented in a Venn diagram, Figure 4.3 is obtained. Here the larger rectangular box describes all events and hence the area of it is 1. The circle A will in this case be equal to an area of 0.3676 and describes all the events for which it rains.

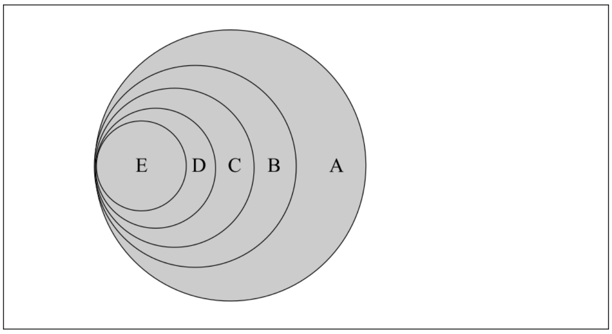


Figure 4.3: Venn diagram of rainfall for various durations of rainfall. The durations are represented by A to E, with A the shortest duration of rainfall and E the longest.

4.1.3. Determine the return period of failure

From the left branch in Figure 4.1, the failure intensity for each duration is obtained. This failure intensity may then be converted to a probability by applying the underlying distributions in the IDF-curve. A conditional probability is obtained that the intensity is larger than the failure intensity given a certain duration: $P(I > I_f | D = D_i)$. Transforming the y-axis of the IDF curve to showing this conditional probability for the various durations and transforming the x-axis to the value of the probability of each duration occurring by using the values of Table 4.1, Figure 4.4 is obtained. The total probability of failure is approximated by the area beneath the dashed lines. This method is later referred to as the 'lower limit'. However, the probabilities are determined on discrete values of rainfall duration and it may be argued that they represent a continuous function. This function is shown as a red curve in the figure. To approximate the area beneath the red curve, a midpoint rule was applied that, for each period, will take the average between the probability of the neighbouring intensities as an estimate for the actual curve. It is noted that for durations longer than 5 days and for durations shorter than 1 day, the curve is cut-off and the probability of failure calculated may still be lower than the actual failure probability.

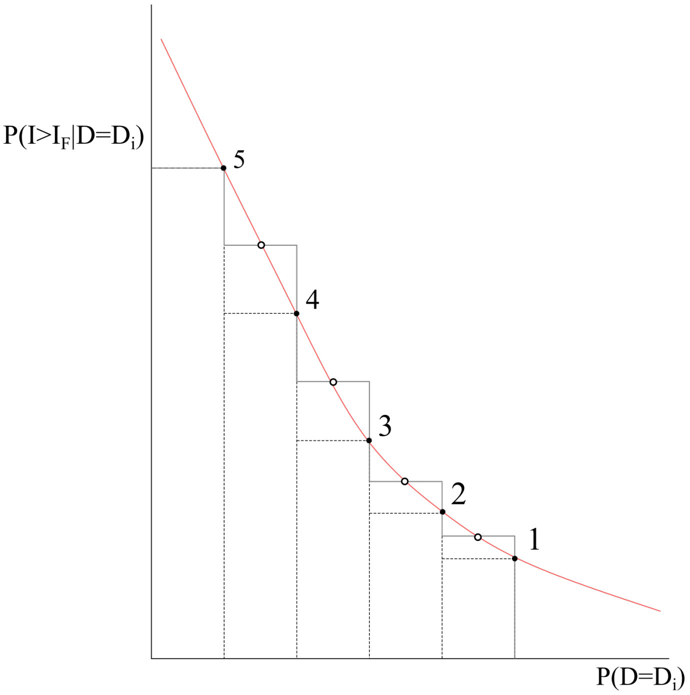


Figure 4.4: Plot for the determination of the total probability of failure. The y-axis holds the conditional probability of exceedance of the failure intensity. The x-axis holds the probability of the duration of rainfall.

The calculated probability of failure is then equal to:

$$
\begin{aligned}
&P_f = \\
&P(I > I_f|D = 5d) \cdot P(D = 5d) \\
&+ \frac{P(I > I_f|D = 4d) + P(I > I_f|D = 5d)}{2} \cdot (P(D = 5d) - p(D = 4d)) \\
&+ \frac{P(I > I_f|D = 4d) + P(I > I_f|D = 3d)}{2} \cdot (P(D = 3d) - P(D = 4d)) \\
&+ \dots
\end{aligned}
\tag{4.10}
$$

The return period of failure is equal to the inverse of the probability of failure as the annual maxima have been used for the distribution fit (Koutsoyiannis et al., 1998).

4.2. Determination of the failure intensity for each duration

In the calculation of the failure intensity, the spatial characterisation, consolidation analysis, geohydrological analysis and liquefaction analysis are needed. Of these the following are treated:

- detailed description of the method;
- assumptions done in the method;
- needed input and desired output.

4.2.1. Spatial characterisation

Detailed description

In the spatial characterisation, the geometry of the impoundment is initialised, presenting the height and the slope of the dam face. No steps or a variation of the slope angle over the dam face are considered. Furthermore, the width of the zones of liquefiable and non-liquefiable tailings are appointed (see Section 3.1 and Figure 3.4). The shell of non-liquefiable tailings is assumed to be of constant width *B*. No foundation is implemented in the analysis and only the impoundment itself will be modelled. The geometry after the spatial characterisation is shown in Figure 4.5.

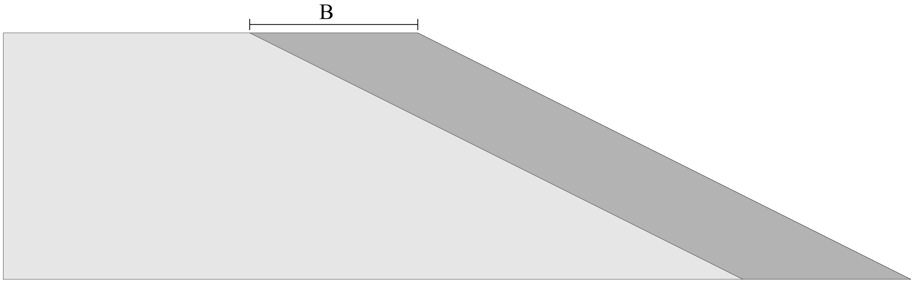


Figure 4.5: The spatial characterisation in phase 3, with liquefiable (light-grey) and non-liquefiable tailings (dark-grey).

Input and output

The following parameters serve as input for the spatial characterisation:

- height of the impoundment;
- width of the non-liquefiable tailings;
- angle of the slope.

Possible values that these parameters may take are treated in Chapter 5. The output is the geometry of the tailings impoundment with the zones of liquefiable and non-liquefiable tailings.

Assumptions
The following assumptions have been made:

- the raised dams in the upstream configuration may be schematised to have constant width;
- 3D effects of flow and strength are negligible.

4.2.2. Consolidation analysis

Detailed description
No consolidation analysis is performed.

Input and output
-

Assumptions
The tailings are assumed to be normally consolidated. No under-consolidation is considered because of the earlier assumption of decommissioned dams as mentioned in Chapter 2. The tailings are also assumed to not be over-consolidated because of continuous deposition that has occurred. This is acknowledged by e.g. Chandler and Tosatti (1995) in which it is stated that the tailings of the Stava tailings dam were normally consolidated to lightly overconsolidated.

4.2.3. Geohydrological analysis

Detailed description
In the geohydrological analysis, the process of rainfall infiltration into the tailings impoundment will be modelled. Traditionally, flow in soils is described through Darcy's law:

$$q = -K(h)\frac{\partial(h+z)}{(\partial z} \tag{4.11}$$

with *q* the water flux [L/T], *K(h)* the hydraulic conductivity of the soil as function of the head [L/T], *h* the soil water pressure head [L] and *z* the vertical position [L] with the positive in the upward direction. Furthermore, the water balance of a soil element should be considered.

$$\frac{\partial\theta}{\partial t} = -\frac{\partial q}{\partial z} - S(h) \tag{4.12}$$

with θ the volumetric water content of the soil [L^3/L^3], *t* a measure of time [T] and *S(h)* any additional source and/or sink terms for the soil element [L/T]. Combination of Equations 4.11 and 4.12 presents the mixed formulation of the Richards equation for soil water flow:

$$\frac{\partial\theta}{\partial t} = \frac{\partial\theta}{\partial h}\frac{\partial h}{\partial t} = \frac{\partial\left[K(h)\left(\frac{\partial h}{\partial z}+1\right)\right]}{\partial z} - S(h). \tag{4.13}$$

The term $\frac{\partial\theta}{\partial h}\frac{\partial h}{\partial t}$ is used for numerical purposes in which the pressure head is solved for over time and the differential moisture capacity $\frac{\partial\theta}{\partial h}$ is applied to relate the water content and the pressure head (Dussaillant et al., 2004). This is done through the use of the soil water characteristics curve (SWCC). This curve is a non-linear function describing the water content inside the soil for various values of the (negative) pressure head. With decreasing pressure head (or increasing suction) the water content in the soil decreases. The SWCC may be described using the Mualem-van Genuchten equation:

$$\theta = \theta_{res} + (\theta_{sat} - \theta_{res})\left(1 + |\alpha h|^{n}\right)^{-m} \tag{4.14}$$

where θ_{sat} is the saturated water content [L^3/L^3], usually equal to the porosity of the soil, θ_{res} is the residual water content [L^3/L^3] and α [1/L], n [-] and m [-] are shape factors from hereon referred to with the subscript ‘mvg’. With regards to the SWCC only a single curve is used and no difference is discerned between the drying and wetting curve. The differential moisture capacity is the derivative of the Mualem-van Genuchten equation for the SWCC and equals:

$$\frac{\partial \theta}{\partial h} = \left| -n_{mvg} m_{mvg} \alpha_{mvg} \frac{(\theta_{sat} - \theta_{res})}{\left(1 + |\alpha_{mvg} h|^{n_{mvg}}\right)^{m_{mvg}+1}} |\alpha_{mvg} h|^{n_{mvg}-1} \right| \tag{4.15}$$

Instead of the water content, the usual state parameter in soil water flow calculations is the relative degree of saturation. The relative degree of saturation describes how much of the pore volume is filled with water on a scale from 0 to 1. It considers the minimum and maximum water content in the soil:

$$S_r = \frac{\theta - \theta_{res}}{\theta_{sat} - \theta_{res}}. \tag{4.16}$$

However, capillary rise should be applied for tailings as noted in Section 3.3. Therefore, the equation needs to be modified near saturation to present a degree of saturation equal to 1 for any head less negative than the air-entry value (Schaap and van Genuchten, 2005; Vogel et al., 2000; Ippisch et al., 2006). This altered equation is equal to:

$$S_r = \begin{cases} \frac{1}{S_c} \left[1 + |\alpha h|^n\right]^{-m} & h < h_e \\ 1 & h \geq h_e \end{cases} \tag{4.17}$$

where S_c is the relative saturation that would normally occur at the air-entry value for the head given by h_e but referred to in this study as the AEV which is more standard in literature (Blight, 2013; Fredlund et al., 2012). The modification of the degree of saturation near saturation also means that a slight change in formula is needed for the Mualem-van Genuchten equation. The saturated water content needs to be presented for any pressure head less negative than the AEV. Likewise, the equation for the differential moisture capacity needs to be adapted as it represents the derivative of the Mualem-van Genuchten equation or the derivative of the SWCC in general. For conciseness of the presented methodology, these functions are not repeated here.

Finally, for equation 4.13, the hydraulic conductivity as function of the head must be described. The unsaturated hydraulic conductivity of a soil is smaller than the saturated hydraulic conductivity due to the reduced pore volume through which water may flow in the unsaturated zone. The unsaturated hydraulic conductivity is expressed by the Mualem equation and is equal to the saturated hydraulic conductivity multiplied with a reduction factor that depends on the degree of saturation:

$$K = \begin{cases} K_{sat} (S_r)^{\lambda} \left[\frac{1-\left(1-(S_r S_c)^{(}1/m)\right)^{m}}{1-\left(1-(S_c)^{1/m}\right)^{m}} \right] & S_r < 1 \\ K_{sat} & S_r = 1 \end{cases} \tag{4.18}$$

with λ as the parameter for the flow path tortuosity, which was set to 0.5 in all analyses.

To limit computational effort in the flow analysis of the two-dimensional impoundment, a ‘quasi-2D approach’ was adopted. In this approach, the impoundment is split in various columns for which the one-dimensional Richards equation, as provided in Equation 4.13, is solved. The results of these columns are then assembled and interpolated onto a 2D grid. This way, a two-dimensional solution is mimicked by performing multiple one-dimensional calculations. The boundary conditions imposed are shown in Figure 4.6 where boundary conditions are defined at the upstream and downstream end of the impoundment. The boundary condition at the downstream end is always equal to zero. Along the slope a seepage/infiltration boundary is implemented.

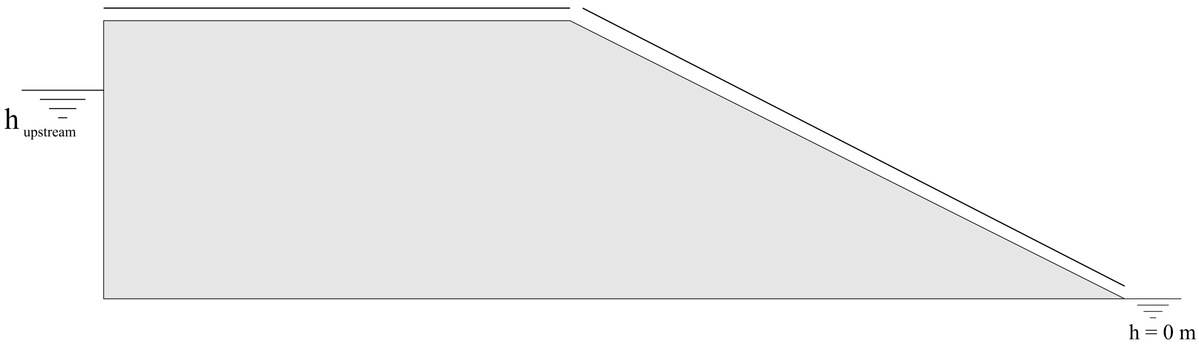


Figure 4.6: The boundary conditions for the 2D flow simulation with a specified boundary head at the upstream end and a head equal to the toe at the downstream end.

The step-by-step procedure for obtaining the two-dimensional results is:

1. **Initial conditions**: The quasi-2D approach is unable to present the initial phreatic surface based on the boundary conditions. A steady-state calculation of the Laplace equation for potentials was performed with the MODFLOW software (Harbaugh, 2005) and FloPy package (Bakker et al., 2016) to present the level of the phreatic surface based on the boundary conditions along the length of the impoundment.

2. The impoundment is split into equally distributed columns. For each column, the height of the slope as well as the initial position of the phreatic surface obtained from the previous step are passed to PySWAP

3. **PySWAP**: PySWAP is a Python-based version of SWAP (Soil, Water, Atmosphere, Plants). SWAP is an open-source finite difference solver of the one-dimensional Richards equation implemented in the language of FORTRAN 90 (Kroes et al., 2017). PySWAP presents the relative degree of saturation and pressure head along the column for various times after the start of rainfall. For the detailed numerical implementation, the reader is referred to Appendix D and Kroes et al. (2017). A few aspects will however be highlighted here.

   First, PySWAP uses a mixed form of the Richards equation for which both the pressure head and the water content are solved simultaneously (Celia and Bouloutas, 1990). A first order implicit time-stepping method is used, which means the solution is obtained in an iterative manner. With regards to the internodal hydraulic conductivity in the finite difference scheme, the weighted arithmic mean of the hydraulic conductivity calculated at the nodes is applied.

   Regarding the boundary conditions, the bottom boundary condition is impervious (zero flow) and the upper boundary condition is an infiltration boundary condition, meaning that it may change between a discharge-based (Neumann) or head-based (Dirichlet) condition depending on the head in the top of the soil column. The maximum allowed level of ponding was set to 0.1 meters. Any additional rainfall will lead to direct run-off.

   The use of this column approach does not allow for horizontal drainage. This provides with conservative results as any recharge to the water table cannot disappear over time. This effect was alleviated by implementing a drainage discharge distributed homogeneously over the part of the column below the phreatic surface. The calculation of the drain discharge follows an altered version of the equations discussed in Kroes et al. (2017). The total discharge is calculated by:

$$q_{drain} = \frac{\phi_{gwl} - \phi_{gwl,ini}}{\gamma_{drain}} \tag{4.19}$$

with ϕ_{gwl} the position of the groundwater table and $\phi_{gwl,ini}$ the position of the groundwater table following the initial conditions. The drainage resistance γ_{drain} is equal to:

$$\gamma_{drain} = \frac{L_{drain}^2}{\beta K_h \phi_{gwl}} \tag{4.20}$$

with L_{drain} the horizontal distance between the column of calculation and the toe of the impoundment, β a fitting parameter and K_h the horizontal hydraulic conductivity. For β a value of 4 was used which yielded slightly conservative results as will be shown in Section 4.3. Use of the PySWAP module for the different columns yields the degree of saturation and pressure heads after the start of rainfall for each column.

4. **Interpolation of results**: The factor of safety will be calculated at some points in time after the start of rainfall. For each of these calculation times, the corresponding values are taken from the results of each column. With these known values at the positions of the calculation nodes, an interpolation is performed using linear barycentric interpolation. The interpolation value is therefore based on the weighted average of the values at the three nearest neighbours. Values are output every 0.1 meters.

The end result is a grid with values of the degree of saturation and the pressure head for each time step to be used in the calculation of the FoS. The grid of the pressure head [m] is transformed to the pore pressure [kPa] by multiplication with the unit weight of water of 10 kN/m^3.

Input and output
The input needed for the geohydrological analysis is:

- boundary condition of the phreatic head at the upstream boundary;
- geometry of the tailings dam;
- geohydrological parameters: Mualem-van Genuchten equation parameters (α_{mvg}, n_{mvg}, AEV) and the saturated hydraulic conductivity (k_{sat});
- porosity n and residual water content θ_{res}.

The output of the geohydrological analysis is a two-dimensional grid of the pore pressure as well as the degree of saturation at various times after the start of rainfall. Values are provided every 0.1 meters.

Assumptions
Below a list of assumptions in this methodology is provided.

- Only rainfall is considered. No evaporation effects will be taken into account.
- No hysteresis of the drying and wetting curves. This simplification is also performed in e.g. Ng and Shi (1998). It will result in an incorrect modelling of the trailing edge of a wetting front.
- The soil skeleton is rigid and there is no coupling between deformation and flow. This assumption is coupled to that of the decommissioned dam. No consolidation is expected anymore.
- The air pressure remains atmospheric (see Chapter 3).
- No horizontal flow is possible in the unsaturated zone (Heinen et al., 2020). A validation of this assumption is provided in section 4.3.

4.2.4. Stability analysis

Detailed description

The undrained strength analysis (USA) is the industry standard approach in the assessment of liquefaction failure mechanisms in tailings dams (Ladd, 1991; Olson, 2001; Olson and Stark, 2003; Brown and Gillani, 2016; Robertson et al., 2019; Morgenstern et al., 2015). With the USA, a fixed undrained shear strength is provided to the liquefiable tailings that is dependant on the in-situ vertical effective stress. The vertical effective stress therefore acts as a preconsolidation pressure. The strength variable corresponding with an USA analysis is given as the strength ratio: S_u/σ'_v. The USA considers the effect of the generation of the excess pore pressures upon shearing which decreases the normal effective stress. The failure envelope is hit at the undrained shear strength of the material. The definition of the strength ratio applied here is also called the peak or yield strength ratio as it corresponds to the largest shear strength the soil holds before failure. The USA differentiates itself from the effective stress analysis (ESA) in which failure is assumed to be fully drained. The normal effective stress is therefore maintained throughout failure. The shear strength failure envelope is hit at a higher shear strength. The strength of a soil in ESA analysis is characterised by the effective friction angle φ'. The different stress paths assumed upon failure for the USA and ESA can be seen in Figure 4.7.

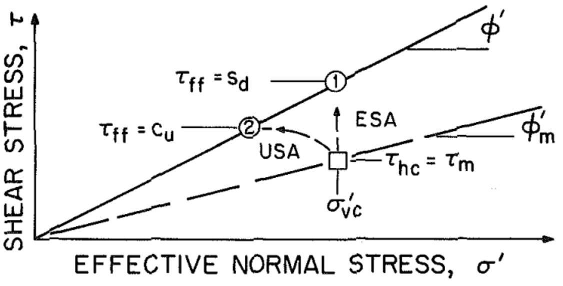


Figure 4.7: The stress paths at failure for the undrained strength analysis and the effective stress analysis for the same preconsolidation pressure before failure. Figure from Ladd (1991).

The USA is applied to the liquefiable tailings with a degree of saturation higher than 90% which was regarded as a boundary between strain-hardening and strain-softening behaviour (see Chapter 3). If the degree of saturation is lower than 90%, the ESA is applied to those tailings. The ESA is also used for the non-liquefiable outer shell as treated in the spatial characterisation. The final strength distribution within the impoundment is shown in Figure 4.8. Such a hybrid strength formulation is also applied in Zhang et al. (2005) and Morgenstern et al. (2016). For conciseness, the friction angle of the non-liquefiable tailings is sometimes referred to as φ'_{dam} and the friction angle of the unsaturated liquefiable tailings is expressed as $\varphi'_{tailings}$. The strength of the foundation is regarded to be far higher then for the impoundment. Failure surfaces through the foundation are therefore not allowed for.

The factor of safety for this strength field is found through the use of the limit equilibrium method (LEM). For the failure surface, the procedure of Cheng (2003) and Rickard and Sitar (2012) is used in which a differential evolution algorithm for a non-circular geometry of the slip surface is applied to find the minimum FoS. The explanation of the differential evolution algorithm will be omitted here and is provided in Appendix D. The key aspects of this algorithm are that it generates a variable slip surface based on 6 degrees of freedom. The entry- and exit point of the slip surface are varied together with the positions of 4 vertices along the failure surface. This creates 5 global sections. Additional slices are placed along these sections to provide with the correct factor of safety. Such slice cuts are also placed at material boundaries.

With regards to the calculation of the factor of safety, the Spencer (1967) formulation of slices has been used which was altered to consider Bishop's equation for effective stress according to Geo-slope international Ltd. (2015). The calculation procedure for the FoS as proposed by Rickard and Sitar

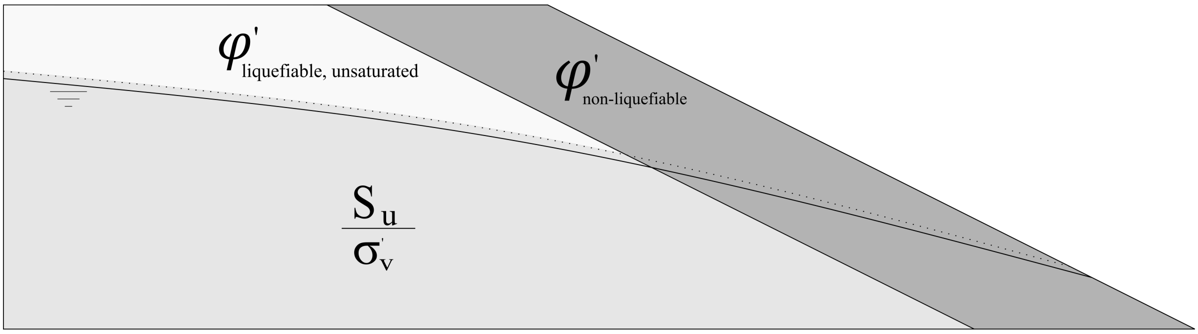

$\varphi'_{\text{liquefiable, unsaturated}}$
$\varphi'_{\text{non-liquefiable}}$
$\frac{S_u}{\sigma'_v}$

Figure 4.8: An overview of the different strengths applied to each zone of the impoundment in the cross-section. The boundary of 90% degree of saturation is indicated by a dashed line.

(2012) was used. The difference between the horizontal interslice forces on each side of the slice is calculated according to:

$$E_r - El = -\frac{(S_u\beta - u_w S_r \tan(\varphi'))\cos(\alpha)}{F} + N\left(\frac{\tan(\varphi')\cos(\alpha)}{F} - \sin(\alpha)\right) \tag{4.21}$$

with E_r the horizontal slice force at the right side and E_l the horizontal slice force at the left side, S_u the undrained shear strength for the liquefiable tailings [kN/m^2], β is the length of the base of each slice [m], u_w is the (negative) pore water pressure [kPa], S_r the relative degree of saturation[-], φ' the friction angle for the unsaturated liquefiable tailings or the non-liquefiable tailings [rad], α is the slope angle of the base of the slice [rad] and F the factor of safety [-]. Regarding the geohydrological parameters, the degree of saturation (S_r) and the (negative) pore pressure (u_w) result from the interpolated grids of the geohydrological analysis. For the strength, each node was either assigned a friction angle (φ') or an undrained shear strength (S_u) according to Figure 4.8. The vertical effective stress, used in the calculation of the undrained shear strength, was based on the total stress according to the unit weight subtracted with the pore pressure at that location.

The vertical interslice forces are related to the horizontal interslice forces by the force inclination parameter λ in the Spencer formulation. λ is constant for all slices:

$$X_r - X_l = \lambda\,(E_r - E_l)\,. \tag{4.22}$$

From this, the normal force at the base of each slice can be calculated according to:

$$N = \frac{\sum W - (X_r - X_l)\frac{S_u\beta\sin(\alpha)}{F} + u_w\frac{\beta\sin(\alpha)}{F}S_r\tan(\varphi')}{\sum m_\alpha} \tag{4.23}$$

with W the weight of each slice [kN] and $m_\alpha = \cos(\alpha) + \frac{\sin(\alpha)\tan(\varphi')}{F}$. This will allow for the calculation of the factor of safety based on the global equilibrium for the moment and vertical force.

$$F_m = \frac{\sum S_u\beta R + (N - u_w\beta S_r R\tan(\varphi')}{\sum Wx - Nf} \tag{4.24}$$

$$F_f = \frac{\sum S_u\beta\cos(\alpha) + (N - u_w\beta S_r)\tan(\varphi')\cos(\alpha)}{\sum N\sin(\alpha)} \tag{4.25}$$

with *x, R* and *f* being additional geometrical parameters of each of the slices for calculation of equilibrium (see also Fredlund and Krahn (1977)).

The factor of safety is implicitly included in these equations. Furthermore, the value of λ is initially unknown. For both, an iterative procedure is needed to solve these equations. Within each equilibrium calculation (Equations 4.24 and 4.25), an iteration is performed for the factor of safety to converge to a certain value. Additionally, a Newton-Raphson scheme is used to find the value of λ for which the converged force and moment factor of safety (F_f and F_m) are equal. That factor of safety is regarded as the FoS following the Spencer formulation.

Input and output
The following input is needed:

- Spatial characterisation
- Grids of the degree of saturation and the pore pressure from the geohydrological analysis
- Strength parameters: S_u/σ'_v for the liquefiable tailings, φ' for the unsaturated liquefiable tailings and φ' for the non-liquefiable tailings.

The output is the lowest factor of safety in the days after the start of rainfall. This factor of safety is checked against the failure factor of safety. Following the method of Olson (2001), The failure FoS is set at a value of 1.1. This value considers to some degree any potential difference of the factor of safety along the slip surface and avoids that the tailings have already liquefied before the failure factor of safety is reached.

Assumptions
The following assumptions are made with the use of the limit equilibrium method in cooperation with the USA and ESA:

- The undrained strength ratio correctly resembles the influence of grain size, void ratio and rate of loading on the finally obtained shear strength.
- No strain dependant behaviour: peak undrained shear strength and friction angle are mobilised simultaneously.
- The factor of safety is homogeneous along the failure plane.

4.3. Validation by parts

The main body of the proposed methodology consists of the combined geohydrological and stability analysis. In this section, the proposed methodology for these analyses will individually be validated using commercial software. Section 4.3.1 treats the validation of the geohydrological analysis. Section 4.3.2 will show the results of the validation of the stability analysis.

4.3.1. Validation of the geohydrological analysis

The validation of the geohydrological analysis has been performed using the programme Plaxis. As Plaxis is not capable of applying the air entry value, the validation of that has been separately performed. Both validations are now treated in the following sections.

Validation of the geohydrological analysis in Plaxis
In the geohydrological analysis it is assumed that an accurate flow field can be described by the quasi-2D approach. To test this assumption, a validation calculation was performed in Plaxis. The transient flow module with flow only was used in Plaxis, assuming a rigid soil body. The parameters that were used both in the Python code and in Plaxis can be seen in Table 4.2. The boundary condition at the upstream side was modelled as closed and a permeable foundation was modelled beneath the dam to

which the boundary condition would be imposed. This avoided interference of the boundary condition with the wetting front. The imposed boundary conditions are visualised in Figure 4.9. The analysis was continued for 15 days after the end of rainfall to show further passing of the wetting front. The results of the Plaxis calculation can be seen in Figure 4.10. In Figure 4.11, the results of the geohydrological analysis in PySWAP are given.

Table 4.2
Parameters used in Plaxis in the verification of the geohydrological analysis.

Height	60	[m]
Boundary condition head	40	[m]
Rainfall intensity	0.4	[m/day]
Rainfall duration	5	[days]
Maximum ponding	0.2	[m]
k_{sat}	0.864	[m/day]
gm_{α}	2.0	[1/m]
gm_{n}	2.0	[-]
S_{sat}	1.0	[-]
S_{res}	0.001	[-]

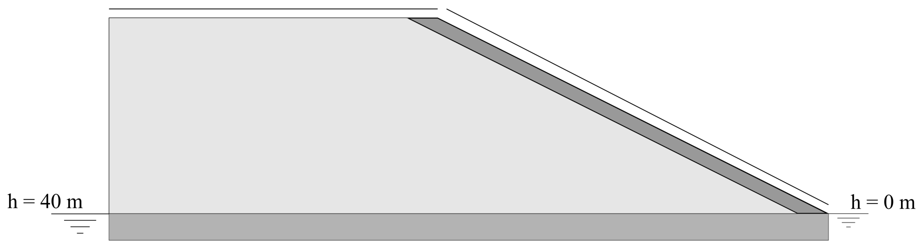
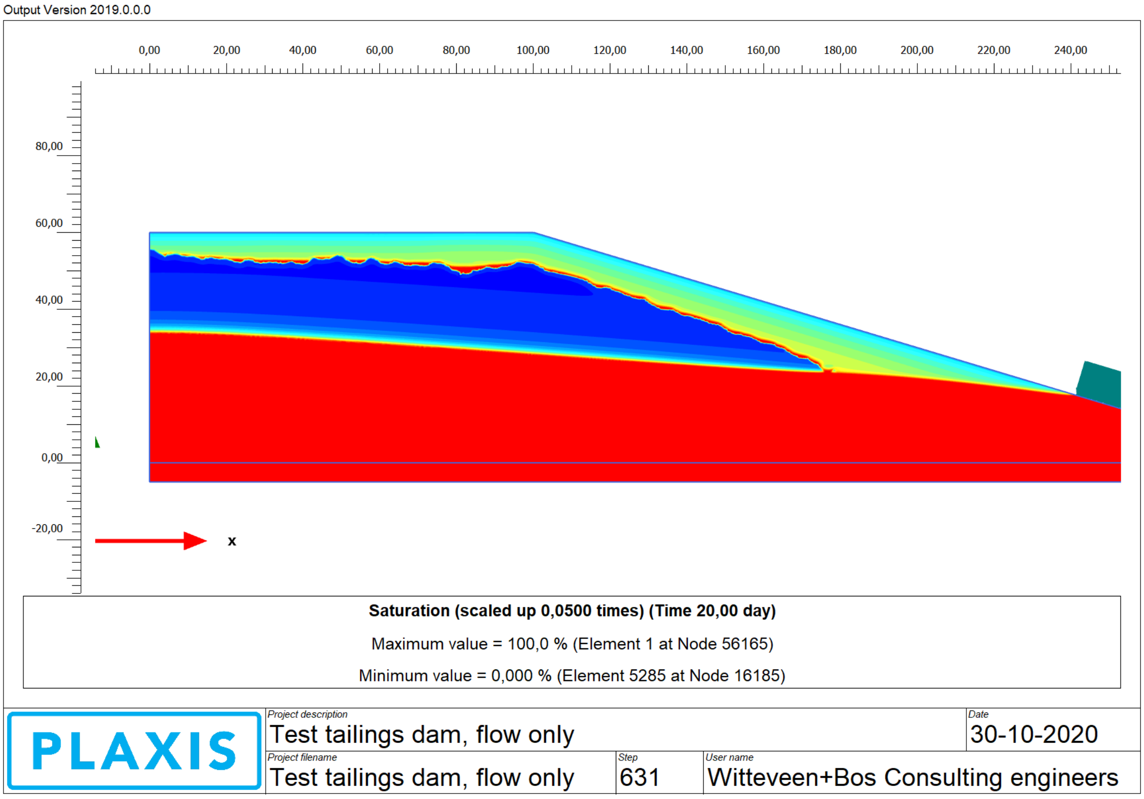


Figure 4.9: The imposed boundary conditions for the flow model in Plaxis. The impoundment holds a closed boundary at the upstream side (left) whereas the head is imposed in the foundation. The other side has a head equal to the toe. The slope has an infiltration condition. The dark grey zone resembles the non-liquefiable shell as was used in the validation of the stability analysis.

Shown in the Plaxis results is a mesh dependency of the wetting front profile caused by the infiltration boundary condition on top. The switch between a discharge (Neumann) and head-based (Dirichlet) boundary condition is not occurring at the same instance along the entire boundary. A smaller mesh size is needed to show a sharp and regular wetting front. The wetting front has reached around 10 meters depth in PySWAP. In Plaxis it is arbitrary to conclude on a single position of the wetting front, but the wetting front has infiltrated less deep. Difference may lie in the calculation of the internodal permeabilities and the different heads at surrounding cells.

Plaxis shows that the quasi-2D approach, as implemented in Python, is a proper approximation and that horizontal flow in the unsaturated part is of little importance. The wetting front hits the phreatic surface after 20 days at an x-position of 178 meters for both analyses. Plaxis tends to better show the effect of the increasing wetting front speed in the capillary zone. In Python this effect is less visible because of the interpolation performed. The drainage package allows for a correct horizontal drainage. At earlier solution times, it was shown that the PySWAP drainage module tends to underestimate the drainage capacity. This is likely caused by the inaccurate value of the parameter β in equation 4.20 which was set to 4. This value is however maintained to obtain conservative results.

It is concluded that the geohydrological analysis performs well and is capable of mimicking important features in 2D unsaturated flow. Additional research could be implemented in calibration of the drainage package. The efficiency of the quasi-2D approach is large, cutting computational time by 95% with respect to the Plaxis model.



Saturation (scaled up 0,0500 times) (Time 20,00 day)

Maximum value = 100,0 % (Element 1 at Node 56165)

Minimum value = 0,000 % (Element 5285 at Node 16185)

Project description: Test tailings dam, flow only

Date: 30-10-2020

Project filename: Test tailings dam, flow only

Step: 631

User name: Witteveen+Bos Consulting engineers

Figure 4.10: The relative degree of saturation in Plaxis for the comparison analysis.

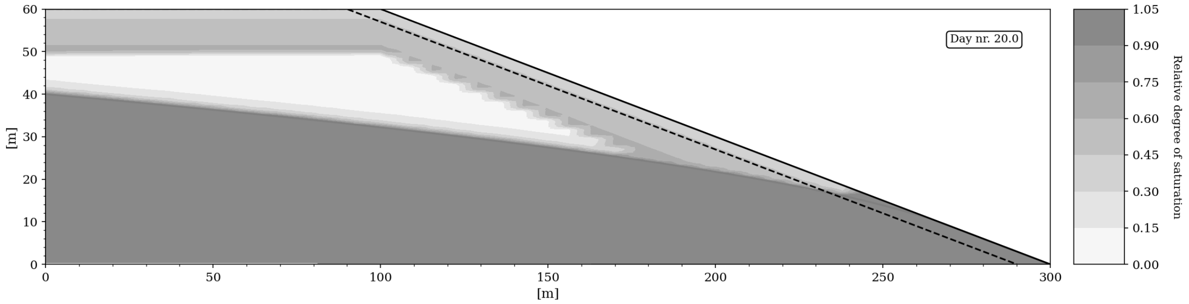


Figure 4.11: The relative degree of saturation in the comparison analysis using PySWAP and corresponding geohydrological analysis.

Validation of the use of the air-entry value

As an additional effect, the air-entry value of the SWCC was modelled through the use of Equation 4.17. The behaviour of the AEV implementation could not be checked in Plaxis. The behaviour was instead compared to the little literature available. In this comparison, some deviance was observed.

The behaviour of the AEV was checked by performing a rainfall infiltration analysis for three different air entry values: 0, 0.1 and 0.5 meter. The wetting front shape was then compared for three different intensities: 0.025, 0.1 and 0.2 m/day for a rainfall period of three days. The saturated hydraulic conductivity was in this case 0.1 m/day. The various analyses presented would thus compare rainfall intensities which are respectively: lower than the saturated hydraulic conductivity, equal to it and larger than the hydraulic conductivity. The results can be observed in Figure 4.12.

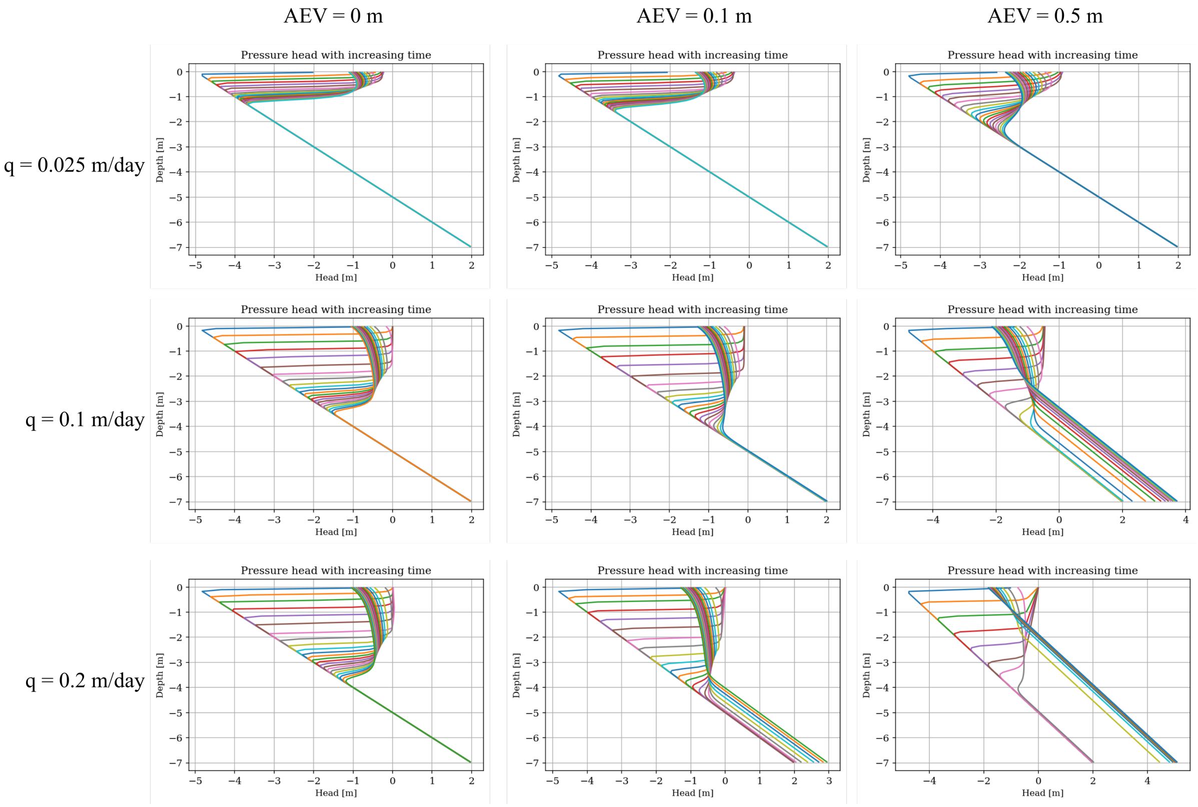


Figure 4.12: The pressure head along the vertical column over time expressed in terms of pressure head for different combinations of intensity and AEV.

Shown is that, with increasing AEV, the wetting front speed increases as well. Also, the shape of the wetting front is less sharp. With increasing intensity, the wetting front speed is slightly increased. Less difference is however visible between the intensities of 0.1 and 0.2 m/day as any additional intensity above the value of the saturated hydraulic conductivity results in direct ponding and possibly run-off. Of significance is that the bound to which the negative pressure head is reduced in the unsaturated zone shifts with increasing AEV. For an AEV equal to 0, the pressure head can be reduced to zero as well. For the AEV larger than zero, the pressure head is reduced to a value equal to that of the AEV. With an intensity of 0.2 m/day, ponding needs to occur and the suction will reduce to zero but shows a linear behaviour between 0 and the depth at which the suction equals the AEV again.

This is some remarkable behaviour that is not shown in literature. Although little comparison examples could be found, Zhang et al. (2016b) provides a similar sensitivity study of the AEV, where the suction always reduces to zero independently of the AEV. Schaap and van Genuchten (2005) do provide wetting front shapes for different AEV, but only a very small value is used. So, this behaviour cannot be discerned. It is unsure whether the observation is realistic or that it is a modeling artefact. The latter is argued to be the case as it was not observed in literature and it follows some of the presented equations. In the original publication of this modification near saturation by Vogel et al. (2000), only a small AEV of 1 to 2 cm is used. It is therefore assumed that the applied approach is generally only applicable to small values of the AEV. Sensitivity analyses with respect to the air-entry value should therefore be performed with caution.

4.3.2. Validation of the stability analysis

Validation of the stability analysis was performed in two different approaches:

- validation through the use of Plaxis to compare the results of LEM and FEM;
- validation by D-Geo stability as a check of the LEM formulation.

Stability analysis validation in Plaxis

The user defined model Shansep NGI-ADP is the constitutive model which resembles most closely the strength ratios implemented in this method (Panagoulias and Brinkgreve, 2017). However, some differences are:

- The use of finite element strength reduction method instead of the limit equilibrium method;
- Plaxis applies a strength ratio of S_u/σ_1' rather than S_u/σ_v', this might cause deviations between the code and the Plaxis results.

It is important to stress that the study has been done to show if the Python slope stability model is correct. It is not intended to show the difference between the various static liquefaction models. The Shansep NGI-ADP model was chosen here as it allows for replication of the Python model.

An impoundment of 60 meters high and 300 meters width was used again with the boundary head at 40 meters. The shell width was equal to 10 meters. The material parameters of the tailings and of the shell can be seen in Table 4.3. For the tailings, the Shansep material parameters are shown as the NGI-ADP model was not used in the analysis. Those parameters (except for one stiffness parameter) did not show any effect. The shell material corresponds to the non-liquefiable shell of 10 meters wide. This is visualised in Figure 4.9

Parameter	Tailings	Shell
Model	Shansep NGI-ADP	Mohr-Coulomb
Drainage type	-	Drained
E' [kN/m^2]	-	10000
ν [-]	-	0.3
φ' [°]	-	25
c [kN/m^2]	-	0
ψ [°]	-	0
γ	22	22
k$_x$ [m/day]	0.864	0.864
k$_y$ [m/day]	0.864	0.864
G/S$_u$ [-]	200	-
α [-]	0.59	-
m [-]	1.0	-
S$_{u,min}$ [kN/m^2]	0	-
OCR$_{min}$ [-]	1.0	-

Table 4.3
The applied strength and stiffness parameters in Plaxis.

A simple steady-state flow analysis was used to calculate the position of the phreatic surface. A rolling boundary is imposed at the upstream side of the impoundment to allow for vertical deformation. In the analysis, two phases were employed:

- Initial phase: gravity loading, simple steady-state flow model for the boundary conditions;

- Phase 1: safety calculation with strength reduction to compare to LEM. Suction effects were considered.

Important in the analysis, and in the difference between FEM and LEM is the rotation of the principle axes due to the slope and the value of the lateral earth pressure coefficient at rest: K_0. In the initial phase, a coefficient of 0.44 was found which is approximately the same as also observed in Robertson et al. (2019), where a value of 0.45 is mentioned. Having a K_0 near 0.5 shows that there is only a minimal rotation of the principal axes. This is in favour of the application of LEM, as this method is not able to capture the effects of such rotation on the material strength.

Plaxis calculates Shansep strength based on the major principal stress as a measure of the confining effective stress rather than the vertical effective stress. Robertson et al. (2019) applies a different ratio which is based on the mean principal effective stress *p'*. No methods of parameter translation between these different ratios have been found in literature. That meant, a back-analysis had to be performed in which:

1. A random strength was considered in Plaxis;

2. The strength ratio based on the vertical effective stress was back-calculated from the Plaxis results and a factor was calculated as to which this parameter differed from the test in Python;

3. The found FoS was divided by the factor as mentioned in the previous step.

A strength ratio of 0.35 was applied in Python with respect to the vertical effective stress. Applying an arbitrary value of 0.59 in Plaxis would result in a value of 0.55 for the same definition of the strength ratio. The corresponding FoS in Plaxis was 1.89. Scaling this FoS with the same factor as the ratio of the found strength ratios gives: $0.35/0.55 \cdot 1.89 = 1.20$. Exactly the same FoS was obtained in Python, showing that for these models of tailings dams there is little difference between the application of FEM and LEM methods.

As may be expected, the analysis was performed again in Plaxis, but the 'guess strength' was reduced with the same factor of $0.35/0.55$. This however caused the solution to be unstable and the material failed upon gravity loading. The fact that the material failed could be attributed to the non-linearity of the calculated factor of safety to the material strength.

4.3.3. Stability analysis validation with D-Geo Stability

D-Geo Stability is a limit equilibrium code for stability analysis and was applied as a check for the implemented LEM method. The geometry applied was the same as for the Plaxis calculations. A slope of 60 meters high with a total length of 300 meters. The slope was set to a value of 3H:1V.

To mimic the dual-strength approach as explained in chapter 4, two sets of materials were defined. The first material was characterised using the drained c-phi approach, the second by using the peak strength ratios. The peak strength ratio is in this case defined by the vertical effective stress. D-geo stability does not perform any geohydrological calculations. Therefore it was not possible to define the material boundary at the 90% degree of saturation line. Instead, it was chosen to have a material boundary at the phreatic surface. As there is no flow module within D-Geo Stability, the phreatic surface was mimicked by applying a polyline. The input geometry is shown in Figure 4.13. The applied parameter values can be seen in Table 4.4. The shell width was both omitted in Python and D-Geo stability. The Spencer LE method was used for a total of 5 slices. The nodes of the slices were fixed to the governing slip surface found through the differential evolution algorithm in Python. Therefore, D-geo stability was only used to check if the equations were implemented correctly. Suction was not implemented in D-Geo stability as this did not fully work for the approach used. It was however applied in Python.

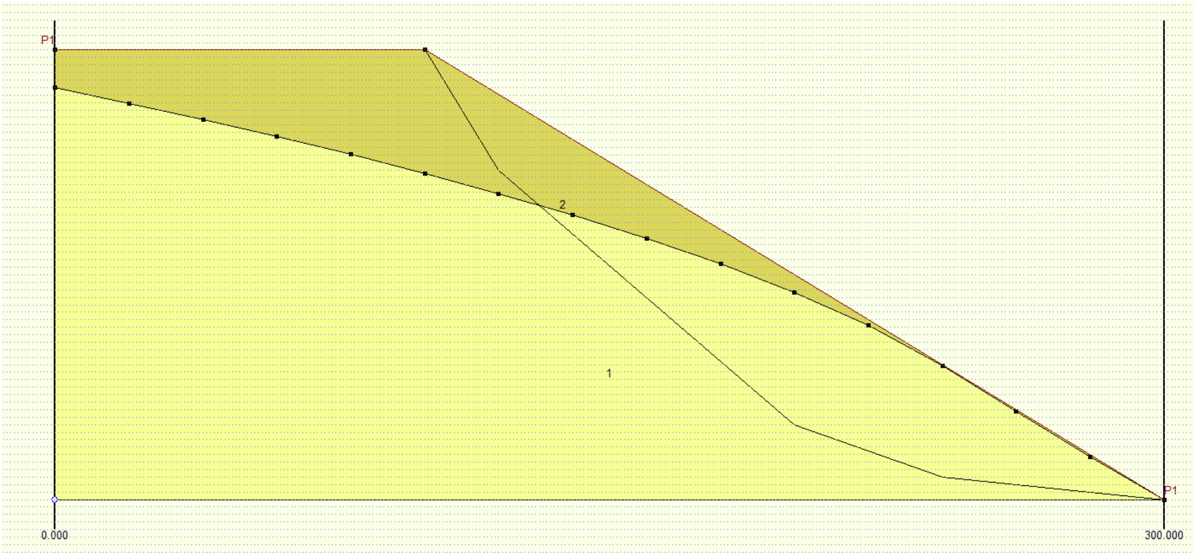


Figure 4.13: The input in the D-Geo stability programme, with the two different materials along the phreatic surface, hence the parabolic material boundary. The governing slip surface is also shown.

	Below PS	Above PS
Unit weight [kN/m^3]	23	23
φ' [°]	-	22
c [kN/m^2]	-	-
S_u/σ'_v	0.4	-

Table 4.4
Input parameters for the D-Geo stability validation calculation. The abbreviation PS refers to the phreatic surface.

Both D-Geo Stability and the Python code showed a factor of safety of 1.04 for this configuration. The Python slope stability module was therefore approved. However, different factors of safety were expected as Python models suction, whereas D-Geo Stability does not. Upon further analysis it was found that this had to do with the fact that only 5 slices were used. The midpoint of the top slice was situated in a point with only a very small degree of saturation. Following Bishop's formulation of effective stress, the suction effect was low and it did not prove to be of vital importance to the factor of safety.

CHAPTER 5

Parameter ranges for sensitivity analysis

Proceeding with the modelling strategy as proposed in chapter 4, parameter ranges are needed for all of the individual parameters in further analysis. As there is a considerable lack of data, a literature study was performed for providing these parameter ranges. Section 5.1 will treat the found values/ranges for the geometrical parameters: height, slope and width of the non-liquefiable tailings shell. Then section 5.2 will continue with the needed strength parameters: peak strength ratio, drained friction angle for unsaturated liquefiable tailings, drained friction angle for non-liquefiable tailings and the unit weight. Section 5.3 will analyse the possible values of the geohydrological parameters: porosity/saturated water content, residual water content, saturated hydraulic conductivity, anisotropy of permeabilities and the Mualem van Genuchten parameters: α_{mvg}, n_{mvg} and AEV. Section 5.4 will provide with the numerical parameters that were applied in the analysis. Although no sensitivity analysis will be performed on these parameters as an integral part of the study, they are included as to provide with all necessary parameters to potentially reproduce the study.

5.1. Geometrical parameters

Considering the geometrical parameters, related to the shape of the tailings dam, three parameters were discussed earlier: the height of the tailings dam, the average slope angle and the width of the non-liquefiable shell. With respect to the height of the dam, a proper overview is provided in Vale (2020) which presents the tailings dams owned by one of the larger Brazilian mining companies. It provides with the description of 21 upstream tailings dams of which their heights are summarised in the histogram in Figure 5.1.

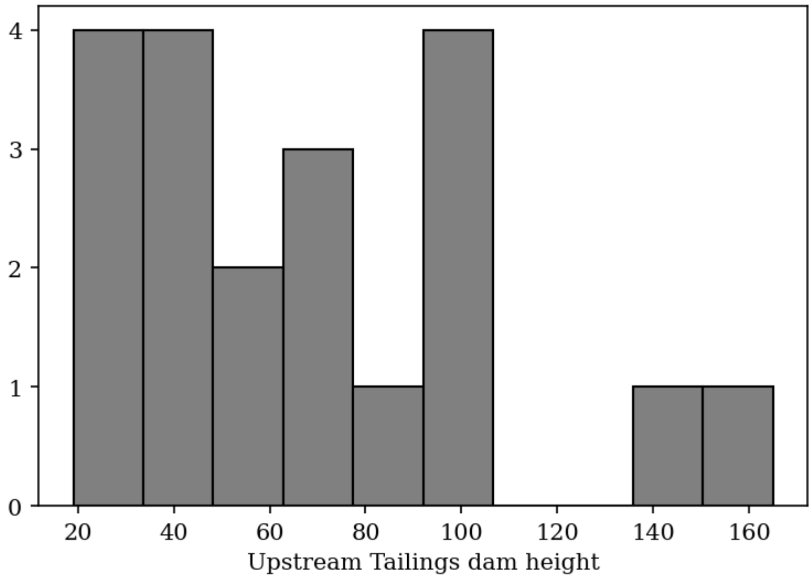


Figure 5.1: A histogram of the height of the upstream tailings dams recorded in Vale (2020).

Although data is scarce, it provides with some idea of the distribution of the heights of upstream tailings dams. Based on this data, it was decided to describe three heights for usage in the sensitivity analysis: 20, 60 and 100 meters. These represent the three ‘peaks’ in the histogram and are considered as low, medium and high tailings dams. The extremely high tailings dams of around 140 meters were neglected in the analysis.

The slope angle of a tailings dam is variable and often it is unsure whether local or global inclination angles are provided in documentation and literature. Robertson et al. (2017) provides with a slope angle of 4H:1V, which equates to 14.0 °. Back-calculated from some figures in Robertson et al. (2019) is an inclination of 3H:1V (18.4 °). Morgenstern et al. (2016) mention 3.2H:1V overall slope angle. It is argued that the slope angle may differ between 2.5H:1V and 4.5H:1V in order to cover most of the tailings dams geometries. There is no correlation assumed between the tailings dam height and its slope angle.

For the last parameter, it is unclear what the typical shell width of tailings dams is as it is a clear modelling parameter rather than a physical parameter. Judging from the expert panel report of Robertson et al. (2019), the shell width is roughly 30 meters, but it is unsure if the tailings deposited immediately near the dam fail in an undrained matter as assumed in section 4.2.1 for the modelling of the spatial characterisation. Vick (1992) reports larger values for the shell width and notes that small shells such as applied at the Feijão dam represents poor practice. In reality, a more gentle effect such as proposed in the master curve of a tailings dam by Blight (1994) is likely the case. Contemplating on the matter it has been decided to not set a range on the width of this shell. It may in that sense take any value. There will also not be any correlation assumed between the height of the dam and the width of the shell.

5.2. Strength parameters

For the strength parameters, there are four parameters which need to be defined: the strength ratio, the drained friction angle for unsaturated liquefiable tailings, the drained friction angle for the dilative shell and the unit weight. The latter is not necessarily a strength parameter, but it contributes both to loading and strength as the undrained shear strength is based on the vertical effective stress through the strength ratio.

Peak strength ratio of tailings

Table 5.1 provides with the found values/ranges of the peak strength ratio of the tailings in literature. Additional notes are provided on the type of tailings and the methods of testing to derive the parameter values if those were provided in the source.

Table 5.1
Values for the peak strength ratios of tailings reported in literature.

Peak strength ratio, S_u/σ'_v [-]	Notes	Source
0.26-0.37	Mostly 0.37 was used	Robertson et al. (2019)
0.25-0.34	but as low as 0.22 used in a heterogeneous analysis	Morgenstern et al. (2016)
0.25-0.3	copper tailings	Ladd (1991)
0.27	estimated from CU test	Hu et al. (2017)
0.25-0.30		Morgenstern et al. (2019)

The full range of the peak/yield strength ratio provided in literature is between 0.25 and 0.37. This entire range will be considered in the analysis, however mostly a value of 0.37 will be used as this value was used in the Feijão dam analysis and was argued to be most reliable.

Drained friction angle liquefiable tailings

The drained friction angle for the liquefiable tailings is needed for computation of the strength of the unsaturated part liquefiable tailings. Table 5.2 holds the found values and their sources.

Table 5.2

Values for the drained friction angles of the unsaturated liquefiable tailings reported in literature.

Friction angle [°]	Notes	Source
22-25		Abadjiev (1976) as cited by Rassam (2002)
35-42.5		Pettibone and Kealy (1971) as cited by Rassam (2002)
33		Morgenstern et al. (2016)
34	critical state friction angle	Robertson et al. (2019)
26-34	copper tailings	Ladd (1991)
32-41		Hu et al. (2017)
34 - 40		Morgenstern et al. (2019)

The absolute range of the drained friction angle of liquefiable tailings is between 22 and 42.5 degrees. The 22 degrees that is reported is low compared to the other studies. It may be noted that the other friction angles mentioned in literature are high. Rassam (2002) provides with the observations in Pettibone and Kealy (1971) which state that due to the needle shaped particles in silt-like tailings these hold a higher strength than sand tailings which have more rounded particles. For this friction angle, a range will be adopted between 30 and 40 degrees. The higher values are cut-off slightly also based on the reported values of the two Brazilian dams which are considered to be more accurate for the scope of this thesis.

Drained friction angle non-liquefiable tailings

Also with respect to the drained friction angle of non-liquefiable tailings, various values have been extracted from literature. Table 5.3 provides an overview of the values noted, as well as their source.

Table 5.3

Values for the drained friction angle for the non-liquefiable tailings reported in literature.

Friction angle [°]	Notes	Source
30-35		Abadjiev (1976) as cited by Rassam (2002)
35		Morgenstern et al. (2016)
30		Morgenstern et al. (2015)
35	copper tailings	Ladd (1991)
35	sand tailings	Chandler and Tosatti (1995)

For the drained friction angle, less values are reported. As some of the earlier reported dams hold clay cores or have a rockfill for additional support and do not rely on a sand shell. What strikes is that the reported values of the sand tailings are not as high as the values of the silt-like, liquefiable tailings. This was also addressed with the previous parameter and the fact that the silt tailings hold more needle shaped particles for additional strength (Rassam, 2002). It may well be that indeed the drained friction angle of the sand tailings in the berm is lower. However, as they will always perform in a drained manner, their strength in an undrained analysis may be higher. The range of 30-35 as reported in literature is used in the sensitivity analysis. The chosen parameter ranges for both friction angles contradicts the master profile as discussed in Chapter 3 from which it would be expected that the shell near the dam face holds the highest strength. However, given the amount of sources that provide with a lower friction angle for the shell and a higher one for the tailings, these ranges are accepted.

Unit weight

In table 5.4 the various found values for the tailings unit weight and their sources are given. It is assumed that the tailings above and below the phreatic surface hold the same value for the unit weight.

Table 5.4
Values for the unit weight of tailings reported in literature.

Unit weight $[kN/m^3]$	Notes	Source
22		Morgenstern et al. (2016)
19.1	copper tailings	Ladd (1991)
26		Robertson et al. (2019)
22.6-24.4	(estimated from specific gravity)	Hu et al. (2017)
19.0		Chandler and Tosatti (1995)
20		Morgenstern et al. (2019)

There is some large variation in the unit weight. Iron ore tailings seem to contain large amounts of iron minerals after processing and the tailings are therefore heavily metallic. This results in a higher unit weight. The values as reported for the tailings dams in Brazil are found to be more relevant in this study. A value of 23 kN/m^3 is used throughout all analyses.

5.3. Geohydrological parameters

Here the geohydrological parameters will be addressed. The needed parameters for the analysis are:

- boundary head at the upstream side of the impoundment
- porosity or saturated water content;
- residual water content;
- saturated (vertical) hydraulic conductivity;
- anisotropy (k_h/k_v);
- van Genuchten parameter n;
- van Genuchten parameter α;
- Air-entry value (AEV).

Special use is made of Zhang et al. (2016b) as a source for the different parameters. From this source, the values of 'silt loam' are taken as representative values for tailings, based on the saturated hydraulic conductivity. Provided for the silt loam are a mean and standard deviation of every parameter. In order to get to a range for a sensitivity analysis for this study, an offset of 1.96 times the standard deviation away from the mean on both sides is used.

It must be noted that data on the van Genuchten fit for the SWCC of tailings is scarce. Also it is not always reported whether the fitted parameters belong to the wetting or the drying curve of the SWCC. The uncertainty will be reflected in the considered range in the sensitivity analysis.

Upstream boundary head

The quasi-2D approach applies two boundary conditions for the formulation of the initial phreatic surface before the start of rainfall. These two boundary conditions are the head at the toe, which always equals zero, and the head at the upstream side of the impoundment. Normally, the upstream boundary head is given by the level and position of the tailings pond. However, in the decommissioned phase, this tailings pond might no longer exist and the boundary condition is a result of a complex interaction mechanism between the tailings impoundment and its surroundings (see Chapter 2). In this study, the boundary condition may therefore take any value along the height of the impoundment. However, it will not be set to rise above it. It is recommended to perform additional research on the occurring boundary conditions in impoundments for increasing the accuracy of simulations as performed in this study.

Porosity

Table 5.5 holds the values of the porosity of tailings to be used in the analysis. Reporting on single values of the porosity is in principal not correct as this parameter is dependant on the or confining pressure. In analysis the porosity of the tailings was not varied and its influence on the return period of failure was not studied. The chosen value for the porosity was set at 0.4.

Table 5.5
Values for the porosity of tailings reported in literature.

Porosity [-]	Notes	source
0.39		Rassam (2002)
0.5		Robertson et al. (2019)
0.41-0.47	At 100 kPa	Hu et al. (2017)
0.29 - 0.61		Zhang et al. (2016b)
0.35		Booshehrian et al. (2020)
0.402-0.501		Zhang et al. (2016a)

Saturated hydraulic conductivity

Normally, the tailings dam will be divided in different zones in terms of hydraulic conductivity modelling. Usually this means that the sand tailings have a larger hydraulic conductivity due to the larger grain sizes and associated larger pores. Although a wide variety of permeabilities is mentioned in literature, a single range has to be chosen for the use in the sensitivity analysis because of the quasi-2D approach. In this way, the hydraulic conductivity of both sand and slimes tailings will be applied here and no difference between them will be made. The found values can be seen in Table 5.6.

Table 5.6
Values for the saturated hydraulic conductivity reported in literature.

Saturated hydraulic conductivity [m/s]	Notes	Source
3e-5 - 1e-7		Rassam (2002)
1e-6 - 2e-8	Coarse to fine	Robertson et al. (2019)
3e-6	Sands	Morgenstern et al. (2016)
1e-7 - 1e-5		Zhang et al. (2016b)
1e-6		Zandarín et al. (2009)
1e-7	silt tailings	Chandler and Tosatti (1995)

The found permeabilities show a high variability. All values found in reports were included, not separating between different void ratios or different material characterisations. It was chosen to adopt 1e-6 m/s as a standard value and consider a range in between 5e-6 and 5e-7 m/s.

Residual water content

The residual water content is needed in the geohydrological analysis as to model the minimal water available in a fully desaturated soil. Although not many values of this parameter could be found, it is not believed that it will influence the results to a large extent. The values from literature are shown in Table 5.7.

Table 5.7
Values for the residual water content of tailings reported in literature.

Residual water content [-]	source
0.039 - 0.095	Zhang et al. (2016b)
0.05	Booshehrian et al. (2020)
0.067-0.09	Zhang et al. (2016a)

Considering the minor effect that it will likely have on the results, a standard value of 0.05 is adopted and the residual water content will not be considered in the sensitivity analysis.

Van Genuchten parameter n_{mvg}
The parameter n_{mvg} is related to the steepness of the SWCC. The found values are shown in table 5.8.

Table 5.8
Values for the Mualem-van Genuchten parameter n_{mvg} for description of the SWCC reported in literature.

n_{mvg}	Notes	Source
1.17 - 1.65		Zhang et al. (2016b)
1.61		Zandarín et al. (2009)
2.0		Booshehrian et al. (2020)
1-1.7	Stava silty tailings wetting curves	Bella (2017)
1.68-2.07	drying curve	Zhang et al. (2016a)

A range of 1.1 to 2.1 is adopted in analysis. The value of 1 is not included as this will not be numerically possible. Hysteresis and potential differences between the drying and wetting curve are not considered.

Values for the Mualem-van Genuchten parameter α_{mvg}
Also for the parameter of α_{mvg}, values were found.

Table 5.9
Values for the Mualem-van Genuchten parameter α reported in literature.

α [1/m]	Notes	Source
0.1-4.5	silt loam	Zhang et al. (2016b)
1.5	originally in 1/kPa	Booshehrian et al. (2020)
3.4	Stava silt tailings	Bella (2017)
0.14-0.37		Aubertin et al. (1998)

In analysis, values of α_{mvg} below 1 resulted in poor convergence of the solution. The chosen range is therefore from 1.0 to 4.0 indicating some upper bound as well.

Air entry value
Additional to the van Genuchten Mualem function is a slight modification near saturation in which the actual capillary rise is considered. The soil will remain fully saturated until reaching a pressure head equal to the air entry value. Literature findings are shown in Table 5.10

The application of the AEV was limited to values between 0 and 0.5 m for analyses that significantly contributed to the conclusion of this thesis in order to avoid erroneous results. This also captures most of the ranges provided in the table. For some other sensitivity analyses, not focusing on return period,

Table 5.10
Values of the air entry value (AEV) reported in literature.

AEV [m]	Notes	Source
0.33-0.5		Aubertin et al. (1998)
0.475	Drying curve	Zhang et al. (2016a)
0.12-1.25		Rassam (2002)
1.0		Pak and Nabipour (2017)

also larger values were used. The chosen value for the AEV is much smaller than was initially reported by Blight (2013) and discussed in Chapter 3. The difference may be with the type of tailings that are discussed. Blight (2013) provides with a value for gold tailings which tend to have a smaller grain size and therefore possibly a higher air-entry value.

Anisotropy of permeabilities
For use within the Modflow model for the initial conditions, a value for the anisotropy is needed. Also for the drainage package of PySWAP, a value of the horizontal hydraulic conductivity is necessary. The horizontal hydraulic conductivity may be calculated from the vertical hydraulic conductivity by multiplication with the anisotropy defined as k_h/k_v. Based on the fact that this study applies a quasi-2D analysis, it is important not to put too much emphasis on the value of the anisotropy. Any sensitivity study will likely not be a realistic representation as a large part of the analysis focuses on vertical flow only. Within that retrospect, only a single value of the anisotropy will be applied. Based on several sources, including Robertson et al. (2019), Morgenstern et al. (2016) and Zandarín et al. (2009), a value of 5 will be applied in all analyses for the anisotropy.

5.4. Numerical parameters

There is a substantial amount of numerical parameters within both the PySWAP and the differential evolution failure surface algorithm. For completeness of this chapter, these values will be provided here as well, as to make the research easily reproducible. For the definition of these parameters, the reader is referred to appendix D.

5.4.1. Numerical parameters PySwap

In Table 5.11, the numerical parameters, their value and a short description of their meaning are provided with.

5.4.2. Numerical parameters limit equilibrium and failure surface

The used numerical parameters in the differential evolution algorithm are shown in Table 5.12

Table 5.11
Numerical parameters for the PySWAP code.

Parameter	Value	Description
dTMin	1e-7	Minimum allowed time step
dTMax	0.3	Maximum allowed time step
MaxIt	75	Maximum number of iterations within a time step
maxBackTr	8	Maximum number of allowed back-tracing steps within the Newton Raphson algorithm
CritDevBalCp	1e-5	Convergence criterion for the maximum offset allowed in the water balance for a node
CritDevh2Cp	1e-2	Convergence criterion for the absolute difference in pressure heads between time steps
CritDevh1Cp	1e-2	Convergence criterion for the relative difference in the pressure heads time steps
CritDevBalTot	1e-4	Convergence criterion for the maximum cumulative offset allowed in the water balance

Table 5.12
Differential evolution algorithm parameters.

Parameter	Value	Description
noSlicesNonCirc	5	number of degrees of freedom -1 (optimisation points)
DivisionSlice	4	number of added slices on each segment between degrees of freedom
popSize	20*	the population size within the differential evolution algorithm
numGenerations	700	maximum number of generations within the differential evolution algorithm
mut	0.5	parameter within the difference formulation of the mutant
crossp	0.7	if a random value in the domain [0,1] for the uniform distribution is below crossp, crossover is performed between the mutant and the family member
avgMovement	0.002	the average movement of the members of the population after each generation, that is satisfactory low
stopCounter	22	the number of generations in a row that avgMovement has to be satisfied in order to stop the simulation

* Population size was increased for a duration of 5 days to reduce the stochastic noise for this critical duration.

CHAPTER 6

Results of the analyses

This chapter will treat the results of the analyses that have been performed. Together the results describe how rainfall affects the probability of static liquefaction of tailings dams. The results therefore form the central element for answering the main research question of the thesis. All analyses that have been performed can be divided into three different categories:

- General analyses: these analyses focus on the behaviour of the wetting front and the factor of safety to static liquefaction for different combinations of parameters. The numbering of the analyses follows the form GA#, where # is replaced with a number. These studies are treated in Section 6.1

- Return periods analyses: these analyses focus on the calculation of the return period of failure. The numbering of the analyses follows the form RP#, where # is replaced with a number. The return period studies are treated in Section 6.2.

- Miscellaneous analyses: a set of smaller studies within this thesis that did provide interesting results but did not contribute to the answering of the main research question. It also includes those studies of which it was argued that they surpassed the limitations of the modelling strategy and results should therefore be interpreted with caution. These analyses have been included in Appendix C.

6.1. General analyses

The following list provides with a short description of each of the general analyses that have been performed:

- **GA01:** analysis on the development of the FoS over time for different shell widths of non-liquefiable material.

- **GA02:** detailed analysis of the progression of the wetting front and phreatic surface upon rainfall infiltration.

- **GA03:** lowest realisation of the FoS as a function of intensity.

- **GA04:** influence of height of the impoundment on wetting front behaviour and resulting FoS.

- **GA05:** analysis on the sensitivity of the FoS over time to the Mualem-van Genuchten equation parameters: α_{mvg}, n_{mvg} and AEV as well as the saturated hydraulic conductivity k_{sat}.

The following sections will treat each of these analyses. In each section, at first the used parameters and main goal of the study are presented. This is followed by a general conclusion on the results which provides with a quick overview of the main outcome. After that, each of the key findings in analysis is elaborated upon in further detail.

6.1.1. GA01

Analysis GA01 focused on the shell width of the non-liquefiable zone of the tailings impoundment and its influence on the lowest factor of safety occurring after rainfall. The parameters applied in the analysis can be seen in Table 6.1. The study was performed for a total of four shell widths (5, 10, 15 and 20 meters) over a period of 200 days. The intensity of rainfall was 0.2 m/day for a period of 5 days. Also presented in Table 6.1 are some of the key findings of this analysis. These key findings are further elaborated upon later in this section.

In general it was concluded that there is the tendency to develop deep slip surfaces, resulting in the fact that the lowest factor of safety occurs long after rainfall. The failure mechanism identified concerned a loss of suction in the unsaturated zone as well as a strength reduction due to recharge to the water table. The development of deep slip surfaces also minimised the effect of the shell width below a certain treshold. If the shell width was further increased, the influence of rainfall on the FoS was significantly decreased as strength reduction through a rise of the phreatic table was minimised.

Table 6.1
Analysis parameters of GA01

Goal	Sensitivity of the change of FoS for different shell widths, but also identify when the lowest FoS is to occur.
Geometric parameters	Height of 20 meters; slope of 3H:1V; shell width = varying
Strength parameters	$\varphi'_{liquefiable} = 40°$; $\varphi'_{non-liquefiable} = 35°$; $s_u/\sigma'_v = 0.37$
Geohydrologic parameters	$\theta_s = 0.4$; $\theta_r = 0.05$; $k_{sat} = 0.1$ m/day; $n_{mvg} = 1.7$; $\alpha_{mvg} =$ 2.0 1/m; $AEV = 0.5$ m
Key findings	• Lowest factor of safety occurs between 20 and 30 days after the start of rainfall. • The reduction of the FoS following rainfall is in the order of 3% depending on the various parameters. • Tendency to develop deep slip surfaces through the liquefiable material. • Clear boundary of shell width beyond which the dam is stable. • Assumption of initial hydrostatic suction profile leads to too high suction values.

The following sections will treat each of the mentioned key findings. Presented are the observations and a discussion of those observations. Especial effort was expended to the explanation of the results in relation to the modelling strategy.

Lowest factor of safety occurs between 20 and 30 days after the start of rainfall, giving a reduction of the FoS of around 3%

Shown in Figure 6.1 is the normalised factor of safety following the days after rainfall for the impoundments with different shell widths. The FoS was normalised to the FoS at the start of rainfall for clear comparison between different shell widths. For the shell widths ranging from 5 meters to 15 meters, the lowest FoS is obtained at around 20 to 30 days after the start of rainfall. The low FoS is maintained for nearly 5 to 10 days after which it starts to increase again. The normalised reduction of the FoS is in the order of 3-3.5% with respect to the start of rainfall. The undrained failure mechanism is therefore relatively insensitive to the rainfall events that have been imposed. E.g. the same reduction in FoS might be obtained by steepening the slope of the dam by only 2 degrees which shows that any variability in some of the geometric parameters may have an even larger influence.

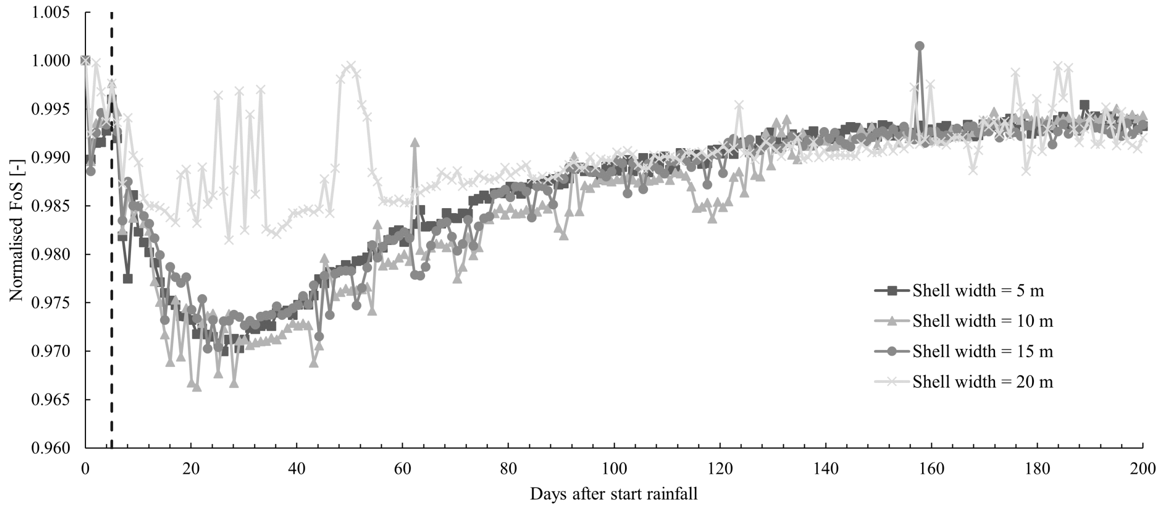


Figure 6.1: The normalised FoS for different widths of the dam shell, for given days after start rainfall.

Tendency for deep slip surfaces and discrete boundary on effect of shell width
Zooming in on the differences in the response to rainfall for different shell widths in Figure 6.1, for a shell width of 20 meters the sensitivity of the FoS to rainfall is lower especially in comparison to a width of 15 meters. To compare the effect for these different shell widths, Figure 6.2 shows the degree of saturation and corresponding failure surface at the moment of lowest FoS for a shell width of both 15 and 20 meters.

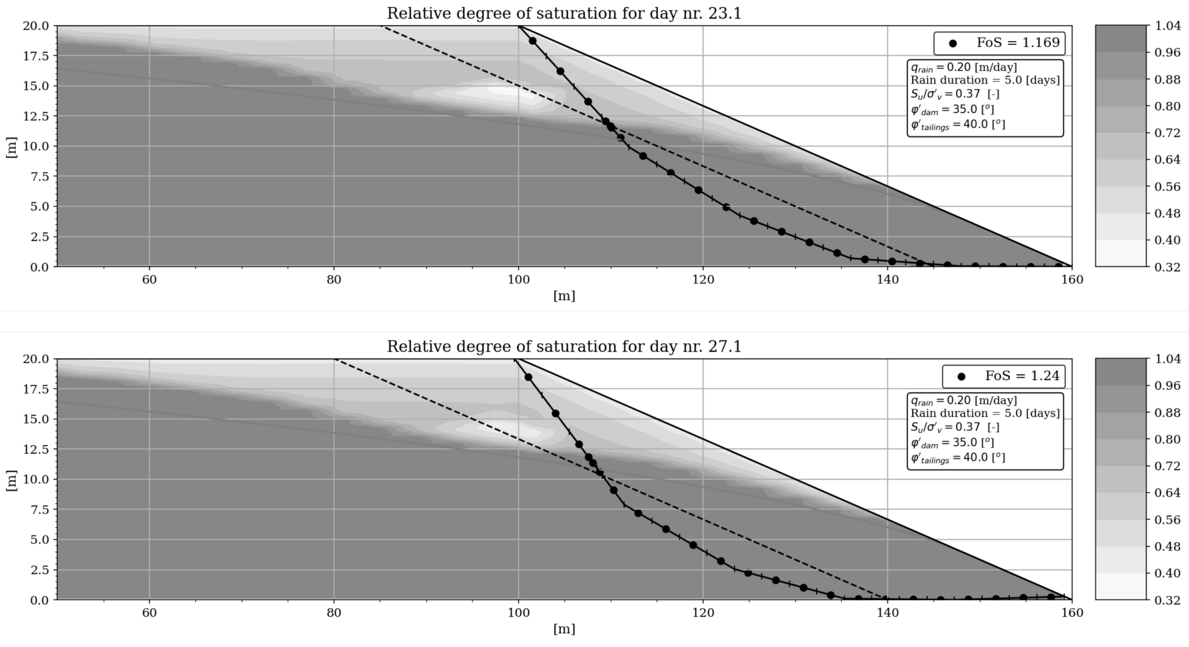


Figure 6.2: The degree of saturation and resulting failure surface at the moment of lowest factor of safety for a shell of 15 meters (top) and 20 meters (bottom).

Shown is the tendency for deep slip surfaces to form due to the significantly lower strength of the saturated liquefiable tailings material. For a width of 15 meters, the failure surface aims to combine the loss of suction as well as a rise of the phreatic surface. Because of mounding of the phreatic table, more material along the failure surface will change from a drained to an undrained state due to more material having an degree of saturation larger than 90%. Furthermore, the pore pressure increases beneath the rising phreatic surface, reducing the effective stress. Following the strength ratios, the undrained shear strength is reduced along the slip surface below. This mixed mechanism was also anticipated on in the literature study in Section 3.2. The deep mechanism also explains the sudden drop in FoS just after the start of rainfall and the short recovery following. This is the effect of quick recharge to the phreatic surface near the toe where the initial phreatic surface is located just below the surface. This recharge is quickly drained away after.

The mixed mechanism cannot occur for a shell width of 20 meters. The shell now extends to the zone in which the rise of the phreatic surface occurs. As these tailings are now non-liquefiable there is not such a large strength decrease upon saturation as would occur for the liquefiable tailings. The only strength loss caused by a rise of the phreatic surface is the earlier mentioned decrease in the vertical effective stress along the undrained part of the slip surface. These results show that the effect of rainfall can significantly be reduced when the non-liquefiable zone extends far into the impoundment. Something that was also addressed in the guidelines on the minimum length of the drained beach as discussed in Section 3.1.

Impoundments with a shell width of 5 or 10 meters show the same normalised FoS behaviour after rainfall as for 15 meters width. This is amplified by the tendency for deep slip surfaces to form but was also computationally imposed. The range of x-coordinates over which the slip surface was allowed to enter the slope is limited to the top of the slope. In this case, the slip surface could enter between a horizontal coordinate of 70 to 100 meters. In chapter 2 the assumption was made that the slope would be covered by vegetation. The slope is therefore reinforced, increasing the factor of safety with regards to a shallow slip surface. Also Brett et al. (2012) argue that shallow slip surfaces should be neglected in analysis as they will usually not occur in practice. Nevertheless the results of a comparison study for a fixed entry range and a variable one can be seen in Figure 6.3. These results are accompanied by Figure 6.4, showing the relative degree of saturation and governing failure surface.

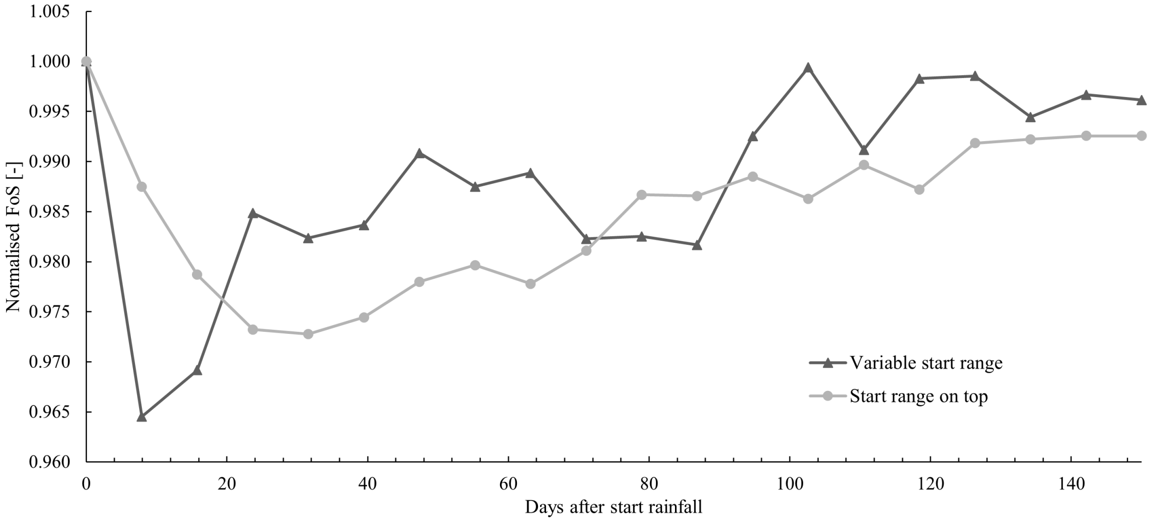


Figure 6.3: The FoS for the standard rainfall implemented in analysis GA01, for both a variable entry point of the slip surface as well as one that is restricted to start on the non-sloping part of the impoundment.

The assumption of a fixed entry point range considerably influences the effect of rainfall on the FoS. The normalised FoS decreases for the variable entry point to around 0.965 in contrast to 0.973 for the fixed entry point range. Like in earlier presented results, for the variable entry point the trajectory through the undrained zone is maximised and that part of the slip surface through the drained zone is minimised.

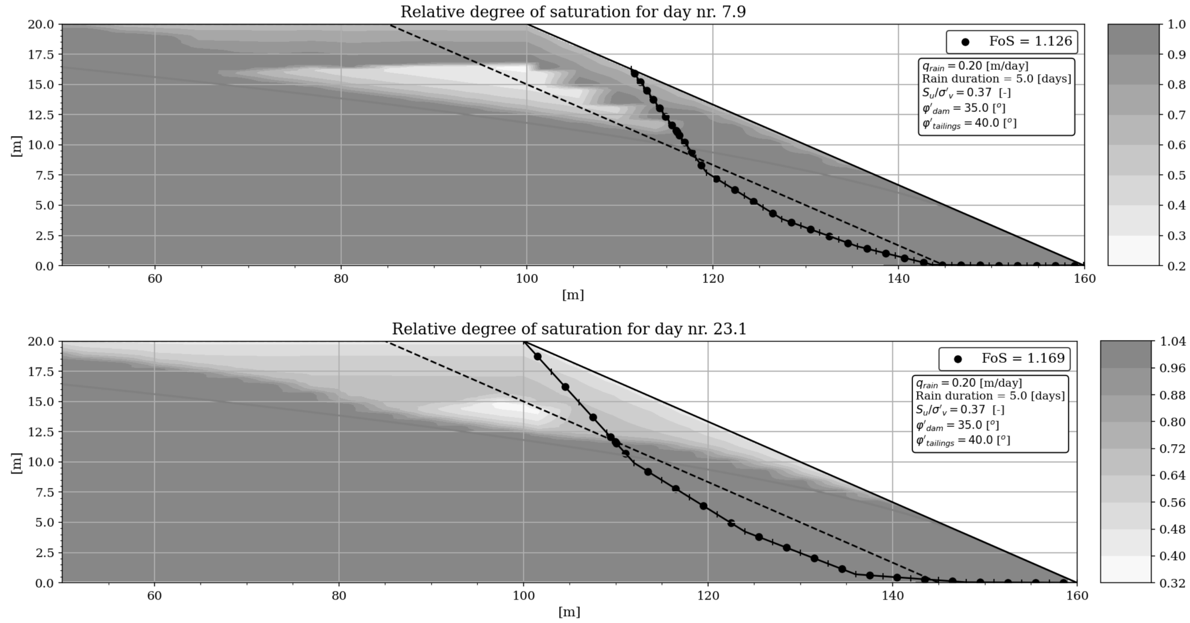


Figure 6.4: The relative degree of saturation over the impoundment as well as the critical failure surface for the moment that the lowest FoS is realised for a variable entry point (top) and limited range of the entry point (bottom).

The difference between the two simulations is that for a variable entry point the slip surface is able to fully profit from the loss of suction whereas this is not the case for the fixed entry point slip surface. The lowest FoS is therefore also found earlier after the start of rainfall. The exit point of the slip surface converges for both simulations to the toe of the slope.

Returning to Figure 6.3, it is also observed that the FoS is highly fluctuating for the simulation with the variable start range. This fluctuation is not due to any physical phenomena. Instead, it is generated by a combination of computational aspects. The first aspect is the stochastic nature of the differential evolution approach. Due to this, the algorithm may get stuck in local optima, therefore showing a small oscillation in the found FoS for consecutive days. Furthermore, the quasi-2D approach causes additional oscillations in the solution due to the interpolation that is performed. For the variable entry point simulation, there is a wider solution space available. Therefore, it is more difficult to converge to the governing slip surface, increasing the fluctuation. Adding to that the earlier mentioned consideration of vegetation, it was decided to restrict the entry point of the failure surface to the top of the slope.

Assumption of initial hydrostatic suction profile is too strict

In Figure 6.1, the normalised FoS asymptotically converges to a value of 0.995 and does not restore to 1. Figure 6.5 shows the degree of saturation at the start of rainfall and at the end of the simulation period of 200 days. Zooming in on the results, it is shown that the minimum relative degree of saturation is 0.15 at the start of rainfall whereas this is 0.3 long after. Due to this higher degree of saturation, the strength is reduced long after rainfall has stopped, although only to minimal extent.

Practically speaking, within such a period of 200 days there will likely be additional rainfall and the suction profile as assumed at the start of the simulation period will not occur. The initial hydrostatic suction profile results in too high values of the negative pressure head. Yet, results considering the trailing edge of the wetting front should be interpreted with caution as there is no distinction between the wetting- and drying curve of the SWCC in analysis. It is also unknown what the suction profile is in decommissioned tailings dams. As an alternative to the hydrostatic assumption one may impose a maximum suction (Kim et al., 2012) or perform a spin-up analysis. If the maximum suction is lower, the effect of rainfall is potentially larger as the wetting front speed is increased and the initial strength is higher because of initially higher degrees of saturation.

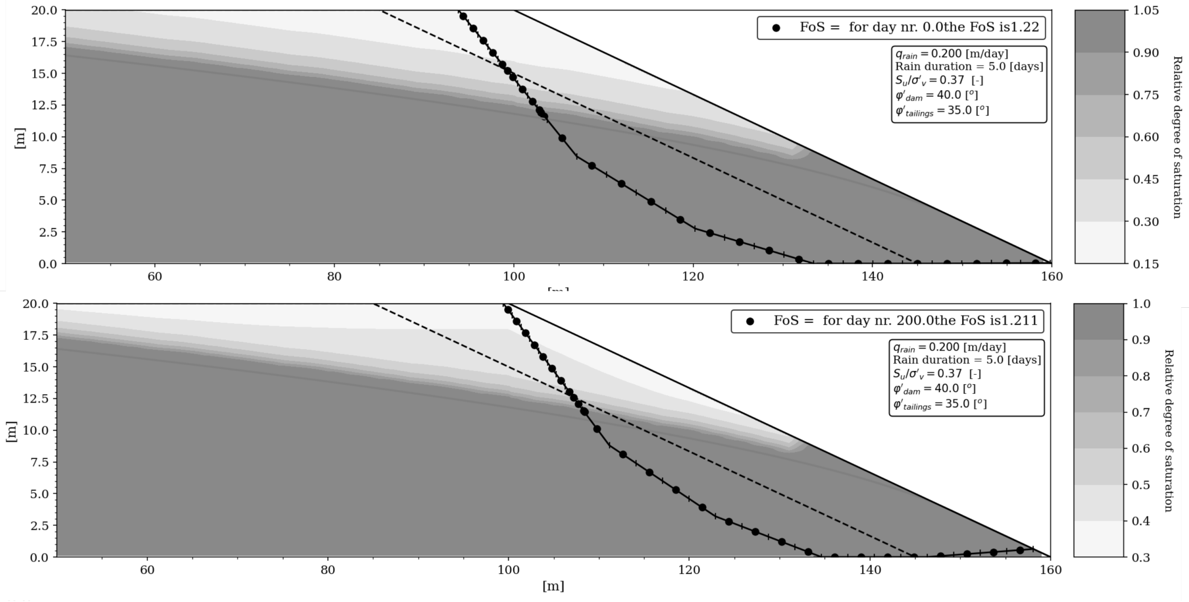


Figure 6.5: The degree of saturation and critical failure surface before rainfall and 200 days after.

6.1.2. GA02

Analysis GA02 focuses on the behaviour of the wetting front upon rainfall infiltration and was performed with the parameters as shown in Table 6.2. In contrast to analysis GA01, a reduced intensity of 0.1 m/day was used for a duration of 5 days.

For all studies, the wetting front and phreatic surface were studied based on the relative degree of saturation. To be able to observe their behaviour, a figure has been assembled in which the wetting front is shown for ten unequally divided steps ranging from 0 to 50 days. Some more figures are provided for the early days after the start of rainfall as these show the progression of the wetting front best. The final assembly of results can be seen in Figure 6.6 at the end of this section. It was concluded that the quasi-2D approach is capable of qualitatively mimicking most characteristics of two-dimensional unsaturated flow.

Table 6.2
Analysis parameters of GA02

Goal	Identification of the wetting front behaviour.
Geometric parameters	Height of 20 meters; slope of 3H:1V;
Geohydrologic parameters	$\theta_s = 0.4$; $\theta_r = 0.05$; $k_{sat} = 0.1$ m/day; $n_{mvg} = 1.7$; $\alpha_{mvg} = 2.0$ 1/m; $AEV = 0.5$ m
Key findings	• Non-linear shape of SWCC reflected in initial conditions of the relative degree of saturation. • Wetting front speed affected by the SWCC and relative degree of saturation. • Recharge to the water table more pronounced with increasing distance from the toe.

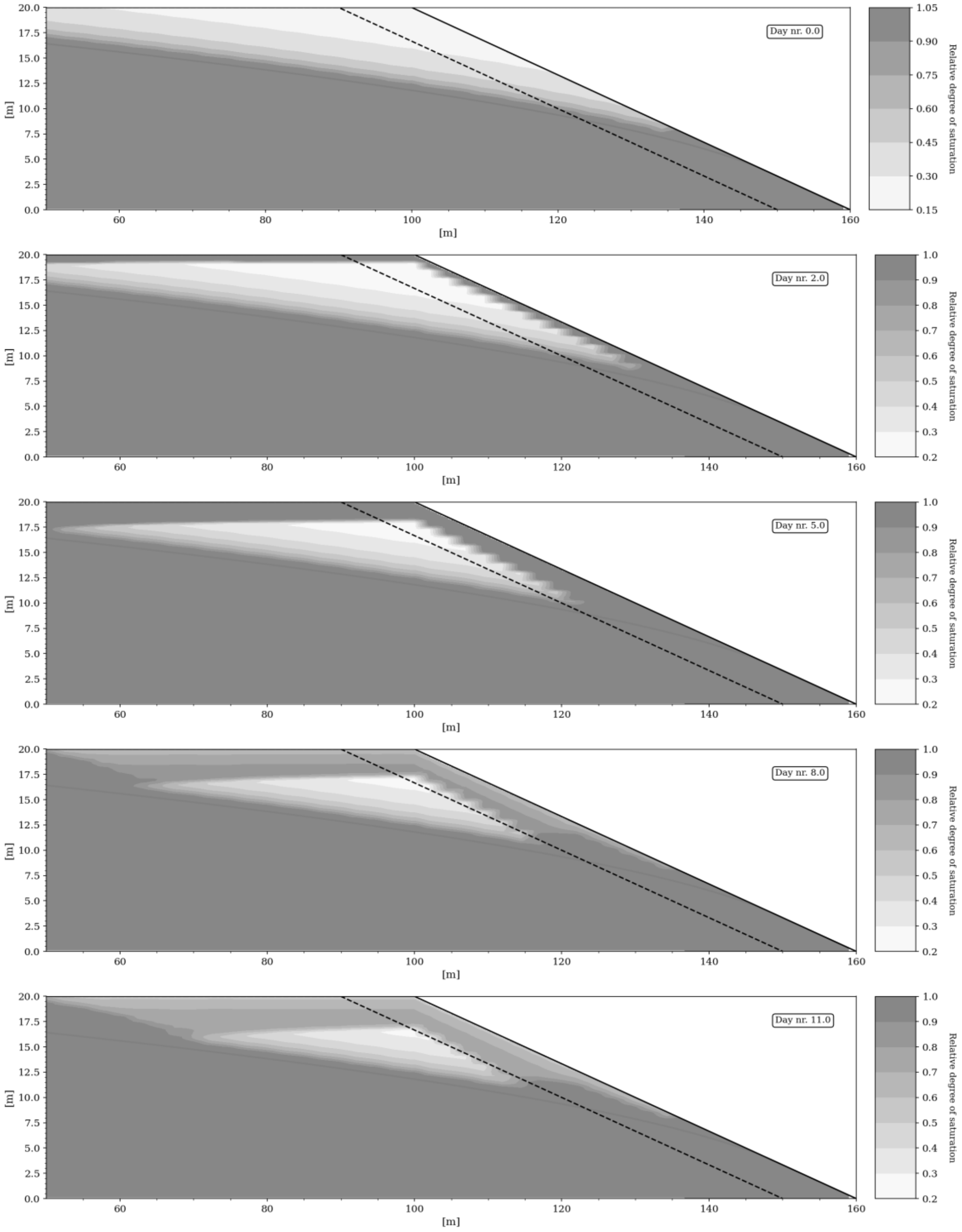
The key findings, as shown, will be treated in each of the sections. These findings follow a chronological approach after the start of rainfall.

Shape of SWCC reflected in initial conditions of the relative degree of saturation
Starting with the relative degree of saturation at day 0, it is observed that, because of the choice of the various Mualem-van Genuchten equation parameters, the unsaturated zone holds a minimum degree of saturation of 0.15, meaning that the residual water content is not reached. The very fine dark grey line is the initial phreatic surface based on the boundary conditions imposed at the upstream and downstream side of the impoundment. Above the phreatic surface the degree of saturation is still equal to 1 which is the capillary rise in the unsaturated zone. Advancing from the phreatic surface upwards, the contours have increased spacing which is the result of steepening of the SWCC towards more negative pressure heads, showing the non-linear effect of this curve.

Where the phreatic surface almost hits the slope, the saturated zone does not follow the phreatic surface. A computational simplification was imposed to avoid complete saturation of the water column upon rainfall, causing numerical instability. It is believed by the author that this numerical instability partially arose from the fact that the infiltration boundary condition by SWAP (and implemented in PySWAP) uses an explicit form of the water balance to calculate the total amount of ponding. This does not perform well in combination with the implicit water balance that is eventually solved for. As the contribution of this part to the strength of the impoundent is negligible, it was chosen to skip these columns in analysis and assume the phreatic surface to be equal to the slope level during the entire simulation period.

Wetting front speed affected by the SWCC and relative degree of saturation
In between day 5 and 8 the wetting front has moved approximately 0.2 meters in the vertical direction, equating to a wetting front ‘celerity’ of 0.07 meters per day. The wetting front celerity is lower than the saturated hydraulic conductivity of the tailings caused by the initially lower hydraulic conductivity in the unsaturated zone (see Equation 4.18). Another related observation is that with shorter distance to the groundwater level the wetting front moves faster vertically as the average relative degree of saturation is higher along such a profile. The hydraulic conductivity along this same vertical is higher and the needed water content to saturate the pores is lower. This phenomenon results in a curved wetting front along the non-sloping part of the impoundment. The effect is also visible in the sloped part, but is less distinguished because of the quasi-2D approach.

Recharge to the water table more pronounced with increasing distance from the toe
At day 5, the wetting front has infiltrated to the phreatic surface near the dam face. Somewhere between day 5 and 8 the same happens at an x-coordinate of 50 meters. The phreatic surface shoots up and recharge to the water table has occurred at this point. The total rise of the phreatic surface is determined by the total precipitation depth, the shape of the SWCC, and the value of the AEV. After recharge the phreatic surface is lowered by horizontal drainage. This feature is shown clearly in the figures of day 11 and day 15. For these days, near the slope face the recharge that was added has already substantially been decreased. The recharge to the water table is smaller than the drainage capacity at this point and the phreatic surface is lowered. On the far left, the drainage capacity is lower. As was given in Equation 4.20, the drainage resistance is proportional to the square of the distance from the toe. This means the total horizontal flow is inversely proportional to the square of this distance. In this way, the recharge at the far upstream side will drain away less quickly then near the slope face, therefore showing higher phreatic levels than near the dam face.


Day nr. 0.0
Day nr. 2.0
Day nr. 5.0
Day nr. 8.0
Day nr. 11.0
[m]
[m]
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
60
80
100
120
140
160
Relative degree of saturation
0.15
0.30
0.45
0.60
0.75
0.90
1.05
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0

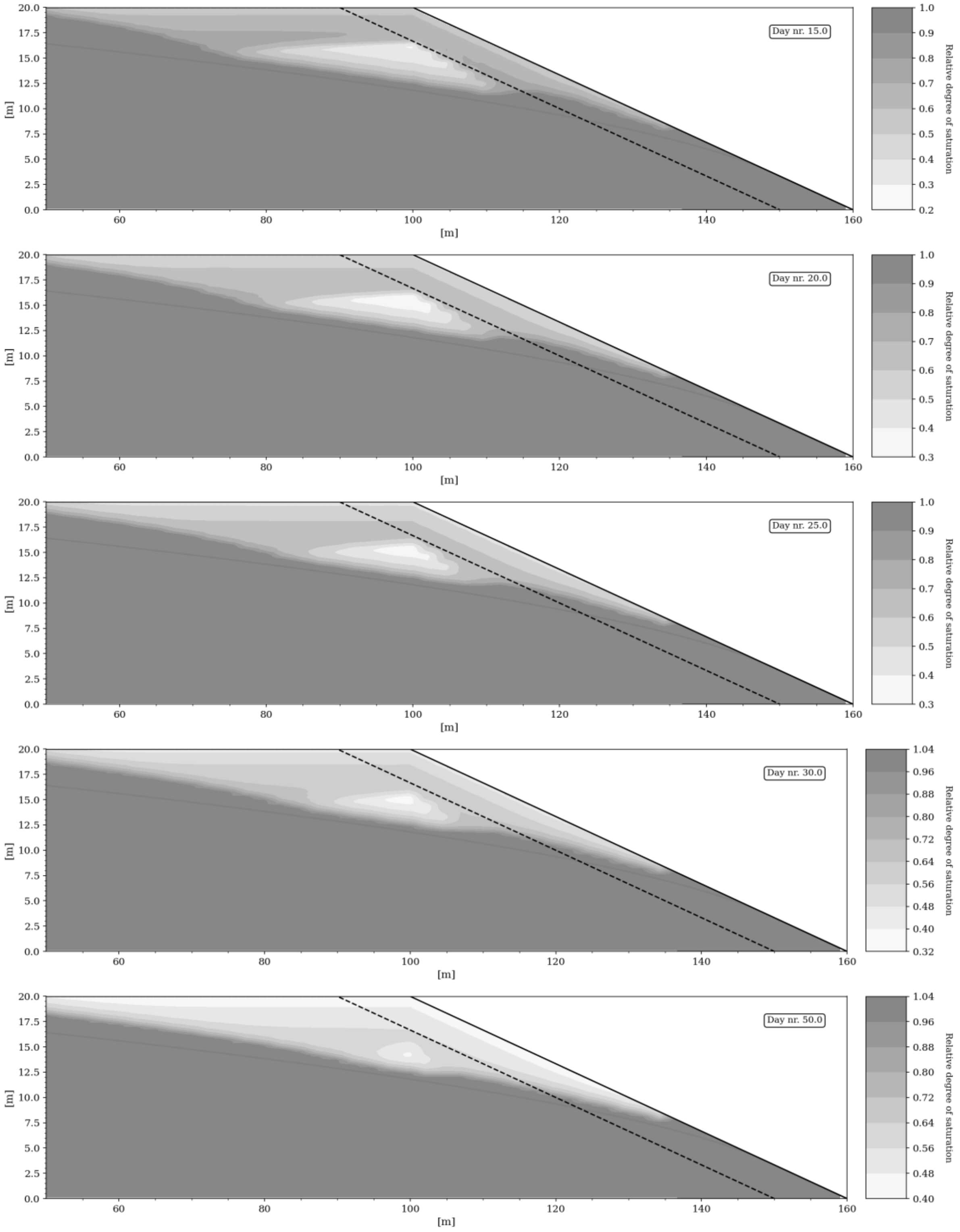


Figure 6.6: The relative degree of saturation upon rainfall infiltration for unequally divided moments between 0 and 50 days.

6.1.3. GA03

A series of simulations was run with increasing intensity to observe the influence of the intensity of rainfall on the FoS. The analysis was performed for an intensity range between 0.01 and 0.2 m/day, for durations of 3, 4 and 5 days. The other parameters applied as well as key findings in the results are shown in Table 6.3.

It was found that for all durations of rainfall the same qualitative behaviour of the FoS is shown with increasing intensity. Three branches may be discerned in the response: for low values of the intensity (<0.06 m/day) there is only a loss of suction which contributes to a decrease in the FoS; for intensities between 0.06 and 0.2 m/day, there is an increased influence of rainfall as the mixed mechanism of loss of suction and rise of the phreatic surface gets effective; for higher intensities, the influence of rainfall is significantly decreased due to the high amount of run-off that occurs. It is stressed that the mentioned values of intensity may differ depending on the parameters used.

Table 6.3
Analysis parameters of GA03

Goal	Influence of the intensity on the minimum FoS in analysis for different durations of rainfall.
Geometric parameters	Height of 20 meters; slope of 3H:1V; shell width = 10 m
Strength parameters	$\varphi'_{liquefiable} = 37°$; $\varphi'_{non-liquefiable} = 32°$; $s_u/\sigma'_v = 0.37$
Geohydrologic parameters	$\theta_s = 0.4$; $\theta_r = 0.05$; $k_{sat} = 0.1$ m/day; $n_{mvg} = 1.7$; $\alpha_{mvg} = 2.0$ 1/m; AEV = 0.5 m
Key findings	• The qualitative behaviour of the relation between the FoS and the intensity is the same for all considered durations. • For lower intensities the FoS is reduced by a loss of suction only. • For higher intensities, a mixed mechanism of loss of suction and rise of the phreatic surface is becoming more pronounced, increasing the influence of rainfall. • At very high intensities, run-off will cause a clear reduction of the influence of rainfall on the FoS.

The qualitative behaviour of the relation between the FoS and the intensity is the same for all considered durations

For each combination of duration and intensity the lowest calculated FoS over a simulation period of 60 days was recorded. The final results are shown in Figure 6.7 with the regular FoS displayed against the intensity. As only the duration of rainfall was varied between the simulations, there was no need for normalising the results.

For intensities between 0.01 and 0.06 m/day, there is a gradual decrease of the FoS with increasing intensity. The relationship between both may be characterised as linear. For higher intensities, the qualitative behaviour for all durations is equal. The FoS decreases with increasing intensity but shows a steeper inclination than earlier, indicating an increased effect of rainfall. This steepness is maintained until an intensity of around 0.14-0.16 m/day, followed by a significant drop in FoS for higher intensity showers. After that, the FoS stabilises and is not further affected by even larger intensities of rainfall. In order to explain the qualitative observations of the results, the failure mechanism of four different intensities must be compared: 0.05, 0.1, 0.14 and 0.20 m/day. For that purpose, Figure 6.8 holds the visualisations of the governing failure surface (lowest FoS) and corresponding degrees of saturation for these four intensities and a rainfall duration of 5 days.

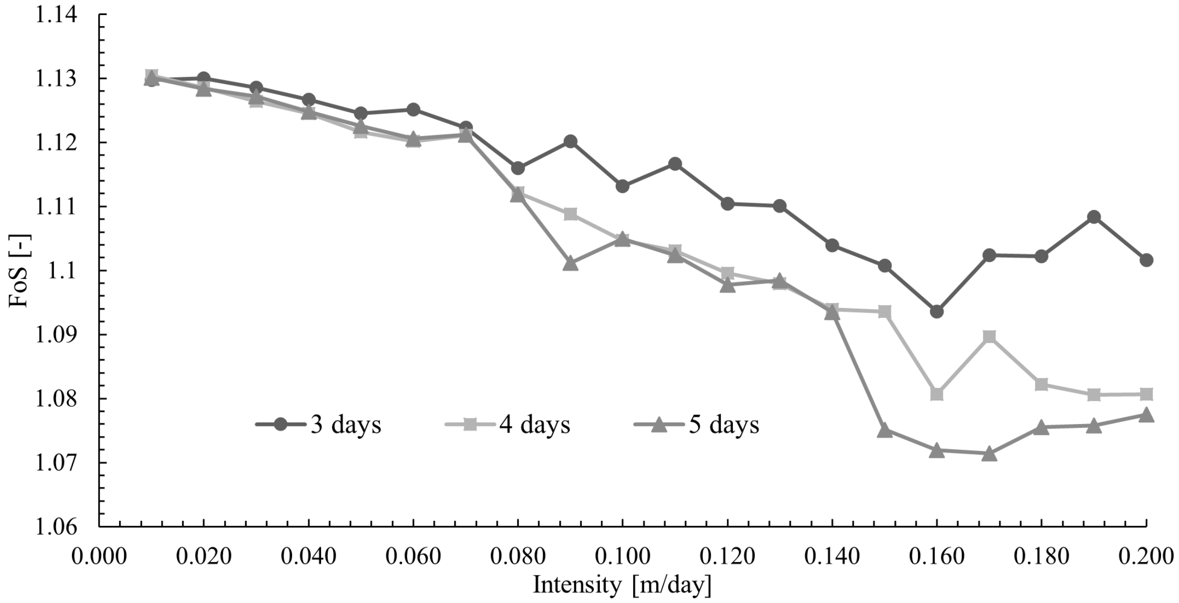


Figure 6.7: The lowest recorded factors of safety upon rainfall infiltration for different combinations of intensity and duration.

For lower intensities the FoS is reduced by a loss of suction only
For an intensity of 0.05 m/day, it is shown in Figure 6.8 that the lowest FoS occurs at day 45 after the start of rainfall. The wetting front has passed to the phreatic surface at the upstream and downstream side of the impoundment. Just left of the governing failure surface the wetting front has not yet reached the phreatic surface. The total precipitation depth is too low to still have a considerable wetting front after it has passed over a few meters. The wetting front slows down and will not penetrate deeper into the impoundment as the hydraulic gradient has become too low with respect to the unsaturated hydraulic conductivity. Any decrease in the FoS is caused by a loss of suction in the upper zone. No increase of the phreatic surface is visible as the recharge rate is lower than the rate of possible drainage.

For higher intensities, a mixed mechanism of loss of suction and rise of the phreatic surface is becoming more pronounced, increasing the influence of rainfall
For an intensity of 0.1 m/day, the lowest FoS is realised later after the start of rainfall compared to the results at an intensity of 0.05 m/day. The wetting front has advanced deeper and also reached the phreatic surface around an x-coordinate of 100 meters. The failure surface is able to catch the effect of recharge to the water table which explains the bi-linearity at an intensity of 0.06 m/day in Figure 6.7. The mixed mechanism has become effective. Judging from the slopes of the two parts of the bi-linear trajectory in Figure 6.7, it seems that the effect of loss of suction and that of the recharge to the water table is roughly equal, although due to the jumps in calculated FoS this is difficult to appraise.

The following plot in Figure 6.8 corresponds to an intensity of 0.14 m/day. The FoS is declining at a constant rate with increasing intensity because of an increased loss of suction and more recharge to the water table. The lowest FoS is obtained earlier after the start of rainfall. Mainly because the wetting front speed is higher with higher intensity because of the larger hydraulic gradients that are achieved. Moreover, there is additional water available to saturate the pores. There are no clear differences in the behaviour of the wetting front and the failure surface for intensities between 0.1 and 0.14 m/day.

At very high intensities, run-off will cause a clear reduction of the influence of rainfall on the FoS
With a large intensity of 0.2 m/day, the visual behaviour of the failure surface is equal to that of 0.1 and 0.14 m/day. Yet, the recharge is larger. No clear observations were done that explain the sudden significant drop in FoS for intensities higher than 0.14 m/day. The FoS levels off after, created by

the large amounts of ponding and run-off at these high intensities. With respect to the choice of the value of k_{sat}, the intensity for which run-off occurs is high. Normally run-off is expected to occur at intensities equal to the saturated hydraulic conductivity which would be 0.1 m/day in this case as also shown in e.g. Rahardjo and Fredlund (1991) and Ng and Shi (1998). This difference is attributed to the fact that the initial infiltration capacity is large because of high suction values in the upper zone. Furthermore, a relatively large amount of ponding is allowed. This amount of ponding was chosen to provide with more conservative results. However, in reality tailings impoundments also hold a large ponding capacity because of the high freeboards to prevent overtopping (Brett et al., 2012).

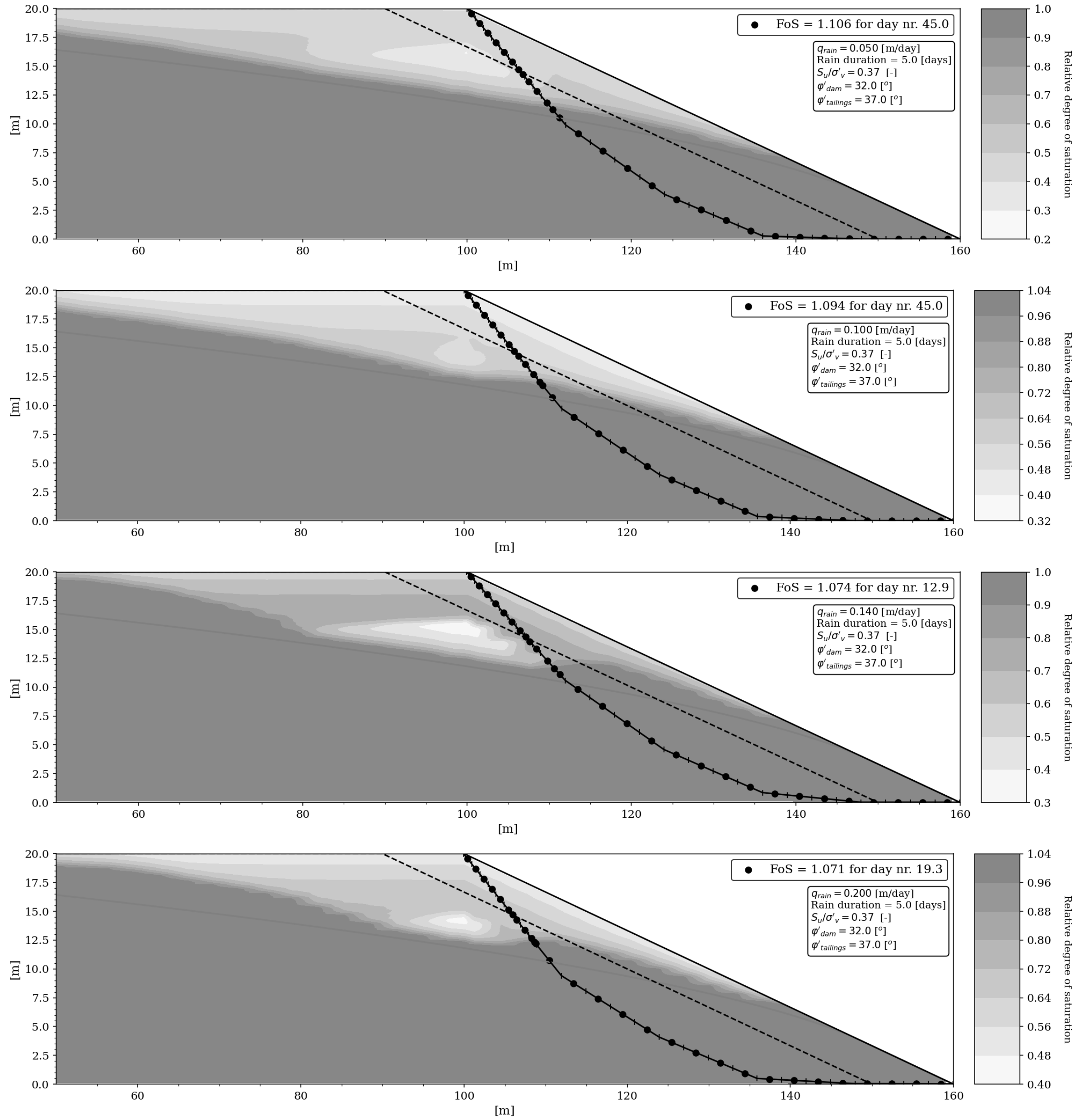


Figure 6.8: The governing failure surface and corresponding degree of saturation for each of the different intensities of: 0.05, 0.1, 0.14 and 0.2 m/day.

6.1.4. GA04

Analysis GA04 has focused on how the height of the tailings impoundment influences the effect of rainfall on its stability. A simulation of a rainfall event of 0.1 m/day for a duration of 5 days has been performed on impoundments with heights of 20, 60 and 100 meters. Table 6.4 holds the values applied for the different parameters as well as the key findings in the results. The shell width is equal to half the height for all simulations. The slope was maintained at 3H:1V. The width of the impoundment was also scaled to maintain the shape of the phreatic surface, allowing for a clear comparison of the effect of height rather than the implicit effect of unscaled parameters.

It was found that the effect of rainfall is significantly reduced for higher dams due to a larger unsaturated zone and a reduced wetting front speed. Due to the minimal length of this analysis, the following will elaborate on the key findings but the same structured approach as used earlier will not be applied.

Table 6.4
Analysis parameters of GA04

Goal	Influence of the height on the FoS after rainfall for the impoundment.
Geometric parameters	Variable height; slope of 3H:1V; variable shell width
Strength parameters	$\varphi'_{liquefiable} = 35°$; $\varphi'_{non-liquefiable} = 32°$; $s_u/\sigma'_v = 0.33$
Geohydrologic parameters	$\theta_s = 0.4$; $\theta_r = 0.05$; $k_{sat} = 0.1$ m/day; $n_{mvg} = 1.7$; $\alpha_{mvg} =$ 2.0 1/m; AEV = 0.5 m
Key findings	• Lower dams are more sensitive to rainfall infiltration. • The height of the unsaturated zone influences the wetting front speed and final infiltration depth.

Figure 6.9 provides with the normalised factor of safety for a simulation period of 100 days. Only for the lower dam there is a reduction of the FoS. Observed is a highly variable FoS caused by a combination of errors in the differential evolution and quasi-2D approach. For the dam of 60 meters high there is a slower decline of the FoS. The FoS is also not reduced as much as for the lower dam. The minimum is reached after around 50 days and there is little recovery of the FoS. As for the higher dam of 100 meters, there is little to no effect of rainfall on the FoS, converging to a value of around 0.998. In order to be able to explain why there is such little influence of rainfall on the dams of 60 and 100 meters high, the governing failure surfaces for the three studied simulations are shown in Figure 6.10.

The distance from the slope to the phreatic surface increases with increasing height. Considering the initial hydrostatic distribution of the pressure head, the suction values near the top are larger, reducing the unsaturated hydraulic conductivity. Additionally, the amount of water needed to fill the pores is high. At some locations, the wetting front has only advanced to a depth of 4 meters at the end of the simulation period. The effect of rainfall is therefore minimal for higher dams as only a little part of the dam is affected. Moreover, the soil body is completely unsaturated and the additional effective stress is negligible at high suctions (see also Equation 3.1). Rainfall will therefore not have high effects as there is no additional strength to reduce upon wetting.

As indicated earlier, these high values of suction are unreliable which means these results should be interpreted with caution. There is however some possible truth in this analysis that higher dams are less probable to fail due to rainfall storm conditions. In an inverse manner this has been treated by Rico et al. (2008) who notes that of all tailings dams failures, most fail due to meteorological conditions and most failures are with smaller dams. Although a tie between those is not treated in that paper, it is likely that the two are connected. In light of these results, this study will continue to analyse smaller dams as the effects are largest for these.

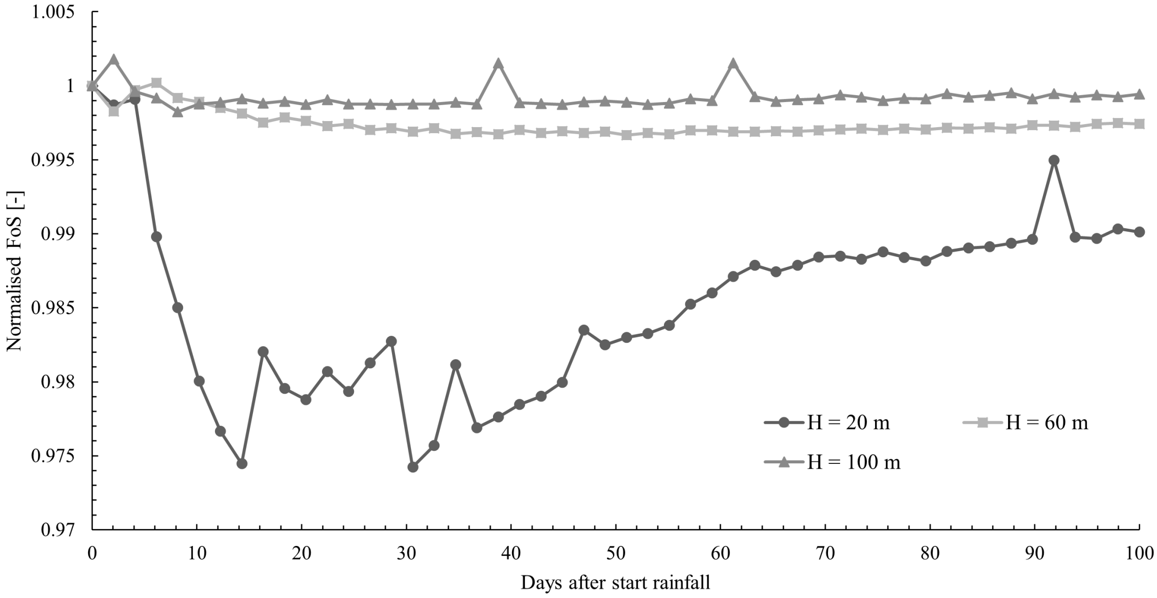


Figure 6.9: The normalised FoS for different heights of the impoundment following rainfall.

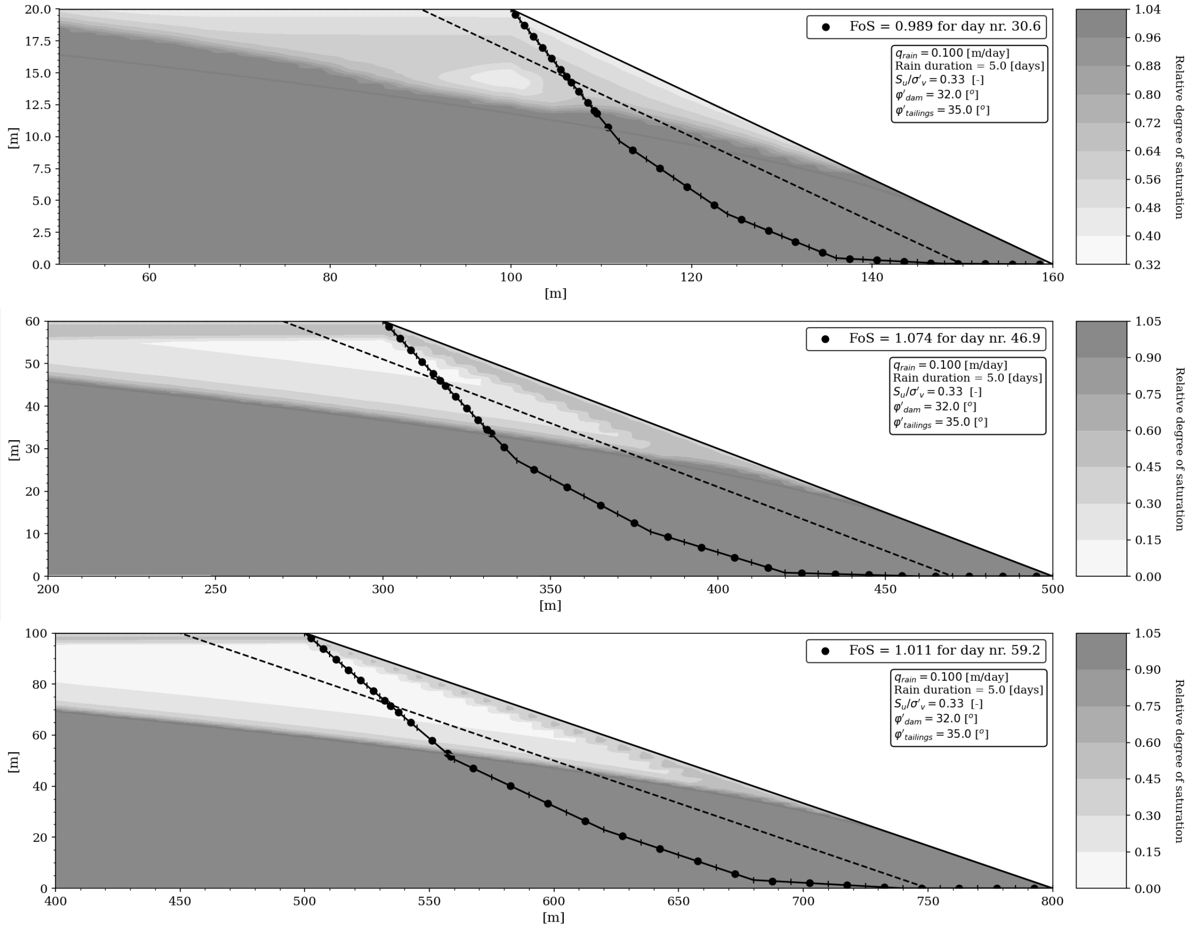


Figure 6.10: The degree of saturation and corresponding failure surfaces for an impoundment with a height of 20 meters (top), 60 meters (middle) and 100 meters (bottom).

6.1.5. GA05

This section will elaborate on the results of analysis GA05, in which the influence on the FoS after rainfall was examined for a range of different values of the geohydrological parameters: the Mualem-van Genuchten equation parameters of Equation 4.14 and the saturated hydraulic conductivity. The parameters used in the analysis as well as the key findings are shown in Table 6.5. The simulation was done for an intensity of 0.1 m/day for a period of 5 days.

Generally, with increasing values of α_{mvg} and n_{mvg} the influence of rainfall on the FoS decreases. Moreover, the moment at which the lowest FoS occurs is delayed. For an increasing AEV the influence of rainfall on the FoS increases. The observed influence of the Mualem-van Genuchten parameters is closely related to the initial degree of saturation in the profile. Higher degrees of saturation cause higher wetting front speeds. Also the initial strength from suction is higher according to Equation 3.1 increasing the effect of rainfall upon rewetting. The same is also observed for the saturated hydraulic conductivity. A higher hydraulic conductivity causes more effect of rainfall on the FoS, but also a relatively quicker recovery. The FoS can remain low for long periods of time when the hydraulic conductivity is reduced. However, when the tailings are very poor-draining, run-off will influence the results and the effect of rainfall on the FoS is reduced to almost zero. The importance of the saturated hydraulic conductivity has been highlighted already in Rahimi et al. (2010), but now also extends to deep slip surfaces. Even though the reduction of the FoS is far lower than for residual slopes only affected by a reduction of suction.

Table 6.5
Analysis parameters of GA05

Goal	Sensitivity of the change of FoS for different values of the Mualem-van Genuchten parameters: α_{mvg}, n_{mvg} and the AEV as well as the saturated hydraulic conductivity.
Geometric parameters	Height of 20 meters; slope of 3H:1V; shell width = 15 m
Strength parameters	$\varphi'_{liquefiable} = 40°$; $\varphi'_{non-liquefiable} = 31°$; $s_u/\sigma'_v = 0.35$
Geohydrologic parameters	$\theta_s = 0.4$; $\theta_r = 0.05$; if not varied in analysis: $n_{mvg} = 1.6$; $\alpha_{mvg} = 2.01/m$; $AEV = 0.5m$; and $k_{sat} = 0.0864m/day$
Key findings	• α_{mvg} has a low influence on the FoS, but its effect is maximised for values near 1. • Low values of n_{mvg} lead to increased recharge and an increased loss of strength. • High AEV values will increase the effect of rainfall. • k_{sat} will be of high influence when the tailings have a low hydraulic conductivity.

α_{mvg} has a low influence on the FoS, but its effect is maximised for values near 1

Four simulations were performed to observe how α_{mvg} influences the effect of rainfall on the FoS. These simulations were done for values of: 1.1, 2, 3 and 4. A value of 1 initiated some numerical instability and could not be performed without providing unreliable results. The final results can be seen in Figure 6.11.

α_{mvg} causes an effect on the FoS especially between a value of 1.1 and 2. For the values of 2, 3 and 4 the behaviour of the FoS after rainfall is approximately equal and a considerable part of the variation is attributed to the noise in the differential evolution process. For a value of 1.1, the lowest FoS is realised earlier after the start of rainfall and just before 30 days have passed. For the other values this

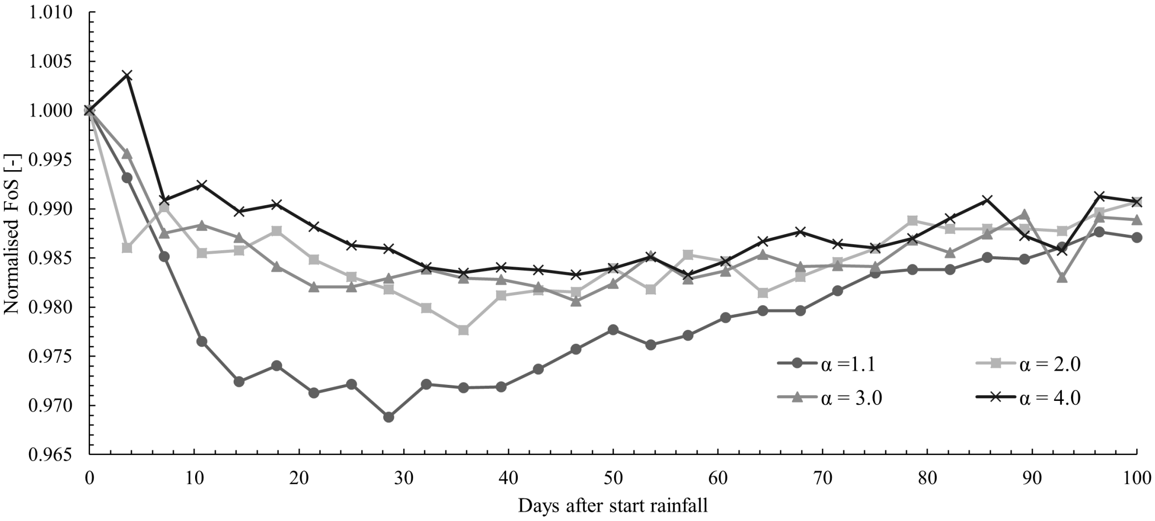


Figure 6.11: The change of FoS since the start of rainfall for various values of the parameter α_{mvg}.

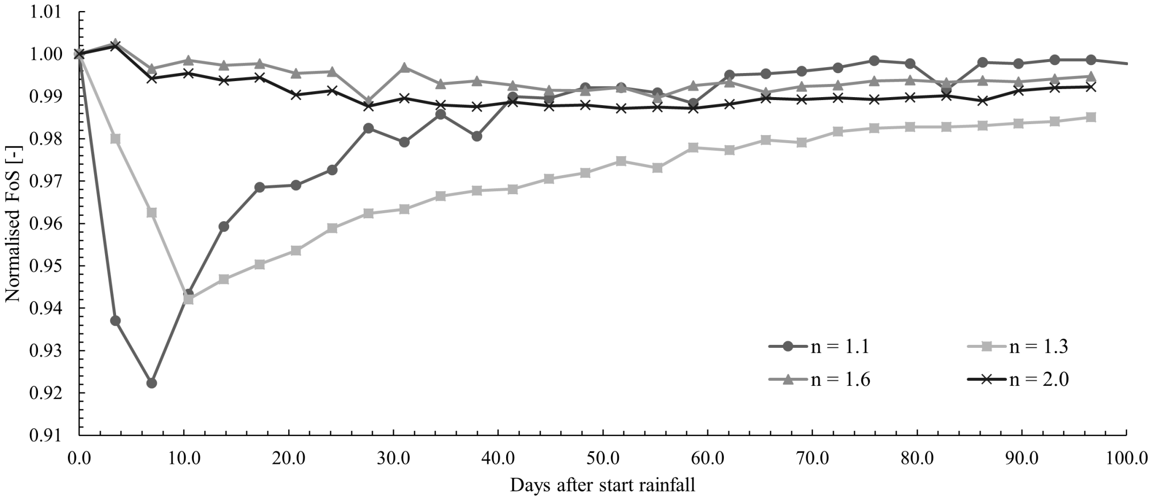


Figure 6.12: The change of FoS since the start of rainfall for various values of the parameter n_{mvg}.

is around 40 days. For a value of 1.1, the trough is deeper (lower normalised FoS), but is also shorter than compared to the other simulations. For a value of 2, the FoS remains constant between 20 and 70 days after the start of rainfall. The reduction of the normalised FoS is less and in the order of 1.5 to 2 %.

The parameter α_{mvg} is the head at which the SWCC is steepest. Following the original Mualem-van Genuchten equation (Equation 4.14, α_{mvg} is therefore the value of suction at the water content halfway between the saturated and residual water content. It is observed that lower values of α_{mvg} resulted in an increased wetting front speed, therefore higher recharge rates are possible, lowering the potential of drainage. This increases the effect of rainfall on the FoS.

Low values of n_{mvg} lead to increased recharge and an increased loss of strength

The normalised FoS after the start of rainfall for different values of n_{mvg} can be seen in Figure 6.12. For low values of n_{mvg}, the FoS is highly reduced. The reduction for a value n of 1.1 is in the order of 8% whereas the reduction is less distinct for the higher values of n_{mvg}. For lower values of the parameter the minimum FoS is also obtained earlier after rainfall than for the higher values.

n_{mvg} is related to the steepness of the SWCC at the same point as where α_{mvg} was defined. Lower values of n_{mvg} will increase the degree of saturation associated with higher negative pressure heads. This means that the additional strength according to Equation 3.1 is initially higher. The wetting front speed is increased as well caused by a higher unsaturated hydraulic conductivity for the same values of the negative pressure head. Analogous to what was observed for the value of α_{mvg}, higher wetting front speeds cause higher recharge rates. The increase of the phreatic surface is therefore larger and the strength of the impoundment is significantly lowered.

For illustration purposes, the critical failure surfaces for the moment of the lowest FoS are shown for values of 1.3 and 1.6 for n_{mvg} in Figure 6.13. The wetting front advances faster and recharge rates are higher for a lower value of n_{mvg}. The effect on the failure surface is considerable. It is shown that for a low value of 1.3, the lowest FoS is obtained at the moment that a significant part near the slope experiences a rise of the phreatic surface. As an effect, the vertical effective stress (and strength) are reduced along a large part of the failure surface. For a value of 1.6, this recharge is smaller and the lowest FoS occurs some time after. The rise is significantly lower and within the unsaturated zone, the degree of saturation has already decreased again, increasing strength. The FoS is in this way not influenced significantly upon rainfall.

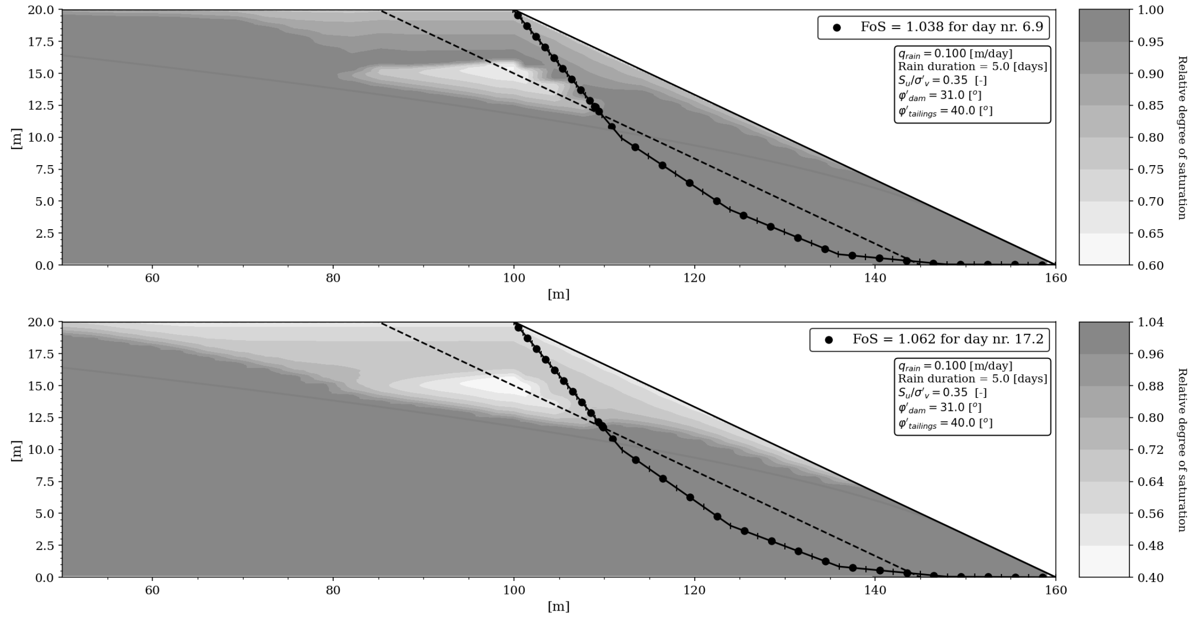


Figure 6.13: Comparison of the failure surfaces and wetting fronts for the n_{mvg} parameter. The top figure is related to a value of 1.3, the bottom figure to a value of 1.6.

High AEV values will increase the effect of rainfall

The same study was also done for the air entry value. In section 4.3.1 it was noted that for large values of the air entry pressure the results were not similar to other studies performed in literature, likely originating from the modelling approach. Though over a range of values, a similar analysis was performed to obtain some idea of the potential influence of the AEV. The resulting factors of safety can be seen in Figure 6.14.

The results show that with increasing air-entry pressure the total reduction of the FoS is lower and that the lowest FoS occurs earlier after rainfall. This may be attributed to the fact that with higher air-entry pressure the degree of saturation is larger above the phreatic surface. This causes an increased wetting front speed. Moreover, the shape of the wetting front is significantly affected. As observed in Figure 4.12, the wetting front will be less sharp for higher values of the AEV. Therefore the reduction of suction along the entire column is influenced.

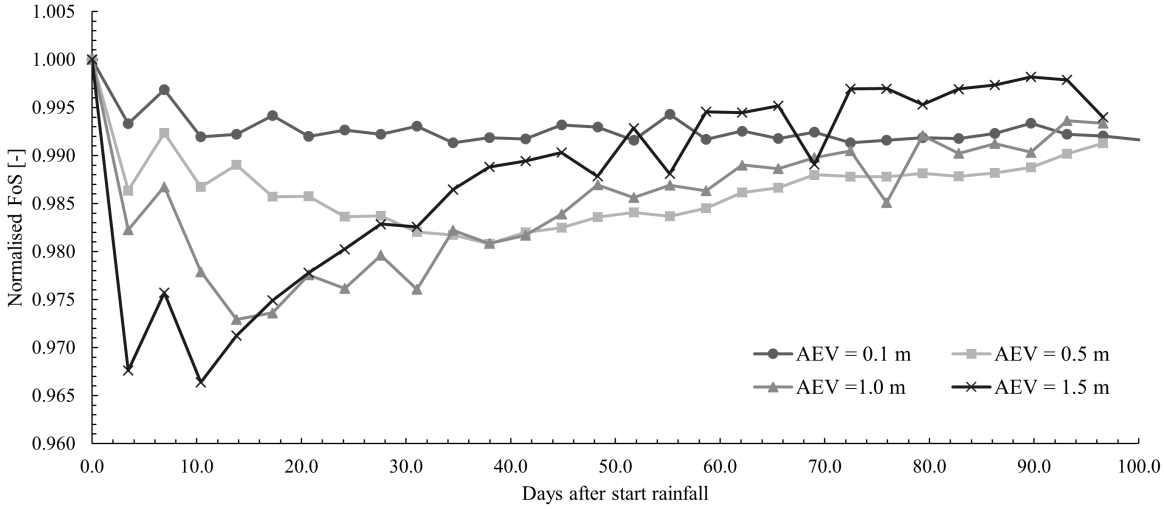


Figure 6.14: The change of FoS since the start of rainfall for various values of the air entry pressure.

k_{sat} will be of high influence when the tailings have a low hydraulic conductivity

The same analysis was performed for k_{sat} of which the simulation results are provided in Figure 6.15. For low hydraulic conductivity tailings, the influence of rainfall on the FoS is small with a reduction of the FoS upon infiltration of less than 1 %. Primarily, as a low hydraulic conductivity causes low infiltration rates and high run-off. Something that was also concluded by Rahimi et al. (2010) as provided in the literature study in Chapter 3. For higher values of the saturated hydraulic conductivity, the effect of rainfall on the FoS is more significant. In general, with increasing hydraulic conductivity, the minimum FoS becomes smaller with a small exception at hydraulic conductivity values between 0.1 and 0.15 m/day. A more common observation between the datasets is that with increasing hydraulic conductivity, the moment of the critical FoS is realised earlier after the start of rainfall. In addition, the trough of the low FoS is smaller. Both are caused by the increased wetting front speed for larger hydraulic conductivity, increasing recharge.

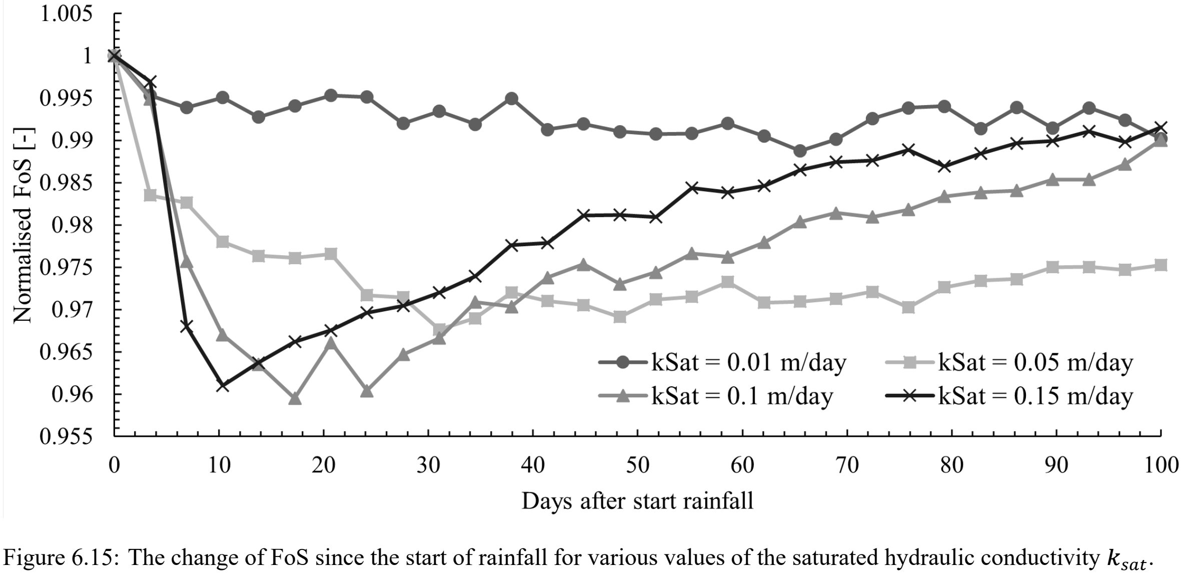


Figure 6.15: The change of FoS since the start of rainfall for various values of the saturated hydraulic conductivity k_{sat}.

6.2. Return periods analyses

In this section the analyses focusing on the return period of rainfall-induced static liquefaction will be treated. The various studies centre on the influence of the various strength parameters, the shell width and the geohydrological parameters. Furthermore some general comments will be made on the application of the return period of failure in analysis. The following analyses have been performed:

- **RP01:** Formulation of the IDF-curve and general comments on computed return periods and the contribution of the various durations of rainfall.
- **RP02:** stability charts of return periods of failure focusing on strength parameters.
- **RP03:** effect of shell width on the return period.
- **RP04:** return period based early warning system for the phreatic level before rainfall.
- **RP05:** influence of geohydrological parameters n_{mvg} and k_{sat} on the return period.

6.2.1. RP01

In this section, the IDF-curve will be presented as has been introduced in Chapter 4. Furthermore, this analysis will zoom in on the intermediate steps needed to perform the total return period calculation. To this end, two simulations were used that are part of analysis RP02. The parameters for these simulations as well as key findings are shown in Table 6.6.

It was shown that the Gumbel distributions provide with the most accurate estimate of the annual maxima of the rainfall dataset. The final IDF-curve showed equal qualitative behaviour to curves presented in literature. However, in comparison to the study of de Carvalho et al. (2014), which also considers precipitation data from the Minas Gerais state, slightly higher intensities are expected for the same return period of occurrence in that study. In the calculation of the return period of failure, it was shown that especially the longer duration events, longer than 3 days, contribute to the rainfall-induced static liquefaction of tailings dams. Additionally, the midpoint rule as explained in Chapter 4 showed higher return periods of failure than the lower limit, as was expected. Again the key findings are treated in the subsequent subsections.

Table 6.6
Analysis parameters of RP01

Goal	Formulation of the IDF- curve and determination of total return period
Geometric parameters	Height of 20 meters; slope of 3H:1V; shell width = 10 m
Strength parameters	Varying
Geohydrologic parameters	$\theta_s = 0.4$; $\theta_r = 0.05$; $k_{sat} = 0.1$ m/day; $n_{mvg} = 1.7$; $\alpha_{mvg} = 2.0$ 1/m; $AEV = 0.5$ m
Key findings	• The Gumbel distribution is most accurate for formulation of the IDF-curve. • The midpoint rule yields more conservative results than the lower limit. • There is no specific total rainfall depth which leads to failure. • Only the longer duration rainfall events contribute significantly to the return period of failure.

The Gumbel distribution is most accurate for formulation of the IDF-curve

For the IDF-curve, for each duration, either a Gumbel distribution or Generalised Extreme Value distribution was fitted to the dataset. For the Gumbel distribution, two parameter estimation methods were used: the method of moments and the maximum likelihood estimator. For the GEV distribution, only the maximum likelihood estimation was applied. Table 6.7 shows the results of these fits for each duration of rainfall. The Kolmogorov-Smirnov test value is given, as well as the two-sided p-value for accepting the null hypothesis. The null hypothesis being that the empirical distribution function is equal to the fitted distribution function.

Table 6.7
Results of the parametric distribution fits.

Length [days]	GEV MLE		Gumbel MM		Gumbel MLE	
	ks	p	ks	p	ks	p
1	0.018	0.658	0.021	0.487	0.015	0.833
2	0.022	0.407	0.017	0.729	0.020	0.556
3	0.023	0.377	0.018	0.673	0.020	0.526
4	0.020	0.524	0.023	0.374	0.016	0.819
5	0.018	0.640	0.021	0.463	0.015	0.880

Following these results, the GEV distribution does provide a good fit but less than the Gumbel distribution. Having an additional degree of freedom with the shape parameter, the MLE of the GEV is less able to obtain correct estimates. The scale parameter to which the ML estimator has converged is equal to 0.0099 which means that it converges to a Gumbel distribution as the shape parameter is nearly zero. Concerning the Gumbel distribution fits, alternatively the method of moments and the ML estimator perform best. The MLE performs best on the duration of 1, 4 and 5 days of rainfall. The method of moments shows the best fit for 2 and 3 days of rainfall. The Kolmogorov-Smirnov test outcomes are satisfactory with only a minimal distance between the empirical and fitted distribution function. The corresponding p-value is statistically significant. The final parameters used for the Gumbel distributions are shown in Table 6.8. The final IDF curve based on the return period of each intensity and duration is shown in Figure 6.16. It shows the same qualitative behaviour as e.g. in Tfwala et al. (2017), Bhatt et al. (2014) and Koutsoyiannis et al. (1998).

Duration	u	α
1	0.07476	0.02285
2	0.05325	0.01597
3	0.04180	0.01256
4	0.03524	0.01108
5	0.03105	0.00980

Table 6.8
The fitted location and scale parameters for the Gumbel distribution for each duration.

Comparing to the results of de Carvalho et al. (2014), in which GEV distributions are fitted to Brazilian precipitation data, the same qualitative behaviour of the IDF-curve is shown in comparison to this study. With respect to the quantitative behaviour, in Figure 6.16 higher return periods are found for the same intensity. This could mean that the found return periods of failure in this study are non-conservative. Any difference in values found may lie with the fact that different datasets are used and that the final distribution applied is different. Also the goodness-of-fit test results in this study are better. The Kolmogorov-Smirnov distance in that study is equal to 0.066 in contrast to the maximum 0.018 in this analysis.

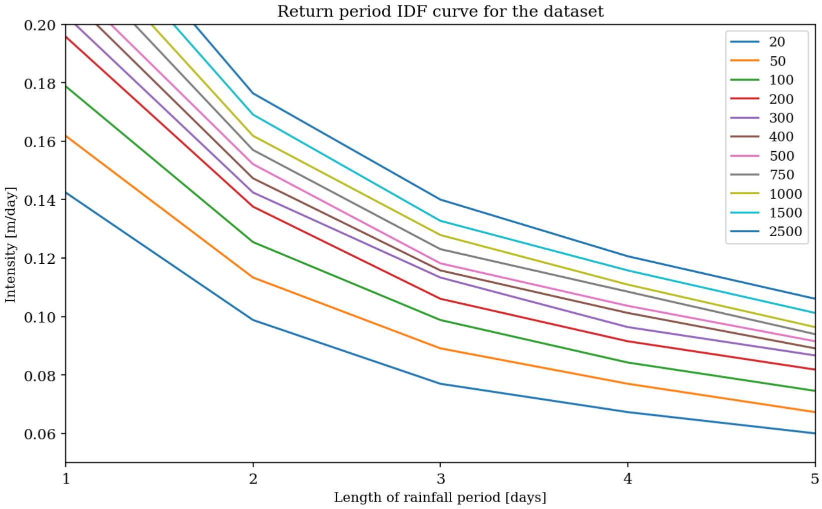


Figure 6.16: The Intensity-Duration-Frequency curve as found, applying the Gumbel distribution and its corresponding parameters to each of the durations. The contour lines represent the return period of occurrence of each combination of intensity and duration.

Table 6.9
Intensities and probability of failure for strength combination 0.37-36-32.

Duration [days]	Intensity [m/day]	P_f lower limit	P_f midpoint rule
1	0.490	1.37E-9	5.45E-6
2	0.200	6.91E-6	7.35E-5
3	0.119	9.18E-5	4.66E-5
4	0.075	9.41E-6	5.02E-4
5	0.065	3.77E-3	3.77E-3
Total P_f	-	3.87E-3	4.39E-3
Return period [a]	-	258	227

The midpoint rule yields more conservative results than the lower limit
To illustrate the steps needed to get from the failure intensity of each duration to the final return period of rainfall-induced static liquefaction, two simulations from analysis RP02 are treated in detail here. The first simulation has the following combination of strength parameters: 0.37-36-32. Where the order corresponds to the used: peak strength ratio - friction angle of the liquefiable tailings - friction angle of the non-liquefiable tailings. Shown in Table 6.9 for each duration of rainfall is: the needed intensity to obtain an FoS of 1.1; the probability of failure for that duration considering the lower limit estimate; and the probability of failure for that duration according to the midpoint rule. Then in the last two rows of the table the total probability of failure as well as the return period of failure are provided for both the lower limit and midpoint rule approach. Table 6.10 shows the same for a strength combination of 0.37-37-31.

The midpoint rule that was applied results in a lower return period of failure (i.e. higher probability of failure). The other method based on the simple lower bound yields non-conservative estimates. From hereon, the reported return periods are accordingly all based on the midpoint rule.

There is no specific total rainfall depth that leads to failure
In earlier analysis it was observed that the failure intensity multiplied with the duration of rainfall will equate to the same rainfall depth over all durations (see Appendix B.3). This is not true for the simulations shown in Tables 6.9 and 6.10. For the given two strength combinations, the total rainfall depth is the same for rainfall durations of 3 to 5 days. Therefore, for longer duration rainfall events there is a

Table 6.10
Intensities and probability of failure for strength combination 0.37-37-31.

Duration [days]	Intensity [m/day]	P_f lower bound	P_f midpoint rule
1	0.394	9.14E-8	3.43E-5
2	0.171	4.34E-5	2.87E-4
3	0.103	3.48E-4	7.10E-4
4	0.076	7.51E-4	1.28E-3
5	0.059	6.98E-3	6.89E-3
Total P_f	-	8.03E-3	9.21E-3
Return period [a]	-	125	109

linearity in the needed intensity to failure with increasing duration. However, for the durations of 1 and 2 days of rainfall more precipitation is needed. This is likely caused by expected run-off for these high intensities. Referring back to analysis GA03, it was noted that for intensities larger than 0.2 m/day, the effect of additional intensity is reduced considerably.

Only the longer duration rainfall events contribute significantly to the return period of failure

It is shown in the same tables that the probability of failure due to a short rainfall event of 1 or 2 days is many orders of magnitude smaller than for 3, 4 or 5 days of rainfall. The boundary of 3 days for a significant contribution to the return period of failure was also found in Jasim et al. (2017). It is however unlikely that this observation is omnipresent and the exact duration from which the probability of failure starts increasing significantly will be dependant on the situation studied. It however shows that tailings dams and other embankments are more vulnerable to longer duration rainfall events. The increased effect of these longer events is stimulated by the convex shape of the IDF-curve in which the likelihood of large intensity rainfall will tend to flatten for longer duration storms. Given the linearity of needed intensity with increasing duration, the probability of occurrence of that intensity does not decrease at the same rate due to flattening of the IDF-contours. The likelihood of the scenario therefore increases. Because of the increased effect of longer duration rainfall it was decided to perform the analysis for 3 to 5 days of rainfall only, omitting any contribution to the return period of failure by shorter duration rainfalls.

6.2.2. RP02

Analysis RP02 was conducted to investigate the sensitivity of the return period of failure to a set of varying strength parameters. At first, the effect of varying combinations of friction angles for the unsaturated liquefiable tailings and the non-liquefiable tailings was studied. Here, the shell width was set at 10 meters and the peak strength ratio at 0.37. Later the friction angles were fixed and the influence of the peak strength ratio was analysed. The effect of a varying shell width is described in analysis RP03. The parameters used in and key findings from analysis RP02 are shown in Table 6.2.

It was shown that, because of the little influence of rainfall on the FoS, the return period of failure increases sharply with increasing strength. Rainfall is therefore only a potential failure trigger when the slope is marginally stable. Something that was also argued by Ng and Shi (1998) when studying regular slopes. Some boundary is shown for which the return period of failure significantly increases. The shape and character of this boundary is dependant on the combination of the strength parameters and is attributed to how these parameters cause the failure surface to change shape and location. The peak strength ratio has a large effect on the stability of the dam and shifts the boundary of marginal stability.

Table 6.11
Analysis parameters of RP02

Goal	Estimating the effect of the strength parameters on the return period of failure.
Geometric parameters	Height of 20 meters; slope of 3H:1V; shell width = 10 m
Strength parameters	Varying
Geohydrologic parameters	$\theta_s = 0.4$; $\theta_r = 0.05$; $k_{sat} = 0.1$ m/day; $n_{mvg} = 1.7$; $\alpha_{mvg} = 2.0$ 1/m; $AEV = 0.5$ m
Key findings	• The return period of failure is significantly influenced by the drained friction angles of the tailings and the shell. • Effect of a higher intensity of rainfall to the FoS decreases the effect of strength on the return period. • The peak strength ratio shifts the boundary of marginal stability.

The return period of failure is significantly influenced by the drained friction angles of the tailings and the shell

A sensitivity analysis of the return period of failure was performed for 20 combinations of the drained friction angles, of the non-liquefiable shell and the liquefiable unsaturated tailings, according to the ranges as defined in chapter 5. No simulations were performed for which the friction angle choices resulted in return periods far larger than 1000 years because of the sensitivity of the result to the Gumbel distribution fit. The results of the various combinations were processed to form a contour plot of the return periods over the considered strength range. This 'stability chart' is visible in Figure 6.17. The x-axis ranges between 30 and 35 degrees and resembles the angle of internal friction of the shell. The y-axis ranges between 30 and 40 degrees and is the drained friction angle of the tailings. All combinations have the same peak strength ratio of 0.37. The 'x' marks represent simulations. The contour lines are interpolated values based on a linear spline method. Two of the results incorporated in this plot have been treated in analysis RP01, showing how the final return period is calculated.

The white space with a return period smaller than zero resembles an FoS lower than the boundary of 1.1 for no rain imposed, or was not examined. Then with increasing strength along an increasing diagonal in the contour plot, there is an increasing return period of failure. Over the entire solution domain, the contours are approximately spaced equally. The contours show largely linear behaviour over the solution domain with an angle greater than 1:1, indicating that the shell tailings friction angle is more important compared to the friction angle of the tailings. Figure 6.18 shows that this is a result the fact that a larger part of the failure surface runs through the drained shell.

When closely observing the contours in Figure 6.17, the behaviour of these is not fully linear but slightly convex towards higher angles of internal friction of the shell. This is primarily caused by the relatively growing importance of the unsaturated liquefiable tailings at high angles of friction of the shell. When the angle of friction for the shells increases relative to that of the tailings, the path through those tailings will be maximised. This can also be seen in Figure 6.19 where the shell holds a higher strength, showing an almost circular failure surface. When the path through the tailings increases, the FoS is more sensitive to the friction angle of the tailings. Hence, the slope of the contours will flatten.

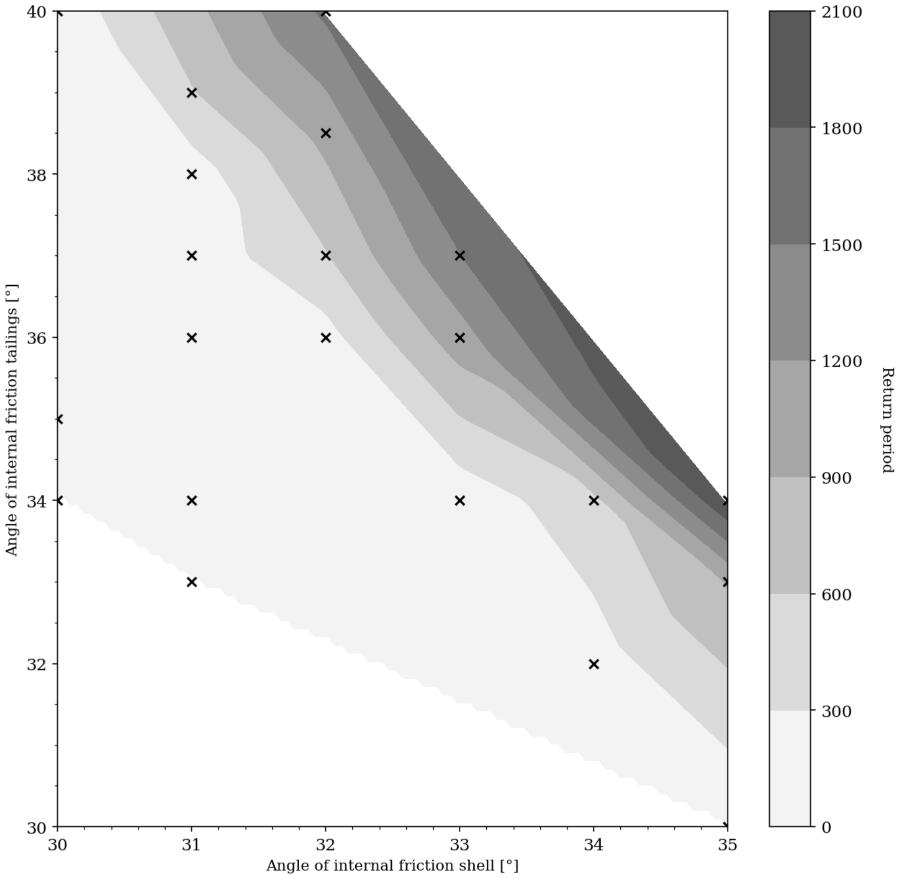


Figure 6.17: The contour plot of the return periods for combinations of the drained friction angles considering a shell width of 10 meters and a fixed peak strength ratio of 0.37. Simulations are marked by an ’x’, the contour lines are linearly interpolated values.

Effect of a higher intensity of rainfall to the FoS decreases the effect of strength on the return period

More information can be extracted from Figure 6.17 by examining the return period along verticals or horizontals. To elucidate on the behaviour of the return period along such a line of constant friction angle, the quantitative return period values were extracted for a friction angle value of 34° for the tailings. Figure 6.20 shows the return period along this horizontal and also includes the needed intensity for failure for a duration of 5 days. The values for both are displayed on separate axes. The duration of 5 days is most significant to the return period of failure as has also been shown in analysis RP01 and has therefore been chosen in this plot.

The trend can be characterised as bi-linear with a clear kink at a friction angle of 33 degrees. To some degree this may have been caused by a less proper estimate of the needed intensity for a duration of 5 days at a friction angle of 31 degrees. The return period is shown on a log-scale and shows the same trend as the needed intensity, although the kink is less visible and the curve is smoother. The effect of a poor estimate is less visible in the return period as the effect is reduced by the probabilities of failure for the other durations of rainfall. It is however shown that the found return period is a mere translation of the needed intensity. The fact that the return period needs to be displayed on a log-scale to show the same behaviour may be attributed to the exponential nature of the Gumbel distribution which is underlying these return period calculations. It also explains why the return period explodes for higher strengths in contour plots such as that of Figure 6.17.

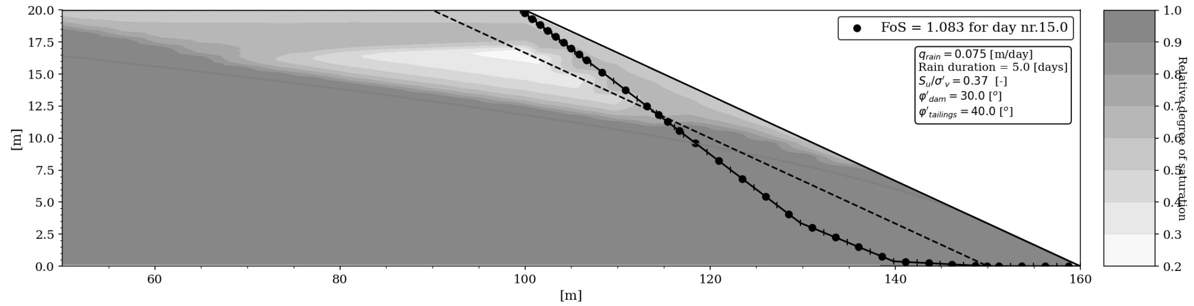


Figure 6.18: The failure surface for an impoundment with a high friction angle of the shell and a low friction angle for the unsaturated liquefiable tailings.

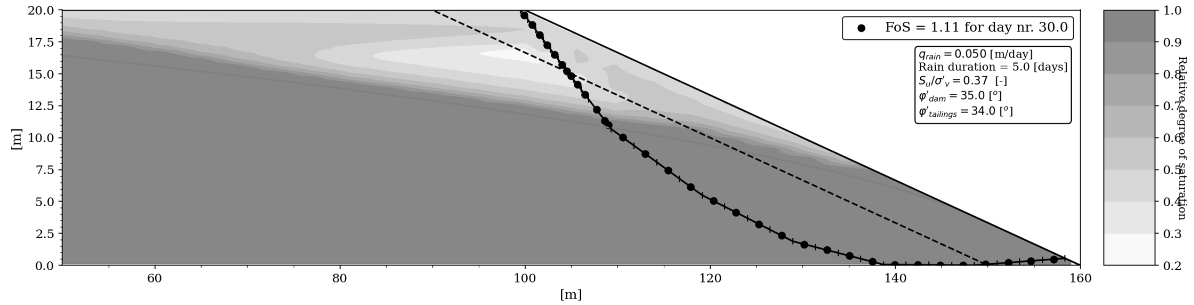


Figure 6.19: The failure surface for an impoundment with a high friction angle of the unsaturated liquefiable tailings and a low friction angle for the shell.

In comparison, an identical study has been performed along a vertical for a constant friction angle of 32 degrees for the shell. This result is visible in Figure 6.21. Likewise, the behaviour of the return period follows that of the needed intensity to failure and shows a bi-linear relationship. However, the shape is concave as opposed to the convex behaviour shown in Figure 6.20. In light of the results given in analysis GA03, at intensities around 0.06-0.07 m/day, recharge to the water table will start to occur. In this manner, the effect of rainfall is increased. Therefore, the curves tend to level off. It is unsure why this behaviour is not shown in Figure 6.20. Here, increasingly higher intensities are needed to get to failure. One possible explanation is that for higher values of the drained friction angle of the shell the failure surface will tend to move upstream in the impoundment to avoid this stronger zone. Therefore, any recharge effects will be less evident as the mixed mechanism might not be able to occur.

The peak strength ratio shifts the boundary of marginal stability

The return periods of failure were also analysed for peak strength ratios of 0.36 and 0.38. The value of 0.38 is higher than indicated in the sensitivity analysis range in Chapter 5 and is applied here for demonstration purposes only.

The shift of the peak strength ratio had a significant impact on the return period results. The highest possible strength combination for a strength ratio of 0.36 would be 0.36-40-35, following the parameter ranges as defined in Chapter 5. The resulting return period for that analysis was equal to 188 years. Surrounding combinations of 0.36-40-34 and 0.36-39-35 would yield 54 and 64 years respectively. Alternatively, for the higher peak strength of 0.38 reasonable return periods were obtained only in the lower strength spectrum. The minimal return period at a combination of 0.38-30-30 was 48 years. A comparison of the results can be seen in Figure 6.22, showing the 150-year return period contour for the different peak strength ratios. It may be observed that the peak strength ratio affects the return period of failure by a large extent. Essentially a different estimate of the peak strength ratio may render a slope completely safe or completely unsafe when looking at rainfall.

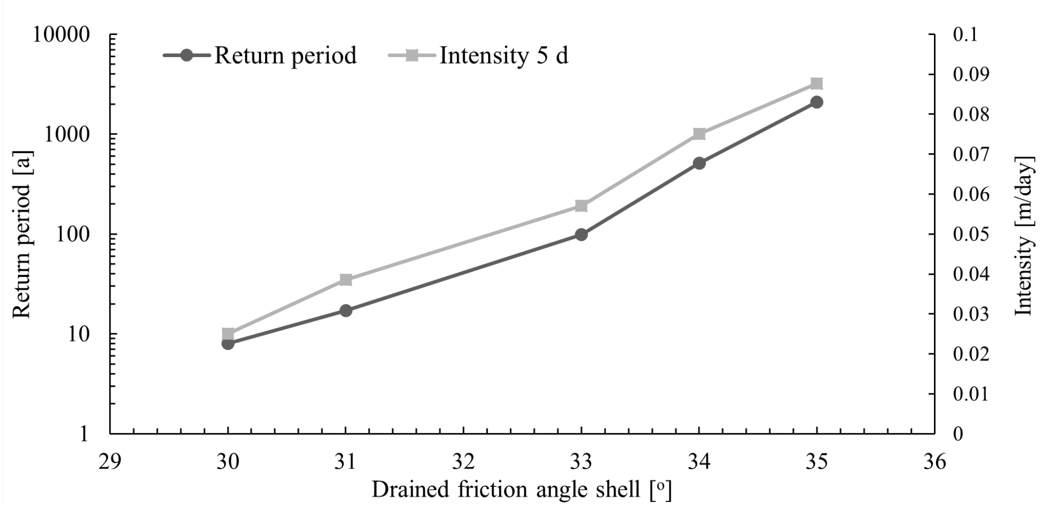


Figure 6.20: The total return period of failure and needed intensity to failure for a duration of 5 days for a constant friction angle of the tailings of 34 °.

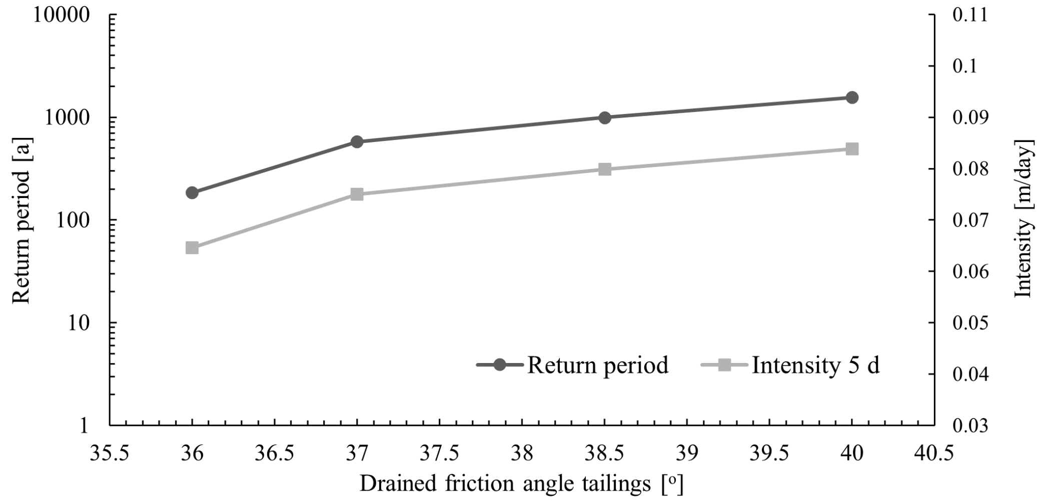


Figure 6.21: The total return period of failure and needed intensity to failure for a duration of 5 days for a constant friction angle of the shell of 32 °.

6.2.3. RP03

Analysis RP03 elaborates on the influence of the shell width on the return period. Analysis RP02 has had a set width of 10 meters and focused on the application of different strengths. A similar simulation will be performed here. A shell width of 20 meters will now be considered for varying combinations of the friction angles of the non-liquefiable and the unsaturated liquefiable tailings. The shell width of 20 meters was chosen based upon the potentially related results of analysis GA01 where, for this shell width, the influence of rainfall on the FoS was significantly decreased. All other parameters are the same as in analysis RP02. The goal and key findings of analysis RP03 are shown in Table 6.12.

In total, 6 simulations were performed. The final results of the simulations series for the contour plot on the return period can be seen in Figure 6.23. It was concluded that also with regards to the return period of failure the shell width is of large influence. As the effect of rainfall is minimised for a wide shell, the return period sharply increases with increasing strength. A very narrow boundary is obtained between dams which are not stable even before rainfall, and dams with return periods of 1000 years.

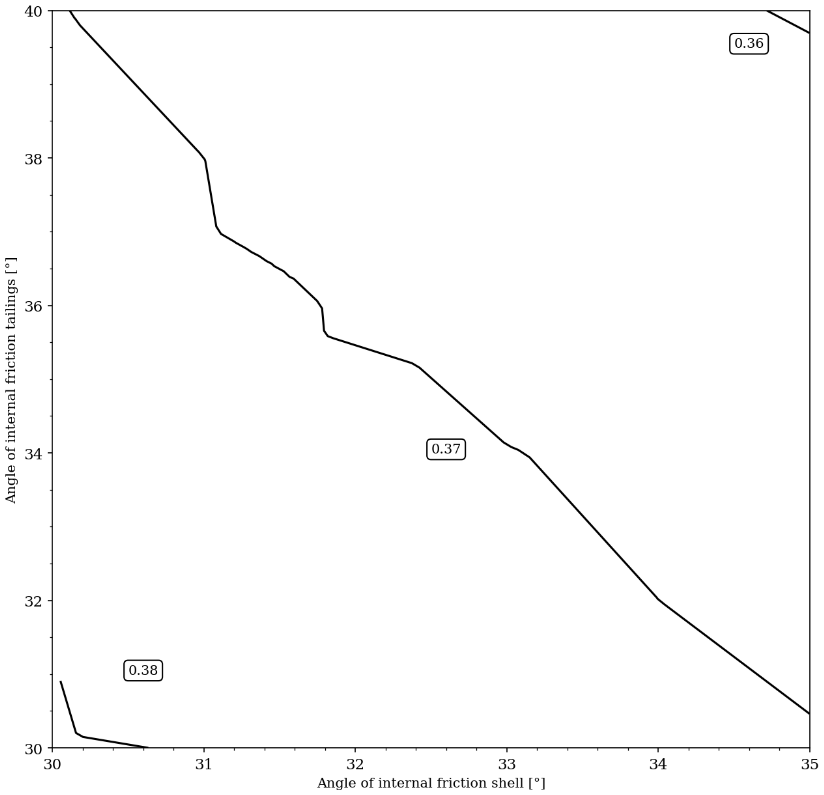


Figure 6.22: A comparison of the 150 year return period contour for different values of the peak strength.

Table 6.12
Analysis parameters of RP03

Goal	Estimating the effect of the shell width on the return period of failure.
Key findings	• No influence of the friction angle of the tailings for large shell widths. • Rise of the phreatic surface governing in return period analyses.

No influence of the friction angle of the tailings

There is some clear effect of the shell width on the slope of the contours. The resulting return periods of the simulations are now identical along a vertical, i.e. constant friction angle of the shell. This indicates that there is no influence of the friction angle of the tailings. In light of the results of analysis GA01, this behaviour could be expected, as the critical failure surface for this shell width does no longer run through the unsaturated tailings.

Rise of the phreatic surface governing in return period analyses

Shown in Figure 6.23 is that, by increasing the shell width the effect of the friction angle of the shell on the return period has significantly increased. Effectively, the return period changes from a value between 0 and 150 years to around 1000 years within 0.5 degrees of the friction angle. For this shell width, the shell suppresses that the slope is weakened by a rise of the phreatic surface. The lower FoS is in this case only resulting from a loss of suction. This reduces the effect of rainfall on the return period and hence the contours are smaller spaced. It is however shown that the loss of suction only causes a minimal effect on the return period of failure as otherwise the found boundary would be less narrow.

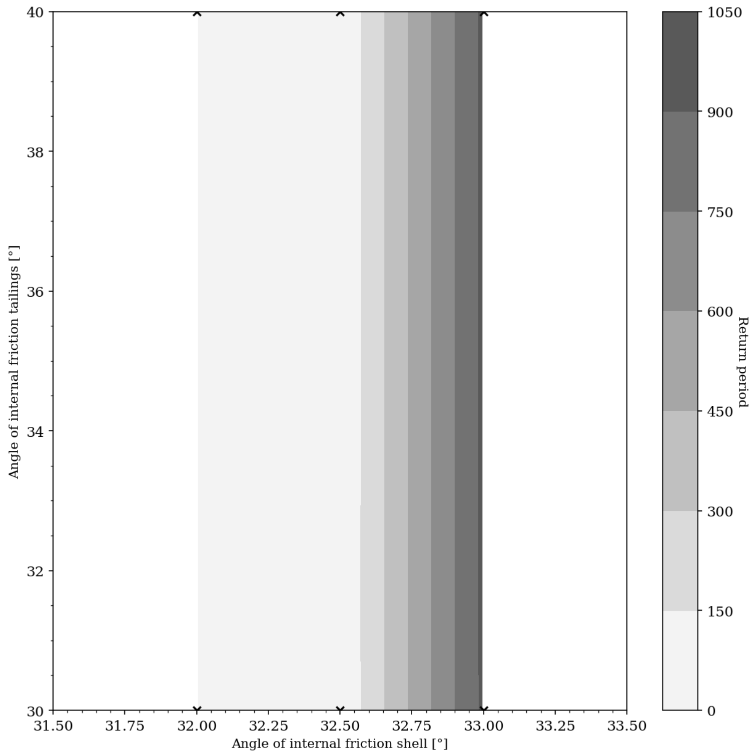


Figure 6.23: The return period contours over a range of friction angles for a small dam of 20 meters with a shell width of 20 meters. The peak strength ratio is fixed to a value of 0.37.

This is contrary to what was found in analysis GA03 as here it was shown that the influence of loss of suction on the FoS is roughly equal to that of mounding of the water table. It also contradicts the hypothesis of the Feijão dam expert panel that a loss of suction only could have contributed to instability of the dam. It is shown in this and earlier analyses that this failure hypothesis is unlikely and the dam was probably only marginally stable. Any other trigger would have likely had the same or an even larger effect.

6.2.4. RP04

Only marginally stable dams are prone to collapse upon rainfall infiltration. The strength as well as the shell width have been considered in this. There is one other parameter which is of importance in establishing whether an impoundment is prone to rainfall-induced failure. This parameter is the chosen boundary condition for the head at the far upstream side of the impoundment. It defines the initial margin to instability and also influences the effect of rainfall on the slope as it alters the length of the path that the wetting front needs to pass to the phreatic surface. The choice of the initial phreatic surface is therefore likely a governing parameter in the results. In order to exploit this hypothesis, a series of analyses was performed in which the procedure was inverted. Rather than taking a fixed boundary condition of the head and considering a range of possible peak strength ratios. For each strength ratio, the boundary condition was found for which the return period of failure equalled roughly 1000 years. Table 6.13 holds the parameters used. There has been no need to document on the basis of key findings here.

The phreatic level for which the return period is roughly 1000 years for a varying strength ratio can be seen in Figure 6.24. The return period had to be approximated based on linear interpolation between various simulations. The relationship between peak strength ratio and the boundary condition is approximately linear.

Table 6.13
Analysis parameters of RP04

Goal	Sensitivity of the return period towards the phreatic surface issued as the initial and boundary condition.
Geometric parameters	Height of 20 meters; slope of 3H:1V; shell width = 10 m
Strength parameters	$\varphi'_{liquefiable} = 35°$; $\varphi'_{non-liquefiable} = 32°$; s_u/σ_v = varying
Geohydrologic parameters	$\theta_s = 0.4$; $\theta_r = 0.05$; $n_{mvg} = 1.7$; $\alpha_{mvg} = 2.0$ 1/m; $AEV = 0.5m$; and $k_{sat} = 0.1 m/day$

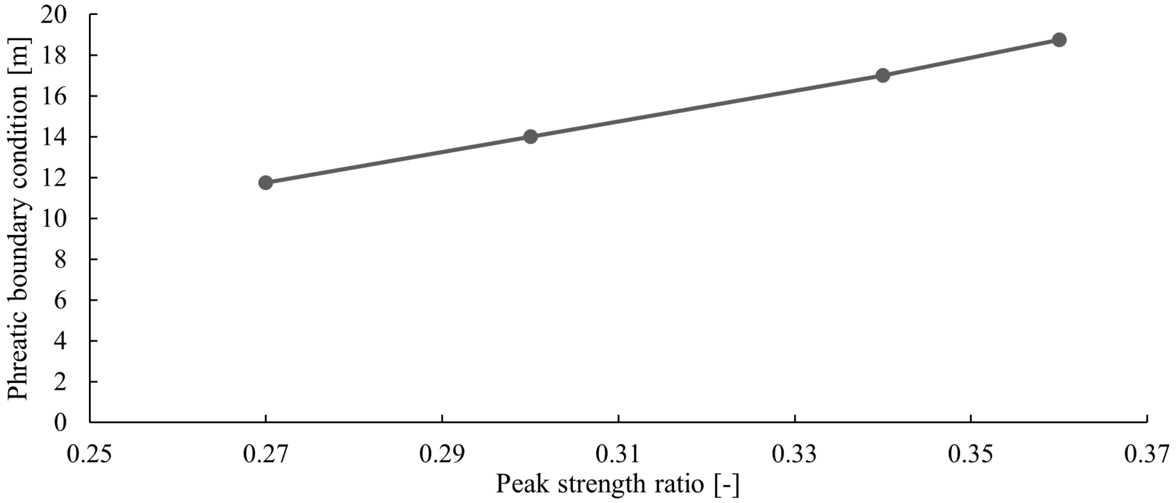


Figure 6.24: The boundary condition of the phreatic surface for which, for different values of the peak strength ratio, the return period of failure is approximately 1000 years.

For these levels of the phreatic surface, except for that of the strength ratio of 0.36, failure is solely attributed to a loss of suction. The phreatic surface is situated at too large depths with respect to the surface level in order to have recharge to the water table. Also the influence of rainfall is lower, similarly to how the height has affected this influence as was shown in analysis GA04. For lower strengths (and associated lower boundary conditions), the early warning system is not effective. The effect of rainfall is only marginal and the FoS for such dams without rainfall is only in the order of 1.11. The difference between a phreatic level that results in a return period of 1000 years and the level for which the FoS without rainfall is equal to 1.1 is around 0.1 meters in this case. There is a wide variety of other loads that may trigger instability when the dam is only marginally stable. Furthermore, any variability in the deposit may be of a larger effect.

For the higher strengths, where rainfall has a larger influence due to the higher position of the phreatic surface, the difference between the shown phreatic level and the level at which the FoS is equal to 1.1 already is in the order of 0.5 meters. It is largely unknown what the variation of phreatic levels inside a tailings dam is, but for such values (around 2.5 % of the height), the proposed method could serve as an early warning system. However, the engineering approach will likely be to determine that groundwater level for which the FoS is equal to 1.1 and issue a warning when the phreatic surface is 0.5 meters underneath that level. Although simple, this approach is already favourable for the safety, considering some of the effects that rainfall may have on the stability of the impoundment.

6.2.5. RP05

Analysis RP05 is an extension of the earlier GA05 analysis and considers the following geohydrological parameters: n_{mvg}, α_{mvg} and k_{sat}. The AEV will not be applied in a return period analysis. The goal is to obtain an idea of the sensitivity of the return period of failure to each of these parameters. The used parameters are shown in Table 6.14.

It was shown that only the n_{mvg} and k_{sat} influence the return period of failure. The return period increases for decreasing values of n_{mvg} whilst this is the other way around for k_{sat}. α_{mvg} did not affect the return period.

Table 6.14

Analysis parameters of RP05 for the parameter n_{mvg}

Geometric parameters	Height of 20 meters; slope of 3H:1V; shell width = 15 m
Strength parameters	$\varphi'_{liquefiable} = 37°$; $\varphi'_{non-liquefiable} = 32°$; $s_u/\sigma_v = 0.37'$
Geohydrologic parameters	$\theta_s = 0.4$; $\theta_r = 0.05$; $k_{sat} = 0.1$ m/day; $\alpha_{mvg} = 2.0$ 1/m; $AEV = 0.5$ m

The significance of n_{mvg} value was found to be especially high near the low parameter values around 1.1 to 1.3. Figure 6.25 shows the resulting return periods of failure for some values of n_{mvg}. For the lowest value of the indicated range that this parameter holds, the return value is exceptionally low. This is not because the initial FoS is low, but is fully attributed to a larger sensitivity of the FoS to rainfall. This sensitivity has also been noticed in analysis GA05 where for lower values of n_{mvg}, the wetting front speed is higher and the possible reduction of the suction is larger. The return period roughly linearly increases on a log-scale. This indicates that the needed intensity will increase linearly with n_{mvg}. Deviations from this linear behaviour are most likely caused by the non-linear effect of the wetting front as well as the change in the initial suction profile.

The return period of failure with varying k_{sat} was studied for a strength combination of 0.36-37-34. For a value of 0.01 m/day, the return period was essentially infinite as the influence of rainfall was so small that the FoS for 5 days of rainfall at an intensity of 0.15 m/day was still 1.102. This in contrast to an FoS without rainfall of 1.104. However, analyses done with hydraulic conductivity values of 0.05 and 0.1 m/day immediately yielded low return periods of only 12 years. This shows that the saturated hydraulic conductivity does have a large effect on the return period, but only when it has low values.

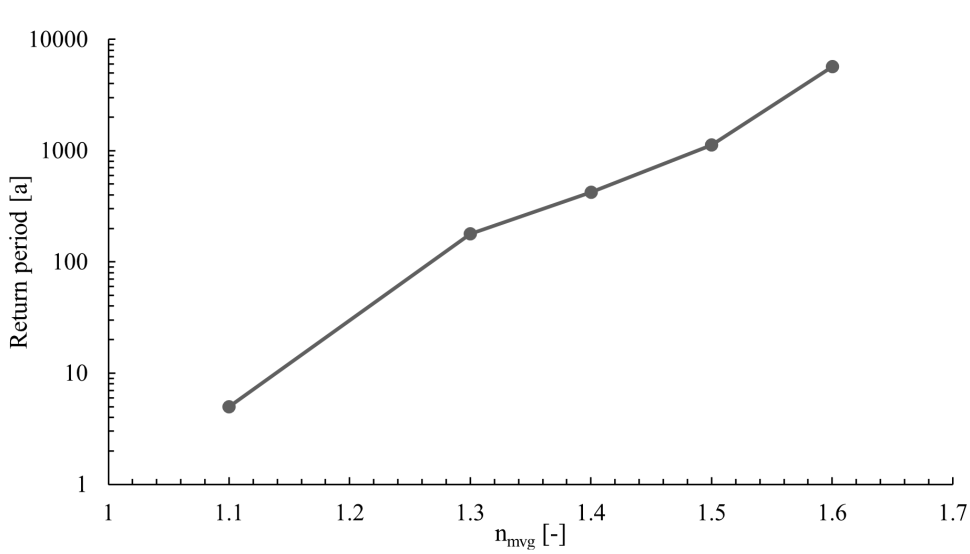


Figure 6.25: Return period of failure for different values of the SWCC shape parameter n_{mvg}.

CHAPTER 7

Conclusion

Sparked by recent failures of large tailings dams in which rainfall has played a significant role, a study was started towards rainfall-induced static liquefaction. Given the lack of a fundamental risk assessment framework for these structures, emphasis was put on the extreme value analysis of rainfall. Therefore, this thesis has aimed to answer the following research question:

”How does the probability of static liquefaction of a tailings dam relate to extreme rainfall events?”

In this conclusion, an answer will be provided based on the same structured approach as used in the results, treating first the general effect that rainfall has on the stability of a tailings dam and later concluding on how this translates to the return period of failure.

7.1. General effect of rainfall infiltration on static liquefaction of a tailings dam

It was observed that tailings dams can liquefy upon rainfall infiltration, which confirms observations that have been done during past failures of these storage facilities. Failure is attributed in this study to a ‘mixed mechanism’. The FoS is lowest when there is a rise of the phreatic surface, increasing effective stress and lowerings strength, as well as a reduction of suction due to the wetting front passing along the failure surface. The fact that the finer tailings with high saturation will liquefy upon shearing has had some significant impact on the shape of the failure surface as it caused the tendency for deep slip surfaces to form. Such slip surface maximise their path through liquefiable tailings. The path through the non-liquefiable shell as well as the unsaturated tailings is minimised. As a result, the effect of rainfall is delayed with the lowest factor of safety occurring roughly after 20 days. Furthermore, as there is the tendency to create these deep failure mechanisms, the effect of rainfall is low. The infiltration of water influences only a very minimal amount of the slip surface. There was also little movement of the governing slip surface due to the lower strength of the impoundment below the phreatic surface. The FoS was in some simulations therefore lowered by only 3%, depending on the various parameters. Only marginally stable dams are in this way prone to collapse because of extreme rainfall events.

The influence of rainfall was further reduced when studying higher dams. Although these dams had nearly similar factors of safety, the influence of rainfall was strongly reduced because of the additional distance the wetting front needs to travel for these dams. The SWCC plays an important role here in significantly reducing the permeability of the tailings for more negative pressure heads. Additionally, the effect of the geohydrological parameters on the FoS is only minimal. The Mualem-van Genuchten parameter n_{mvg} did have an effect but for low values only. For higher values of this parameter the influence diminished. The saturated hydraulic conductivity only influenced results when this parameter had very low values. Infiltration was then minimised.

7.2. Sensitivity study to the return period of rainfall-induced static liquefaction

The return period of rainfall-induced static liquefaction was calculated using an integration method for the intensity-duration-frequency curves. A midpoint rule, partially based on the subset rules within probability theory, allows for a conservative estimation of the return period. Reflected in the resulting return periods is the extent to which rainfall is able to influence the FoS. Only for marginally stable dams, reasonable return periods were found. In this way, the influence of strength on the return period was considerable. A small increase of strength can significantly increase the return period of failure. It was shown that by increasing the strength, the needed intensity to failure will approximately linearly increase as well. Given the Gumbel distribution to translate needed intensity to the return period of failure, the resulting return period however shows an exponential behaviour. In this fashion, the return period will increase from around 100 years to over a 1000 years within only 2 degrees increase for the friction angle of the non-liquefiable shell. The effect of the peak strength ratio is even larger given the fact that a considerable portion of the failure surface runs through the liquefiable tailings. The width of the non-liquefiable tailings shell is important in that sense that it is able to suppress possible shifts of the failure surface. When the shell width exceeds a certain value, the failure surface is no longer able to 'profit' from a mixed mechanism of loss of suction and rise of the phreatic surface. This results in an even lower influence of rainfall and hence a larger influence of the strength parameters (especially the friction angle of the shell) onto the return period. Essentially, a discrete boundary is formed between a non-stable and a very stable tailings dam when considering rainfall-induced static liquefaction as the failure mechanism.

The geohydrological parameters were only of limited influence. Akin to the observations in the general analyses, only n_{mvg} showed to impose a large effect for especially low values of this parameter. It influences the wetting front speed and results in a significant loss of strength due to mounding of the water table.

Of large effect as well is the initial phreatic surface that was used in the model. It influences the degree of saturation and length of the path that the wetting front needs to pass in order to reach the groundwater table. It will therefore affect the earlier mentioned mixed mechanism. In view of these observations, the calculation procedure has also been applied in an inverse manner. Based on deterministic parameters for the dam, a phreatic surface was defined for which the return period of rainfall-induced static liquefaction is around 1000 years. It was shown that if the phreatic surface rises above that, the return period is quickly reduced. The given phreatic level can be used as a early warning system to impose additional mitigation measures to reduce the risk. It is however noted that the dam is only marginal stable when this warning system becomes effective. Other loads will therefore need to be considered as well.

CHAPTER 8

Recommendations

This chapter reflects on the methodology, results and conclusions and will propose recommendations on various scale levels. These scale levels are:

- recommendations for new studies with the applied methodology in this thesis;
- recommendations for the study towards rainfall-induced static liquefaction (of tailings dams);
- recommendations for the general analysis of failure of tailings dams.

8.1. Recommendations to the applied methodology

Recommendations will be proposed for two parts within the total analysis methodology. These parts are: the calculation of the single performance indicator for the rainfall spectrum and the code for integrated solving of the unsaturated flow problem and factor of safety.

8.1.1. Recommendations to the performance indicator

Provided is a procedure to obtain, from a series of data points within the IDF-curve, the total return period of failure. This has been of value as shown in the results and conclusion. The implication of an undrained failure mechanism has however been severe in this analysis, minimising the influence of rainfall on the overall FoS. It is therefore of interest to study the return period of failure of natural slopes or in a normal effective stress analysis. Literature research has shown that the effect of rainfall may be much greater for these situations and the methodology could potentially be applied in societal risk studies involving landslide hazards of natural slopes.

Furthermore, although extreme value analysis has been applied to this calculation, the applied calculation procedure is still deterministic. Deterministic parameter values are applied and uncertainties on the load are not considered. It is recommended to apply the procedure to a full probabilistic study, e.g. with a Monte Carlo method. These results might be valuable as they allow for consideration of variability over the deposit. In addition it was argued that improved fitting methods were needed for the fits of the Gumbel and GEV distributions to the dataset.

8.1.2. Recommendations to the unsaturated flow and FoS solver

The code of PySWAP and the slope stability algorithm have been a major component of this thesis. The code is however not flawless and there are some considerable advantages to be gained. There is a long list of possible improvements such as but not limited to:

- implementation of variable unit weight above the phreatic surface;
- better distribution of the slice width;
- have PySWAP return values for specific events rather than that calculation times may be different for the various columns;
- use of implicit hydraulic conductivity updating in PySWAP

- application of Python libraries for cubic spline adaption of the Mualem-van Genuchten equation
- better coding structure of the main code.

A large issue has also been the computational performance of the code. Python is a relatively slow programming language in comparison to e.g. C or FORTRAN. It is not recommended to translate the programme again to another programming language. Python is a language that is quickly gaining popularity and is relatively easy to learn. It is therefore of great value to potential users that the code has been implemented in Python. Further research should however strive for a more computationally efficient code, e.g. by parallel programming of the slope stability calculation or use of extensive Python libraries such as Cython.

Most complicated was however the accuracy/performance of the differential evolution algorithm. Application of this method ensures that only a minimal amount of computations is needed. In this study, 5 or 6 global slices/degrees of freedom were implemented. It is advised in literature to have a population size of 5 to 10 times the degrees of freedom. This was not possible within a reasonable computational time. This meant that in terms of population size and convergence criteria, some compromise had to be found. This compromise was sensitive to the situation examined and the calculated FoS could differ per simulation with a value of around 0.003. In absolute terms this is small and an FoS calculator with this accuracy satisfies normal use. Since the decrease in FoS is only minimal given rainfall, very accurate estimates are however needed in the calculation of the needed intensity to obtain the failure FoS. It is therefore recommended to opt for a more accurate and efficient evolutionary algorithm.

8.2. Recommendations for research towards rainfall-induced static liquefaction

This study has focused on the failure mechanism of static liquefaction, triggered by rainfall infiltration, and used the IDF-curves as a basis for return period calculations. The research is therefore limited to those durations for which the IDF-curve is relevant. In this case, the lower bound of the duration was the sampling rate of the data which was 1 day. The upper bound was a period of 5 days of consecutive rainfall. It has been proven that tailings dams are insensitive to these storm conditions, and that only marginally stable dams are prone to failure. The influence of longer duration rainfall is likely higher as it possibly influences the phreatic surface on the longer term. This phreatic surface has proven to be essential in this study. Following this reasoning, it will be of interest to simulate the influence of rainfall for much longer periods. Literature has provided with case studies of rainfall induced failures caused by extremely wet seasons. Although failure due to very intense rainfall of only a few days has been recorded as well, it is of interest to see the effects of long term low intensity rainfall. To avoid ill-conditioning of the results it is best to simulate rainfall and evaporation for long periods of e.g. 1000 years. Methodologies such as the Monte Carlo Markov Chain methods are therefore to be applied. The methodology as proposed in this thesis will have to be adapted to consider hysteresis between drying and wetting curves and will need to deal with evaporation effects.

There are however some complicating factors. In order to properly model the impoundment, a 2D flow module will be needed. The level of complexity could be lowered to only perform a direct recharge model, increasing computational efficiency. It is however unsure whether correct results are obtained by neglecting the unsaturated zone. The rise of the phreatic surface is less accurately modelled and omitting ponding together with run-off will create less realistic infiltration models. It is also still expected that these models are sensitive to the boundary conditions that are imposed to the geohydrological analysis. The phreatic level in the impoundment will be severely affected by any head in the surrounding foundation and valley. More research is needed into the occurring boundary conditions and the interaction in the flow between the impoundment and the surrounding valley.

Additionally, the modelling of construction defects will need attention. Various studies, that have also been treated in the literature study, show that with application of the upstream construction method and spigotting there are some inherent features which may not be favourable for stability. The first feature being the raising of the dam over weaker tailings deposits. Especially if so called set-backs are constructed. This may significantly affect the stability of a tailings dam and cause an initially stable dam suddenly to become only marginally stable. Recalling the fact that rainfall will especially influence the upper zone of a dam, the dam may be as weak as its last installed raise. Continuous warrant is therefore needed.

Another construction defect which may potentially be of large effect is the layering caused by spigotting. When e.g. material is mined with lots of joints with clay infill, these clay minerals are deposited in the impoundment forming a layer with a lower permeability. This will cause perched water tables in the impoundment. In light of the fact that the tailings material above the phreatic surface will become liquefiable near saturation, strength is significantly lowered and the return period of failure may therefore be decreased. The effect of perched water tables is expected to be high, but a 2D analysis is needed to correctly cope with this effect and allow for sideways drainage. Additionally, a more elaborate description of the permeability variation over the impoundment will be needed for these studies.

8.3. Recommendations for the general analysis of failure of tailings dams

Although this thesis has addressed the development of a return-period based method for rainfall-induced static liquefaction, the analysis methodology as presented is generally applicable to various other loads and failure mechanisms. This allows, to limited extent, for some recommendations to be provided for the general analysis of tailings dams. These will be treated in this final section. These recommendations will to a large degree reflect the author's opinion on the design and safety assessment of tailings dams, and may be regarded as an opinion piece on the topic. It is therefore also partially untied from the matter as expressed in the main body of the thesis.

In view of the problem statement which has been formulated at the start of this thesis, it was observed with great surprise by the author that there is very little focus on risk-based assessment in the field of tailings dams. And although these structures pose an enormous risk to their environment, these impoundments are not treated as such. Upon reading on the way in which a regular safety assessment is performed, the author believes that there are several reasons as to why a risk-based design and assessment has not been embraced into the field of tailings dams.

First of all, these structures differ greatly from normal water retention dams. Perhaps not in their function as they both retain water/soil, but mainly in their value to society. Water retention dams are often built for flood control, or to provide with water for irrigation of agricultural activities. Either way, these retention dams are built with the idea of providing added value to society. This is not the case for tailings dams. These dams are essentially just stockpiles of what may be regarded a waste material and a proper balance has to be obtained between the investment needed for the dam to be safe, in contrast to the risks these structures pose. At this moment, this balance is not equal and investments have been too little. With very strict legislation, the balance may however tilt the other way which should also be avoided. It is believed by the author that application of the downstream method of raising for tailings dams will solve a large set of problems. The phreatic level can better be controlled when e.g. installing an impermeable lining or drainage blankets. Furthermore, the outer shell will not prevent the tailings from liquefying when sheared, but the shell may be designed to not breach upon increased horizontal pressure because of liquefaction of the tailings. In that sense, the issue is tackled considerably. Demand for a risk-based design framework will however be omnipresent. As the failure rate of tailings dams is decreased, it is believed that different failure mechanisms become governing. The same can be seen with flood defences in the Netherlands. Due to increased safety standards as well as the changing loads and boundary conditions, work on these defences is never finished. It is believed by the author that the same can be expected for regular tailings dams.

Yet, there remains the issue of the management of decommissioned (upstream) dams. A popular belief is that the probability of failure of decommissioned dams generally decreases over time. It is argued that due to the halting of the spigotting process, less process water is released on top of the tailings dam. Water from the tailings pond will evaporate and the level of the phreatic surface decreases over time. Furthermore, any excess pore pressures may further disappear due to consolidation, increasing the effective stress. The author agrees with this view, but wants to express two important remarks. First of all, the failure of Feijão dam has shown the effect of a failure mechanism called ‘creep rupture’. Relatively quickly after deposition the iron ore tailings have formed oxidation bonds between leftover iron ore minerals. These oxidation bonds may well be regarded as some form of cohesion. Given the high stress ratio of deviatoric stress over confining stress in these upstream raised embankments, the tailings experience continuous deviatoric creep. These strains will break the oxidation bonds and hence the material will lose strength over time. It is argued by the expert panel that the loss of suction has given the final push to failure. Although this thesis shows to a large extent that loss of suction may not have been important in this failure, it is a very essential observation that the material strength deteriorates with time. Imagine the FoS of the dam to be a state parameter that differs at each instance. Due to e.g. rainfall infiltration, recharge to the water table, etc., the FoS will be fluctuating. With decreasing strength, there will be a moment where the FoS is lower than the failure FoS. There is the potential for these tailings dams to become marginally stable during time. No additional information is given on whether this could occur to other tailings dams as well. Yet again, the importance for a risk-based assessment of all failure mechanisms is advised as such deterioration effects can be properly assessed in such probabilistic frameworks.

Another issue with decommissioned dams is that they are products of an operational phase that often spans multiple decades. In that sense, the knowledge about any construction defects and embedded weaker/impermeable layers is largely missing. However, variability is hardly considered in the analysis of these dams. Especially the width of the non-liquefiable shell was found to be of great importance in the overall stability of the dam. Additional site investigation is needed on this parameter. In addition, probabilistic methods should be employed in order to take into consideration variability of this parameter. Earlier in these recommendations it was proposed to e.g. use Monte Carlo methods. It is however also brought to the attention that relatively simple design tools and principles should be implemented in order to be accepted in industry. Only then, the risks of tailings dams can effectively be decreased within a relatively short period of time. This means that computational time of models should be kept low whilst maximising the application of theoretically sophisticated models.

References

Abadjiev, C. B. (1976). Seepage through mill tailings. In *Proceedings of the 12th International Congress on Large Tailings Dams*, pages 381–392, Mexico city.

Alonso, E. E., Gens, A., and Josa, A. (1990). A constitutive model for partially saturated soils. *Géotechnique*, 40(3):405–430.

Associacão Brasileira de Normas Téchnicas (2017). Abnt nbr 13028: Mining - preparation and presentation of design of tailings, sediments and/or water dams - requirements. Technical report, Associacão Brasileira de Normas Téchnicas.

Aubertin, M., Ricard, J.-F., and Chapuis, R. P. (1998). A predictive model for the water retention curve: application to tailings from hard-rock mines. *Canadian Geotechnical Journal*, 35:55–69.

Bakker, M., Post, V., Langevin, C. D., Hughes, J. D., White, J. T., Starn, J. J., and Fienen, M. N. (2016). Scripting MODFLOW model development using Python and FloPy. *Groundwater*, 54(5):733–739.

Beersma, J., Hakvoort, H., Jilderda, R., Overeem, A., and Versteeg, R. (2019). Neerslagstatistiek en -reeksen voor het waterbeheer 2019. Technical report, Stichting toegepast onderzoek waterbeheer (Stowa). In Dutch.

Bella, G. (2017). *Hydro-Mechanical Behaviour of Tailings in Unsaturated Conditions*. PhD thesis, Politecnico di Torino, Torino, Italy.

Bhatt, J. P., Gandhi, H. M., and Gohil, K. (2014). Generation of intensity duration frequency curve using daily rainfall data for different return period. *Journal of International Academic Research for Multidisciplinary*, 2(2):717–722.

Bjerrum, L., Kringstad, S., and Kummeneje, O. (1961). The shear strength of a fine sand. In *Proc., 5th ICSMFE*, volume 1, pages 29–37.

Blight, G. E. (1994). The master profile for hydraulic fill tailings beaches. *Proceedings of the Institution of Civil Engineers - Geotechnical Engineering*, 107(1):27–40.

Blight, G. E. (2010). *Geotechnical Engineering for Mine Waste Storage Facilities*. CRC press, Taylor & Francis Group, London, UK.

Blight, G. E. (2013). *Unsaturated Soil Mechanics in Geotechnical Practice*. CRC press, Taylor & Francis Group, LLC, Boca Raton.

Bolzon, G., Schrefler, B. A., and Zienkiewizc, O. C. (1996). Elastoplastic soil constitutive laws generalized to partially saturated states. *Géotechnique*, 46(2):279–289.

Booshehrian, A., Wan, R., and Su, X. (2020). Hydraulic variations in permafrost due to open-pit mining and climate change: a case study in the Canadian Arctic. *Acta Geotechnica*, 15:883–905.

Brett, D., Gillani, I., Seddon, K., Himsley, N., McConnel, R., Bentel, G., Brown, B., Fourie, A., and Williams, D. (2012). Guidelines on tailings dam: planning, design, construction, operation and closure. Technical report, Australian National Committee on Large Dams.

Brown, B. and Gillani, I. (2016). Common errors in the slope stability analyses of tailings dams. In Dight, P. M., editor, *Proceedings of the First Asia Pacific Slope Stability in Mining Conference*, pages 545–556, Perth. Australian Centre for Geomechanics.

Cambridge, M., Coppin, N., Czajewski, K., Diaz, M., Ferguson, G., Molloy, C., Monroy, R., Roberts, J., and Saint, J. (2018). *The Hydraulic Transport and Storage of Extractive Waste: Guideline to European Practice*. Springer International Publishing.

Celia, M. A. and Bouloutas, E. T. (1990). A general mass-conservative numerical solution for the unsaturated flow equation. *Water Resources Research*, 26(7):1483–1496.

Chandler, R. J. and Tosatti, G. (1995). The stava tailings dams failure, Italy, July 1985. *Proceedings of the Institution of Civil Engineers: Geotechnical Engineering*, 113(2):67–79.

Cheng, Y. M. (2003). Location of critical failure surface and some further studies on slope stability analysis. *Computers and Geotechnics*, 30:255–267.

Chu, J., Leroueil, S., and Leong, W. K. (2003). Unstable behaviour of sand and its implication for slope instability. *Canadian Geotechnical Journal*, 40:873–885.

Chu, J. and Wantowski, D. (2012). New design approach for tailing dams based on recent study on instability of sand. In *Third International Workshop on Modern Trends in Geomechanics IW-MTG3*, pages 64–66, Nottingham.

Chui, T. F. M. and Freyberg, D. L. (2009). Implementing hydrologic boundary conditions in a multiphysics model. *Journal of Hydrologic Engineering*, 14(12):1374–1377.

Clout, J. M. F. and Manuel, J. R. (2015). Mineralogical, chemical and physical characteristics of iron ore. In *Iron Ore: Mineralogy, Processing and Environmental Sustainability*, chapter 2. Woodhead Publishing, Elsevier Ltd.

Davies, M. P. (2002). Tailings impoundment failures: Are geotechical engineers listening? *Geotechnical News*.

de Assis Diniz, F., Ramos, A. M., and Rebello, E. R. G. (2018). Brazilian climate normals for 1981-2010. *Pesquisa Agropecuária Brasileira*, 53(2):131–143.

de Carvalho, J. R. P., Assad, E. D., de Oliveira, A. F., and Pinto, H. S. (2014). Annual maximum daily rainfall trends in the midwest, Southeast and Southern Brazil in the last 71 years. *Weather and Climate Extremes*, 5-6:7–15.

DRR Team Brazil (2019). DRR Team mission report Minas Gerais state, Brazil. Technical report, Dutch Risk Reduction Team.

Dussaillant, A. R., Wu, C. H., and Potter, K. W. (2004). Richards equation model of a rain garden. *Journal of Hydrologic Engineering*, 9(3):219–225.

Eckersley, D. (1990). Instrumented laboratory flowslides. *Géotechnique*, 40(3):489–502.

Fredlund, D. G. and Krahn, J. (1977). Comparison of slope stability methods of analysis. *Canadian Geotechnical Journal*, 14:429–439.

Fredlund, D. G., Rahardjo, H., and Fredlund, M. D. (2012). *Unsaturated Soil Mechanics in Engineering Practice*. John Wiley & Sons, Inc.

Geo-slope international Ltd. (2015). Stability modeling with Slope/W: An engineering methodology. Technical report, Geo-slope international Ltd.

Gibson, R. E. (1958). The progress of consolidation in a clay layer increasing in thickness with time. *Géotechnique*, 8(4).

Giongo, V., Salviano, A. M., Angelotti, F., Taura, T., Leite, L. F. C., and Cunha, T. J. F. (2018). Low carbon technologies for agriculture in dryland: Brazilian experience. In *Climate Resilient Agriculture - Strategies and Perspectives*. Intech.

Green, W. H. and Ampt, G. A. (1911). Studies on soil physics, part I: The flow of air and water through soils. *Journal of Agricultural Science*, 4:1–24.

Grozic, J. L., Robertson, P. K., and Morgenstern, N. R. (1999). The behavior of loose gassy sand. *Canadian Geotechnical Journal*, 36:482–492.

Harbaugh, A. W. (2005). MODFLOW-2005, the U.S. Geology Survey modular ground-water model – the ground-water flow process. Technical Report 6-A16, U.S. Geological Survey Techniques and Methods.

Heinen, M., Mulder, M., and Kroes, J. (2020). SWAP 4 - Technical addendum to the SWAP documentation. Technical report, Wageningen Environmental Research.

Hu, L., Wu, H., Zhang, L., Zhang, P., and Wen, Q. (2017). Geotechnical properties of mine tailings. *Journal of Materials in Civil Engineering*, 29(2).

ICOLD Committee on Tailings Dams and Waste Lagoons (1995-2001) (2001). Tailings dams risk of dangerous occurrences: Lessons learnt from practical experiences. Bulletin 121, International Committe on Large Dams.

Ippisch, O., Vogel, H. J., and Bastian, P. (2006). Validity limits for the van Genuchten-Mualem model and implications for parameter estimation and numerical simulation. *Advances in Water Resources*, 29:1780–1789.

Jasim, F. H., Vahedifard, F., Ragno, E., Aghakouchak, A., and Ellithy, G. (2017). Effects of climate change on fragility curves of earthen levees subjected to extreme precipitations. In *Geo-risk 2017: Reliability-based Design and Code Developments*, pages 498–507, Denver, USA.

Jefferies, M. and Been, K. (2016). *Soil Liquefaction: A Critical State Approach*. Taylor & Francis Group, LLC, Boca Raton, second edition.

Jefferies, M. G. (1993). Nor-sand: a simple critical state model for sand. *Géotechnique*, 43(1):91–103.

Kim, J., Jeong, S., and Regueiro, R. A. (2012). Instability of partially saturated soil slopes due to alteration of rainfall pattern. *Engineering Geology*, 147-148:28–36.

Klohn, E. J. (1979). Seepage control for tailings dams. In *Tailings and Waste Disposal - Seepage, Contamination, Regulations and Control*, chapter Section 4. International Mine Water Association.

Koutsoyiannis, D., Kozonis, D., and Manetas, A. (1998). A mathematical framework for studying rainfall intensity-duration-frequency relationships. *Journal of Hydrology*, 206:118–135.

Kroes, J. G., van Dam, J. C., Bartholomeus, R. P., Groenendijk, P., Heinen, M., Hendriks, R. F. A., Mulder, H., Supit, I., and van Walsum, P. (2017). SWAP version 4: Theory description and user manual. Technical report, Wageningen Environmental Research, Wageningen. Report 2780, distributed under GNU General Public License version 2, June 1991.

Ladd, C. C. (1991). Stability evaluation during staged construction. *Journal of Geotechnical Engineering*, 117(4).

Lade, P. V. (1992). Static instability and liquefaction of loose fine sandy slopes. *Journal of Geotechnical Engineering*, 118(1):51–71.

Lade, P. V. and Yamamuro, J. A. (2011). Evaluation of static liquefaction potential of silty sand slopes. *Canadian Geotechnical Journal*, 48:247–264.

Maity, R. (2018). *Statistical Methods in Hydrology and Hydroclimatology*. Springer Nature, Singapore.

Martin, T. E. and McRoberts, E. C. (1999). Some considerations in the stability analysis of upstream tailings dams. *Tailings and mine waste*, pages 287–302.

Morgenstern, N. R. (2018). Geotechnical risk, regulation and public policy.

Morgenstern, N. R., Jefferies, M., van Zyl, D., and Wates, J. (2019). Report on NTSF embankment failure. Technical report, Cadia Valley Operations.

Morgenstern, N. R., Vick, S. G., and van Zyl, D. (2015). Report on Mount Polley tailings storage facility breach. Technical report, Province of British Columbia.

Morgenstern, N. R., Vick, S. G., Viotti, C. B., and Watts, B. D. (2016). Report on the immediate causes of the failure of the Fundão dam. Technical report, Fundão Tailings Dam Review Panel.

NCEP (2020). Global weather data for SWAT. `https://globalweather.tamu.edu/`.

Ng, C. W. W. and Shi, Q. (1998). A numerical investigation of the stability of unsaturated soil slopes subjected to transient seepage. *Computers and Geotechnics*, 22(1):1–28.

Noguiera, M. and Plumb, C. (2020). Exclusive: Brazil prosecutor aims to charge vale within days over mining waste dam disaster. `https://www.reuters.com/article/us-vale-sa-disaster-exclusive/exclusive-brazil-prosecutor-aims-to-charge-vale-within-days-over-mining-waste`

Oliviera, M. G. and Kerbany, M. T. (2016). Environmental vulnerability and technological risks in collapse and break of dams in Brazil: lessons from Mariana (MG) disaster. In *15th Conference Global Spatial Data Infrastructure Association*, pages 539–546, Taipei.

Olson, S. M. (2001). *Liquefaction Analysis of Level and Sloping Ground Using Field Case Histories and Penetration Resistance*. PhD thesis, University of Illinois, Urbana-Campaign.

Olson, S. M. and Stark, T. D. (2003). Yield strength ratio and liquefaction analysis of slopes and embankments. *Journal of Geotechnical and Geoenvironmental Engineering*, 129(8).

Overeem, A., Buishand, A., and Holleman, I. (2008). Rainfall depth-duration-frequency curves and their uncertainties. *Journal of Hydrology*, 348:124–134.

Owen, J. R., Kemp, D., Lèbre, E., Svobodova, K., and Murillo, G. P. (2020). Catastrophic tailings dam failures and disaster risk disclosure. *International Journal of Disaster Risk Reduction*, 42:1–10.

Pak, A. and Nabipour, M. (2017). Numerical study of the effects of drainage systems on saturated/unsaturated seepage and stability of tailings dams. *Mine Water and the Environment*, 36:341–355.

Palu, M. C. and Julien, P. Y. (2019). Review of tailings dam failures in Brazil. In *XXIII Simpósio Brasiliero de recursos hidrícos*, Foz do Iguacu, Brazil.

Panagoulias, S. and Brinkgreve, R. B. J. (2017). Shansep NGI-ADP: model description and verification examples. Technical report, POV Macro-stabiliteit.

Pettibone, H. C. and Kealy, C. D. (1971). Engineering properties of mine tailings. *Journal of the Soil Mechanics and Foundations Division*, 97(SM9):1207–1255.

Qiu, Y. and Sego, D. C. (2001). Laboratory properties of mine tailings. *Canadian Geotechnical Journal*, 38:183–190.

Queiroz, H. M., Nóbrega, G. N., Ferreira, T. O., Almeida, L. S., Romero, T. B., Santaella, S. T., Bernardino, A. F., and Otero, X. L. (2018). The Samarco mine tailing disaster: A possible time-bomb for heavy metals contamination? *Science of the total environment*, 637-638:498–506.

Rahardjo, H. and Fredlund, D. G. (1991). Calculation procedures for slope stability analyses involving negative pore-water pressures. In *Proceedings of the International Conference on Slope Stability*, pages 43–49, Isle of Wight, England.

Rahardjo, H., Ong, T. H., Rezaur, R. B., and Leong, E. C. (2007). Factors controlling instability of homogeneous soil slopes under rainfall. *Journal of Geotechnical and Geoenvironmental Engineering*, 133(12):1532–1543.

Rahimi, A., Rahardjo, H., and Leong, E.-C. (2010). Effect of hydraulic properties of soil on rainfall-induced slope failure. *Engineering Geology*, 114:135–143.

Rassam, D. W. (2002). Variation of evaporative and shear strength parameters along a tailings delta. *Canadian Geotechnical Journal*, 39:32–45.

Raynal, J. A. and Salas, J. D. (1986). Estimation procedures for the type-1 extreme value distribution. *Journal of Hydrology*, 87:315–336.

Rickard, O. C. and Sitar, N. (2012). bSlope: A limit equilibrium slope stability analysis code for iOS. Geotechnical engineering report, UC Berkely.

Rico, M., Benito, G., Salgueiro, A. R., Díez-Herrero, A., and Pereira, H. (2008). Reported tailings dam failures: A review of the European incidents in the worldwide context. *Journal of Hazardous Materials*, 152:846–852.

Robertson, A. M. (1987). The influence of depositional methods on the engineering properties of tailings deposits. In *International Waste Management Conference*, Johannesburg, South Africa.

Robertson, P. K. (2010). Evaluation of flow liquefaction and liquefied strength using cone penetration test. *Journal of Geotechnical and Geoenvironmental Engineering*, 136(6):842–853.

Robertson, P. K., da Fonseca, A. V., Ulrich, B., and Coffin, J. (2017). Characterization of unsaturated mine waste: a case history. *Canadian Geotechnical Journal*, 54:1752–1761.

Robertson, P. K., de Melo, L., Williams, D. J., and Wilson, G. W. (2019). Report of the expert panel on the technical causes of the failure of the Feijão Dam I.

Rodriguez-Mier, P. (2017). A tutorial on differential evolution with python. https://pablormier.github.io/2017/09/05/a-tutorial-on-differential-evolution-with-python/.

Sadrekarimi, A. (2016). Static liquefaction analysis considering principal stress directions and anisotropy. *Geotech Geol Eng*, 34:1135–1154.

Schaap, M. G. and van Genuchten, M. T. (2005). A modified Mualem-van Genuchten formulation for improved description of the hydraulic conductivity near saturation. *Vadose Zone Journal*, 5:27–34.

Shuttle, D. A. and Cumming, J. (2007). Liquefaction potential of silts from CPTu. *Canadian Geotechnical Journal*, 44:1–19.

Spencer, E. (1967). A method of analysis of the stability of embankments assuming parallel inter-slice forces. *Géotechnique*, 17:11–26.

Stark, T. D., Choi, H., and Schroeder, P. R. (2005). Settlement of dredged and contaminated material placement areas I: Theory and use of primary consolidation, secondary compression and dessication of dredged fill. *Journal of Waterway, Port, Coastal and Ocean Engineering*, 131(2):43–51.

Tfwala, C. M., van Rensburg, L. D., Schall, R., Mosia, S. M., and Dlamini, P. (2017). Precipitation intensity-duration-frequency curves and their uncertainties for Ghaap plateau. *Climate Risk Management*, 16:1–9.

Vale (2020). Mine tailings disclosure table. `http://www.vale.com/PT/investors/services-investor/Documents/Tailings%20Safety%20Disclosure_ingles_valecom.pdf`.

van Dam, J. C., de Rooij, G. H., Heinen, M., and Stagnitti, F. (2004). Concepts and dimensionality in modeling unsaturated water flow and solute transport. In Feddes, R., de Rooij, G., and van Dam, J., editors, *Unsaturated-zone modeling; progress, challenges and applications*, volume 6, pages 1–36. Wageningen UR Frontis Series.

Vardon, P. J., Yao, Y., van Paassen, L. A., and van Tol, A. F. (2015). The use of a large-strain drying and consolidation model to optimise multilift tailing deposits. In *Tailings and Mine Waste Management for the 21st Century*, pages 1–8, Sydney, NSW.

Vick, S. G. (1983). *Planning, Design and Analysis of Tailings Dams*. John Wiley & Sons, Inc.

Vick, S. G. (1992). Stability evaluation during staged construction, discussion by Steven G. Vick. *Journal of Geotechnical Engineering*, 118(8).

Vogel, T., van Genuchten, M. T., and Cislerova, M. (2000). Effects of the soil hydraulic functions near saturation on variably-saturated flow predictions. *Advances in water research*, 24:133–144.

Volpi, E. (2019). On return period and probability of failure in hydrology. *WIREs Water*, 6.

Zandarín, M. T., Oldecop, L. A., Rodríguez, R., and Zabala, F. (2009). The role of capillary water in the stability of tailing dams. *Engineering Geology*, 105:108–118.

Zhang, C., Liu, H., Yang, C., and Wu, S. (2016a). Mechanical characteristics of non-saturated tailings and dam stability. *International Journal of Mining, Reclamation and Environment*, 31(8):530–543.

Zhang, L., Li, J., Li, X., Zhang, J., and Zhu, H. (2016b). *Rainfall-induced Soil Slope Failure*. Proceedings of the ICE-Geotechnical Engineering, Boca Raton.

Zhang, L. L., Zhang, L. M., and Tang, W. H. (2005). Rainfall-induced slope failure considering variability of soil properties. *Géotechnique*, 55(2):183–188.

Zielinski, K. and Laur, R. (2008). Stopping criteria for differential evolution in constrained single-objective. In *Advances in Differential Evolution*, chapter 4, pages 111–138. Springer-verlag, Berlin Heidelberg.

APPENDIX A

Fitting the extreme value distribution

This appendix will treat the assumptions of a stationary and spatially homogeneous dataset. For this study, the data of 45 weather stations in an area South of Belo Horizonte, Brazil was used. The bounding coordinates of the box encapsulating these stations are: (-19.8248,-42.4361), (-21.2036,-45.0124). Data was retrieved from (NCEP, 2020). The 24-hour total precipitation depth is available for each station for a period from 1979 to 2014. In total, this means that 1620 station years are available for the analysis.

A hot debate within return periods in hydrology is to assume if the dataset is stationary or non-stationary. Stationarity implies that there is no change of the distribution or its parameter estimates with time. Assuming stationarity immediately implies also that the effects of e.g. climate change are not considered for the extrapolation period when considering return periods (Volpi, 2019). The dataset as obtained was therefore analysed to verify if stationarity could be assumed. When applying a visual analysis of the dataset, non-stationarity could not be seen and most of the dataset showed behaviour similar to Figure A.1. The relatively short period of data could be the cause of this. As an extension, the augmented Dickey-Fuller (ADF) test was applied to the dataset to look for any systematic changes in the mean, either by changing constant or as a trend. The results of this test can be seen in Table A.1. A significance level of 0.05 was used to reject the null hypothesis. If this was the case and the p-value was lower than 0.05, the dataset was assumed to be stationary. If it was larger (as also presented for those cases in the table), a trend, or shifting mean, was present in the dataset. This is the case for 13 out of 45 stations. Of these 13 stations, 11 stations had a rising trend in the intensity. An example can be seen in Figure A.2. As the largest part of the dataset is stationary and a non-stationary analysis would demand a level of statistical analysis not appropriate to this study, it was decided to assume the dataset as stationary for the fitting of the distributions.

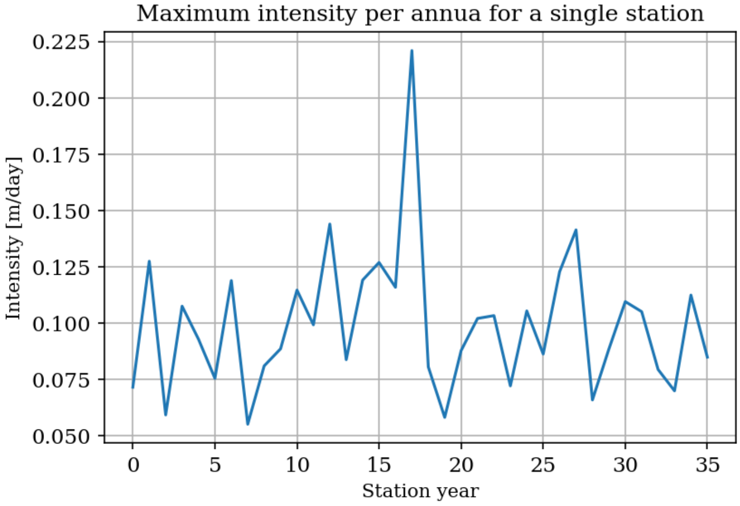


Figure A.1: The maximum intensity per year for a single weather station for the recorded station years.

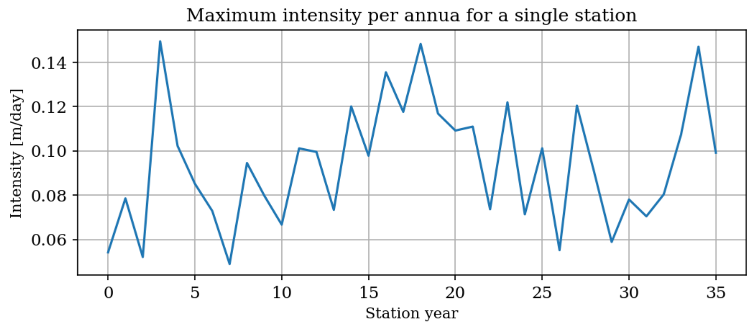


Figure A.2: An example of a non-stationary dataset for a weather station for the recorded station years.

Table A.1
Result of Augmented Dickey-Fuller tests over all station data. The significance (p-) value is provided for values close to 0.05.

Station	ADF value	p-value	Station	ADF-value	p-value
1	-5.89		24	-4.46	
2	-5.06		25	-4.61	
3	-6.15		26	-3.13	0.098
4	-5.38		27	-3.09	0.109
5	-2.78	0.205	28	-4.77	
6	-4.60		29	-3.42	0.049
7	-4.45		30	-6.35	
8	-3.03	0.122	31	-6.39	
9	-3.09	0.109	32	-6.94	
10	-5.73		33	-5.74	
11	-5.01		34	-4.77	
12	-5.38		35	-4.74	
13	-5.40		36	-3.38	0.054
14	-2.97	0.141	37	-5.50	
15	-5.05		38	-4.54	
16	-5.35		39	-5.57	
17	-4.43		40	-4.78	
18	-3.45	0.045	41	-5.12	
19	-5.30		42	-2.84	0.184
20	-2.51	0.039	43	-1.05	0.927
21	-3.59		44	-2.32	0.424
22	-6.02		45	-3.32	0.063
23	-5.97				

Apart from the assumption of stationarity, also the assumption is made that the data is spatially homogeneous. In other words, the data of the stations is identically distributed. This means that not only they hold the same underlying distribution, but also the same distribution parameters. Considering the size of the area examined, this seems reasonable. No statistical tests will however be used to identify whether this is true. Following this assumption, the data of all the stations was assembled in a single set and the fits of the distribution could be performed on all data, achieving a total of 1620 station years of data. Of these 1620 years, 1620 annual maxima were extracted for further analysis.

APPENDIX B

Framework of analysis and earlier phases in the study

The proposed methodology in Chapter 4 has been the result of a longer process in which at first a general framework was established for the analysis of rainfall-induced static liquefaction of tailings dams. This general framework defined various possible levels of analysis. In total three phases of analysis have been performed. The first phase starting at the simplest possible level. For each part in the general framework, the level of complexity was increased if argued necessary. The final methodology and results (of phase three) have been treated in the main matter. This appendix will show the general framework for the rainfall-induced static liquefaction of tailings dams in Section B.1. Furthermore, results, conclusions and recommendations are provided from the two earlier phases of analysis in Sections B.2 and B.3.

B.1. General framework for the rainfall-induced static liquefaction of tailings dams

Reflecting on the conclusion of chapter 3, it was identified that in essence four different analyses are needed. These are:

1. Spatial characterisation: define the distribution of strength, permeability and stiffness over the impoundment.
2. Consolidation analysis: provides with correct void ratio and effective stress at different instances of time. It may be related to the spatial characterisation.
3. Geohydrological analysis: determination of the position of the phreatic surface as well as resulting pressure heads upon rainfall infiltration.
4. Liquefaction analysis: determines based on provided information from the previous analyses whether failure will occur.

Performing these analyses will provide with an overall picture of the safety of the tailings dam. Or more specifically on the probability of failure. The general framework for analysis is visualised in Figure B.1. Shown are the various parts described as well as the input and output for analysis. The input is in this case related to the material parameters of the tailings as well as a detailed description of the construction history of the impoundment. Furthermore, the load and its magnitude should clearly be specified. The probabilistic model used is in essence independent of the various analyses and may be applied in different form to each of the different parts. Together it provides with a probability of failure for the impoundment.

Levels of complexity were defined for each of these parts. Specifying these levels allows for a clear overview of the research that has been performed and the assembly of a final analysis methodology for this study. Although it is arbitrary as to what may be considered ‘high’ and ‘low’ levels of analysis, the defined levels are shown in Table B.8 at the end of this appendix.

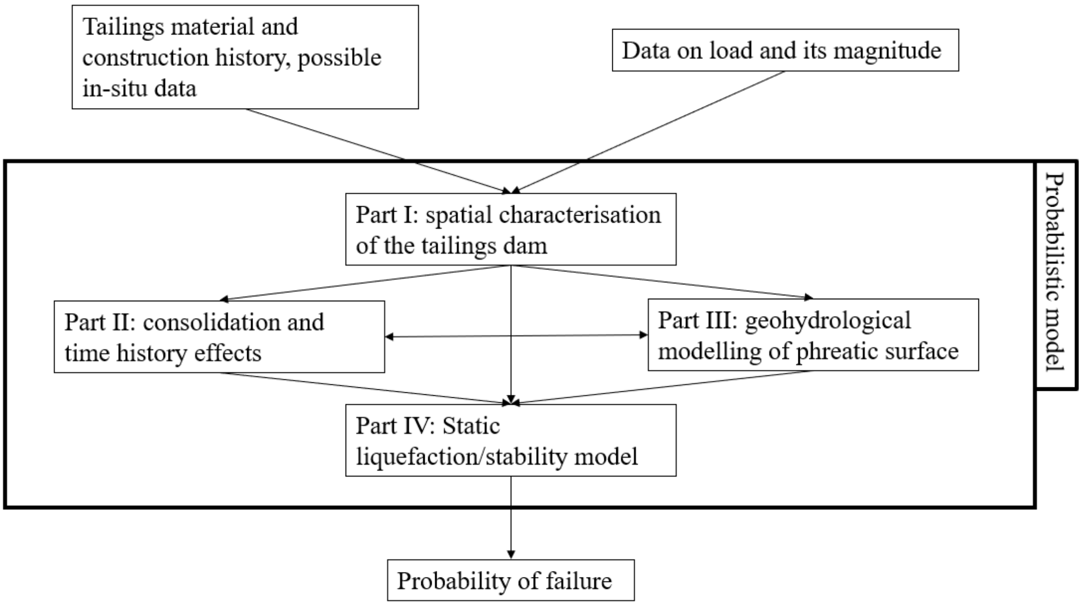

Tailings material and construction history, possible in-situ data
Data on load and its magnitude
Part I: spatial characterisation of the tailings dam
Part II: consolidation and time history effects
Part III: geohydrological modelling of phreatic surface
Part IV: Static liquefaction/stability model
Probabilistic model
Probability of failure

Figure B.1: The general framework of analysis for rainfall-induced static liquefaction of tailings dams

Acknowledging that there has been little integrated research in the field of rainfall-induced static liquefaction, it was decided to start the study at a low level analysis and increase this level when it was argued necessary. In the end, three 'phases' have been performed. B.1 provides with the various levels that were implemented in each phase. The phases 1 and 2 will now be treated in the rest of this appendix. Phase 3 has been treated in full detail in the main matter.

Table B.1
Overview of the different levels at each part applied during the different phases of the study. The levels correspond to those defined in Table B.8.

	Phase 1	Phase 2	Phase 3
Part I: Spatial characterisation	Level 1	Level 1	Level 2
Part II: Consolidation analysis	Level 1	Level 1	Level 1
Part III: Geohydrological analysis	Level 2	Level 4	Level 4
Part IV: Static liquefaction analysis	Level 1	Level 1	Level 1

B.2. Phase 1: methods, results and discussion

This section will describe the methodology and results of the 'phase 1' analysis. Phase 1 was the first analysis that was performed and was conducted to get an idea of the sensitivity of the FoS to rainfall for different slope and strength parameters. For each of the different parts, a simple level of analysis was applied. These were:

- Part I - Spatial characterisation: fully homogeneous dam of liquefiable material.
- Part II - Consolidation analysis: fully consolidated material, no excess pore pressures due to loading.
- Part III - Geohydrological analysis: direct recharge solution to the Laplace equation with given boundary conditions.

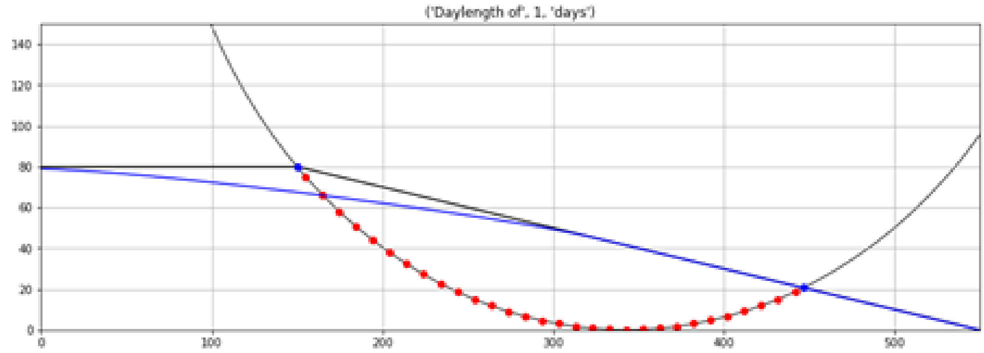


Figure B.2: An example of a circular slip surface for the dam, with in blue the phreatic surface.

- Part IV - Liquefaction analysis: application of peak strength ratios for circular slip surface based on Bishops formulation for limit equilibrium.

The IDF-curve as implemented in the final phase was applied in the same manner in this early phase. Key difference is the sole use of the GEV distribution. The Gumbel distribution, which proved to be a better estimate, was not applied here. In addition, there is the key difference in the manner to which the geohydrological analysis was performed. In this case, unsaturated flow was neglected and any rainfall would directly be added as recharge to the water table. Recalling that there are two ways in which the factor of safety of a slope may be reduced upon rainfall infiltration, by loss of suction or by an increase of the phreatic surface, this first analysis was only capable of modelling the increase in the phreatic surface. Modflow was implemented to perform this direct recharge model (Harbaugh, 2005). The dam was fully homogeneous and peak strength ratios were also applied above the phreatic surface. In that sense, it was implied that, regardless of the degree of saturation or grain size, the tailings will always experience strain softening behaviour.

What results, is a purely frictional slope, comparable to when applying an effective stress analysis with a cohesion of the material equal to zero. Frictional slopes tend to have a slip surface that is parallel to the slope at some limited depth. This depth is depending on the phreatic conditions. Considering the fact that only a circular slip circle formulation was implemented, the result was that the differential evolution (DE) algorithm moved to failure surfaces with a very small radius. This was acknowledged early in the process and it was believed that this was an improbable mode of failure. As for a normal tailings dam, the material along the slope would be compacted and have higher strength. Therefore, the entry point was limited to the flat part along the top of the slope and the exit point would be along a stretch near the toe of the slope. An example of such a slip surface can be seen in Figure B.2. The horizontal tangent of the circle was limited to have a y-coordinate of 0. This was a feature not directly implemented in the code, but for slice bases below 0, a very high strength was assigned so that the differential evolution algorithm would diverge from those solutions.

A sensitivity analysis was performed based on the needed parameters for this analysis. Those parameters were:

- geometry - slope and height;
- yield strength ratio;
- saturated hydraulic conductivity;
- anisotropy of permeability k_h/k_v;
- boundary condition at the upstream end of the impoundment;
- infiltration factor: ratio of run-off and direct recharge.

At first, a small study was performed over some parameters, immediately showing that there is a large effect of the strength. These calculations were performed with a poor estimation of the GEV distribution. Too high probabilities of occurrence were attributed to the tail, leading to small return periods of failure. The results of this small simulation will therefore not be given here as to not supply the reader with any erroneous results. The GEV distribution fit was improved upon and a sensitivity study over the strength was performed. The other parameters were fixed at values as given in Table B.2.

Table B.2
Parameters applied in sensitivity analysis of the peak strength ratios

Variable	Value
Height	80 [m]
Slope	17 [°]
k_{sat}	4.32 [m/day]
Anisotropy	5 [-]
Boundary condition	79 [m]
Infiltration factor	0.5 [-]

These parameters were mostly based on Robertson et al. (2019). The parameters that belonged to 'sand tailings' were chosen. The infiltration factor is a factor that determines how much rainfall will be infiltrating versus the rainfall that will run-off. This prevents a too large amount of water to be considered in the rise of the phreatic surface. In Robertson et al. (2019), a value of 0.5 was found, based on a 1D column analysis for unsaturated flow.

An analysis was performed for a larger set of peak strength ratios. The results of this can be seen in Table B.3. The return period is capped at 10,000 years.

Table B.3
Parameters applied in sensitivity analysis of the peak strength ratios

Peak strength ratio [-]	Return period [a]	Needed intensity for 1,2,.. days [m/day]
0.22	0	-
0.23	0	-
0.24	0	-
0.25	0	-
0.26	0	-
0.27	0	-
0.28	> 10 000	0.93 - 0.67 - 0.61 - 0.59 - 0.57
0.29	> 10 000	>2 - 1.66 - 1.52 - 1.47 - 1.46

As can be seen, there is only a very narrow margin between the peak strength ratio of 0.27 and 0.28 for which this model yields reasonable results in terms of intensity to failure. At a strength ratio of 0.27 the dam will fail immediately. At a value of 0.28, the return period of failure is extremely high. In general, this means that the failure mode for rainfall infiltration is not solely dependant on a rise of the phreatic surface considering storm conditions. The needed rise of the phreatic surface to failure is essentially too high. In hindsight, the value of the hydraulic conductivity chosen here was too large. This allowed for a significant outflow of water out of the system. This can also be seen in the fact that the needed intensity per day does only minimally decrease for longer duration storms.

To consider the effect of other parameters, a simulation matrix was built for additional analysis. Here a peak strength ratio of 0.28 would only be considered. The simulation matrix for combinations of parameters studied can be seen in Table B.4.

Table B.4
Simulation matrix of various parameters for the analysis of a single yield ratio.

Stability analysis			
Geometry	Slope	17 [°]	
	Height	80 [m]	
Peak strength ratio [-]	0.28		
Unit weight [kN/m^3]	20	22	24

Geohydrological analysis			
k_{sat} [m/day]	0.432	4.32	43.2
Anisotropy [-]	5	7	10
Boundary condition [m]	79		
Infiltration factor [-]	0.5		

There are basically only three parameters that are varied in this analysis. Those are: unit weight, saturated permeability and the anisotropy. Other parameters were not changed in this phase as their influence had been studied already (such as for the peak strength) or there was not enough information available on the possible range of values. The latter was the case for the boundary condition as well as the infiltration factor. Only a single geometry was considered in the analysis. Altering the geometry would not have a direct effect on the return period as the geohydrological model was a direct recharge model. Flow through the unsaturated zone is neglected and changing the height will therefore not show an effect, other than changing the initial factor of safety. This is also true for the slope of the geometry. The resulting return period for all combinations of the three varying parameters can be seen in Table B.5.

These results provide little clarity on the sensitivity of the outcome towards the various parameters. It seems that the unit weight dominates the results of these three. Lower unit weight leads to a lower factor of safety. At first sight this is odd since the loading is significantly reduced. However, the vertical effective stress is reduced as well. Therefore the total strength is lowered, since peak strength ratios are used based on the initial vertical effective stress. The strength is calculated as a ‘cohesion’ for use in the limit equilibrium formulation. Under dry circumstances, these would cancel each other out as both loading and strength will decrease proportionally. Under wet circumstances (so below the phreatic surface) this is no longer the case. Since the effective weight changes disproportionally to the total weight, a lower unit weight will decrease the strength by a more significant amount than the load. Hence, the factor of safety will decrease. The permeability and anisotropy are shown to be of relatively little influence.

What may also be interpreted from these results is that in essence the influence of rainfall on the factor of safety when only considering direct recharge is very minimal. This way, the return period jumps from 0 to a value larger than 10000 for only a relatively small change in the parameters. Reflecting on literature, and mostly on the work of Zandarín et al. (2009), it was decided that in order to increase the effect of rainfall, the unsaturated zone would need to be modelled. A loss of suction in the area above the phreatic surface can potentially impose a larger effect on the factor of safety. In that way, for phase 2 some parts were upgraded to higher levels:

- improve geohydrological analysis to model unsaturated zone;
- change factor of safety computation to include the effects of suction above the PS;
- implement a non-circular failure surface finder.

Reflecting on phase 1 after having completed the phase 3 analyses, the phase 1 analysis proved to be valuable in showing the effects of the failure mode where only an increase in the phreatic surface

Table B.5
Result of simulations for varying unit weight, permeability and anisotropy of permeability.

Unit weight $[kN/m^3]$	k_{sat} [m/day]	Anistropy [-]	Return period [a]
20	4.32	5	0
22	4.32	5	>10000
24	4.32	5	-
20	4.32	7	0
22	4.32	7	>10000
24	4.32	7	-
20	4.32	10	0
22	4.32	10	>10000
24	4.32	10	-
20	43.2	5	0
22	43.2	5	-
24	43.2	5	-
20	43.2	7	0
22	43.2	7	-
24	43.2	7	-
20	43.2	10	0
22	43.2	10	-
24	43.2	10	-
20	0.432	5	0
22	0.432	5	>10000
24	0.432	5	>10000
20	0.432	7	0
22	0.432	7	>10000
24	0.432	7	>10000
20	0.432	10	0
22	0.432	10	>10000
24	0.432	10	>10000

was considered. It is argued though, that phase 1 should have been studied more extensively, e.g. by choosing intermediate values for the various variables to try to find which combination of values show reasonable return periods. Furthermore, the value for the hydraulic conductivity proved to not be fully correct after a more extensive literature study. Lower values of the hydraulic conductivity would have reduced the outflow horizontally and therefore the phreatic surface would rise more for the given combination of intensity and duration. In addition, the effect of capillary rise could also be numerically imposed by choosing a very low specific yield within the Modflow model. Then a small addition of water would cause a rather large change in the phreatic surface and the idea of the air entry pressure and capillary rise could basically have been mimicked. Nonetheless, these artificial implementations of rather complex geohydrological phenomena would not be scientifically accurate.

The simple analysis that was employed in this phase is however recommended for a possible study of the change of the phreatic surface over time. The model was computationally rather inexpensive and could therefore be used for studies such as the Monte Carlo Markov Chain. It will show how the phreatic surface behaves under infiltration and evaporation and could therefore help with providing a more accurate idea of possible boundary and initial conditions to analysis. It could also be used to determine a return period of failure based on a lifetime simulation of the impoundment.

B.3. Phase 2: methods, results and discussion

This section will present the methods and results of the 'phase 2' analysis. Moreover, these results will be discussed and it will be argued why a new and improved phase was needed. That phase (phase 3) and its results have been discussed in Chapters 4 and 6. Phase 2 took most time in the process of this thesis, because of the large increase in the complexity of the geohydrological analysis. Following the conclusion to Appendix B.2, an unsaturated flow model was needed for an accurate description of suction within the slope. In addition to that, the FoS calculation procedure needed to be adapted in order to consider suction effects. Furthermore, a non-circular failure surface was applied rather than a circular failure surface. This also meant that the limit equilibrium formulation was changed from Bishop to Spencer in order to accurately model the forces along the non-circular failure surface.

First, the applied methods will be discussed in further detail. Here, also the process of the development of the geohydrological analysis and the difficulties encountered will be considered, to provide with background. After that, the results of this phase will be presented and discussed. The appendix is closed off with the recommendations for the final phase of the analysis as presented in the main body of this thesis.

B.3.1. Methodology

Development of the geohydrological analysis

As highlighted before, the geohydrological analysis had to be increased in complexity from a direct recharge model to full modelling of the unsaturated zone. Given this complexity, it was at first decided to take code from a course at TU Delft in which the Richards equation was solved in Python. The code was enhanced and made suitable for the analysis in this thesis. It was immediately chosen to see if a quasi-2D analysis could be applied in order to avoid complicated analyses. The use of more common software like Hydrus or Plaxis was not considered at this point as to avoid unnecessary computational time or the disfunctional use of batch scripting to initiate the calculation process. The Python code implemented in this phase will not be shown as to avoid confusion by provision of erroneous code.

The code solved for the Richards equation by applying the standard differential equation solvers in Python. In this case, the *solve ivp* function of the scipy package was used. This function uses the 'LSODA' method for solving. The LSODA method is an old Fortan solver, which applies automatic stiffness detection and switching between the Adams and BDF method. Especially the latter is suitable for stiff differential equations, which is also the case for the Richards equation. The code applies a vector form head-based formulation of the Richards equation. The code uses the Mualem-van Genuchten equation to calculate the hydraulic conductivity based on the effective degree of saturation. The Jacobian is assembled and provided to the solver in order to speed up the calculation.

The code did perform well, but had trouble with the implementation of the boundary condition. The boundary condition had to make sure that switching between a discharge or head-based, i.e. Neumann or Dirichlet, boundary condition was possible. This prevents the head to increase by a too large amount. Also it allows for any run-off modelling on e.g. a slope. Based on Chui and Freyberg (2009), such a boundary condition was imposed. This however performed poorly and the computational efficiency was low. Furthermore it was considered that a solely head-based differential equation could lead to mass balance defects. A mixed form is therefore advised (Zhang et al., 2016b). Such mixed form was proposed by Celia and Bouloutas (1990) and has become the standard within modelling of the unsaturated zone and is applied now in many models such as e.g. Dussaillant et al. (2004) and of course SWAP by Kroes et al. (2017). More on the various implementations of unsaturated flow, their methods of modelling and the different pitfalls can be found in van Dam et al. (2004).

Reflecting on the literature on unsaturated flow modelling, it was decided to neglect the above solver for the Richards equation and convert the source code of SWAP to Python. This code called PySWAP is extensively treated in the main matter and also in appendix D. This was followed by an extensive phase of testing and improving on the geohydrological model. Especially the coupling with LEM proved to be difficult at times and prone to errors.

Changes to other parts

Apart from the extensive upgrade on the geohydrological modelling, the limit equilibrium method was changed to be able to deal with the suction. Also the non-circular failure surface was applied and the LEM formulation was changed to that of Spencer. After this phase, no large changes have been implemented in the final phase other than improving on the choice of the amount of slices and the degrees of freedom in the differential evolution algorithm. The dam used here was solely consistent of possibly liquefiable material. Any material above the phreatic surface was modelled as being fully drained, so a degree of friction was assigned to this (although lower than in the final phase). Furthermore, the boundary of a degree of saturation of 0.9 was not implemented yet and was used upon later literature research. A degree of saturation of 1.0 was chosen as from which the tailings show a strain-softening behaviour.

B.3.2. Results

For this phase, only a single geometry was used: a dam of 60 meters high with a boundary condition of the phreatic level of 50 meters high. The slope had an angle of 3H:1V and no non-liquefiable shell. All the material was therefore potentially liquefiable. Important was the full study on the influence of the strength which was a combination of the drained friction angle of the tailings and the peak strength ratio. In this phase, a range was used between 20 and 30 degrees for the friction angle. The peak strength ratio was not limited by a specific range. The other parameters are shown in Table B.6. These parameters are largely based on Zhang et al. (2016b) and Robertson et al. (2019).

Table B.6
Values of the different parameters for the strength study for a fully homogeneous tailings dam.

riable	Value
Unit weight	23 $[kN/m^3]$
θ_s	0.45
θ_r	0.039
k_{sat}	0.864
α_{vgm}	2.0
n_{vgm}	1.41

The study had a heavy focus on the return period of failure. It was aimed to find combinations of the drained friction angle and the peak strength ratios which showed reasonable return periods. This meant that the initial factor of safety, without any influence of rainfall, should be larger than 1.1 as to not fail immediately. Furthermore, the strength was not increased anymore if the return period became far larger than 1000 years. Similar to the approach as later applied in analysis RP02, the results were gathered and a contour plot of those results can be seen in Figure B.3.

At first sight, some linear trend is visible. There is quite some variation in the shape of the contour lines especially at the higher values of the return period. It is unsure whether this is an artefact of the standard contour generator of Python. Yet in hindsight, the differential evolution convergence parameters during this phase were not set strict enough. This means that results were variable and some of these return periods are quantitatively not correct. This allows only to focus on some of the qualitative behaviour of the results.

What may be depicted is the narrow boundary between a low return period (< 200 years) and a high return period of failure (> 1000 years). E.g. at a peak strength ratio of 0.39, the difference between these lies within 1.5 degrees for the drained angle of internal friction. Within soil mechanics, this is considered

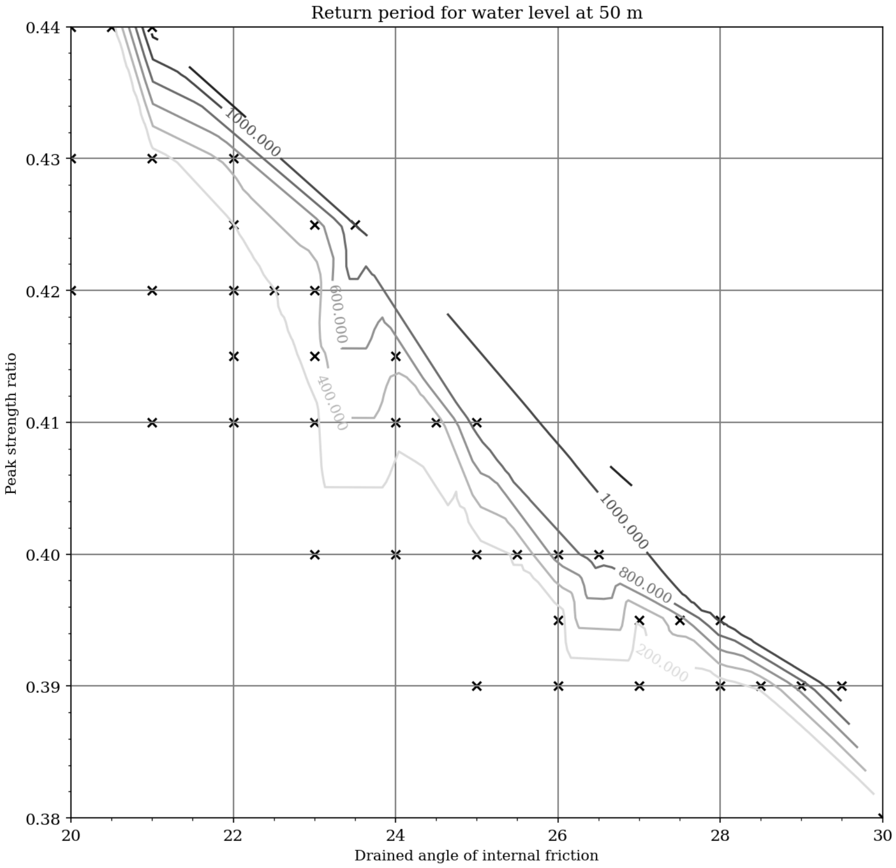


Figure B.3: Contour plot of the return period of the simulations performed. Simulations are marked by an ’x’ in the plot, any values in between are based on linear interpolation between neighbours.

to be a relatively narrow boundary given the uncertainty that these parameters generally hold. There are several reasons as to why the influence of strength is large, or inversely why the sensitivity of the return period to rainfall is low. Figure B.4 shows the slip surface at the start of rainfall and at the moment of the lowest FoS during a certain time span for a random simulation performed during the study. It can be seen that initially, the slip surface tends to form a deep slip surface. This may be attributed to the fact that a dual-strength approach is implemented. The strength above the phreatic surface is far higher than below as above the phreatic surface the material will fail in a drained manner whereas below tailings will fail undrained. The undrained strength is lower due to the generation of the excess pore pressures. Moreover, suction increases the effective stress in the material above the phreatic surface. Within the model, this means that the critical failure surface tends to be deep as to minimise its path through the drained zone and maximise its path through the undrained zone. This in contrast to when only using a single undrained strength as applied in appendix B.2.

Upon rainfall, the suction is reduced and the failure surface tends to move towards the dam surface. However, the influence of the reduced suction is still low due to an increasingly larger portion of the failure surface running through the drained material. It was found that the initial position as well as the shift of the failure surface was influenced by the ratio between the drained strength and the undrained strength that was chosen. If the undrained strength was relatively high compared to the drained strength, the shift would be greater as there is less tendency to stick with the deep slip surface. Within literature, there is no direct relation between the drained and undrained strength of fine-grained materials as the ratio is dependant on e.g. initial void ratio and confining stresses. Bjerrum et al. (1961) shows that the difference between the drained and undrained effective friction angle can be as much as 10 degrees.

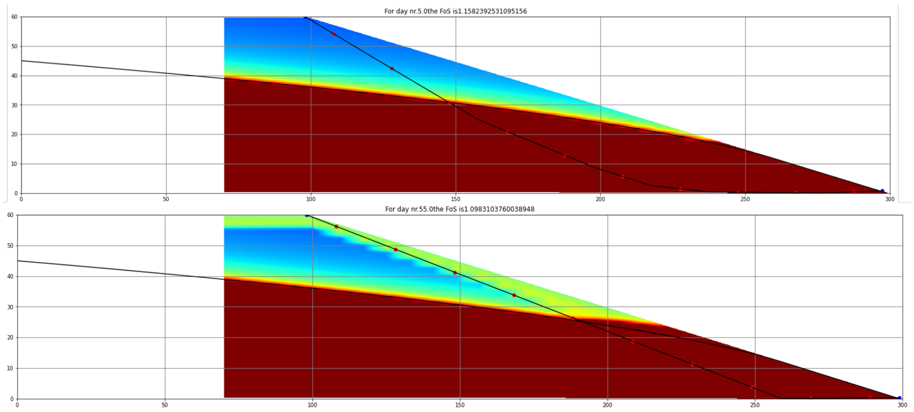


Figure B.4: The slip surface at the start of rainfall, as well as at the moment the lowest FoS is obtained.

The fact that there is the tendency to have deep slip surfaces, has some significant other implications on the results as well. As mentioned already, it was noticed that the FoS is insensitive to rainfall. For a simulation of 5 days, applying the standard geohydrological parameters, a yield strength ratio of 0.38 and a drained friction angle of 28.5 degrees, the change of FoS with increasing intensity can be seen in Figure B.5.

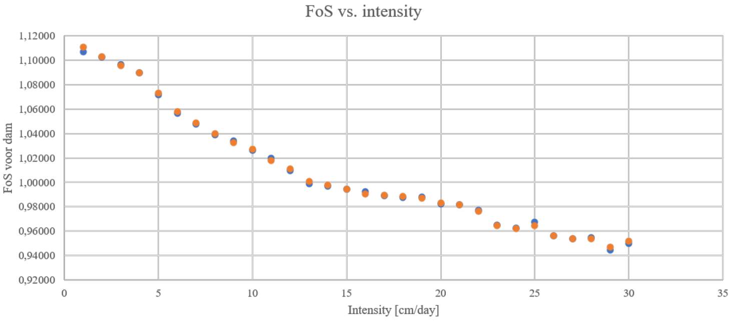


Figure B.5: The lowest FoS encountered for different intensities. Blue and orange dots show same simulation parameters but are different realisations of the stochastic DE.

The reduction with increasing intensity is low. For probable intensities of around 0.1 m/day for 5 days of rainfall, the reduction of the FoS is only in the order of 0.06, which is around 5% of the original FoS. The FoS seems to reduce linearly with increasing intensity which is interesting considering the non-linearity of the problem. The insensitivity is again the result of the deep slip surface and the fact that rainfall has little influence on this. There is the loss of suction along the part of the failure surface running trough the drained material and there is the increase in the phreatic table for the undrained material causing a loss of strength.

In further analysis of the results it was shown that the total rainfall depth is governing. This also explains the linearity of the FoS with increasing intensity, as rainfall depth increases linearly with increasing intensity for the same duration. The results of an example simulation are shown in Table B.7. These rainfall depths are nearly equal. Differences might be accounted for considering the variability in the fact that there is some non-linearity with the sensitivity of the FoS to the intensity. In general however, total rainfall depth seems to be a proper indicator of when failure is to occur.

Another conclusion was that failure will occur a long time after the heavy precipitation event as the wetting front needs to advance to the phreatic surface in order to have some effect on the FoS. This may easily take a month to several months, depending on the geometry and the level of the initial water table.

Table B.7
Resulting intensities from simulation for different lengths of rainfall. Last column shows the multiplication.

Length rainfall [days]	Intensity needed [m/day]	Total depth [m]
1	0.32	0.32
2	0.17	0.34
3	0.11	0.33
4	0.08	0.32
5	0.06	0.30

Furthermore, there seems to be little influence of the geohydrological parameters other than changing the moment at which the lowest FoS is obtained.

Reflecting on recorded case studies of tailings dam failure, there are two that are of especial interest: the failure at Minera Serra, Brazil, 1994 and the failure in Málvesi, France, 2004 (ICOLD Committee on Tailings Dams and Waste Lagoons (1995-2001), 2001). The failure in Brazil mentions the cause to be badly constructed drains and heavy rainfall during late 1993 and early 1994, indicating that precipitation from many months ago may still have influenced the stability. The same is also true for the case in France. Here failure occurred in March 2004, believed to be caused by heavy rain during summer 2003, which is in the order of a few months before failure. These cases highlight the necessity of the wetting front to have advanced deep into the impoundment before failure is to occur.

Other analyses were performed during phase 2 as well, especially concerning the influence of the geohydrological parameters on the return period of failure. Some results were obtained, but as some errors were found later in the analysis procedure those will not be repeated here. Some qualitative information was obtained however. With changing parameters, the change in needed intensity often showed a linear behaviour. This means that the influence on the return period is highly non-linear and shows the same shape as the underlying extreme value distribution. This is something to be considered for later analyses. The hypothesis is formed that the underlying distribution influences the return period significantly.

B.3.3. Recommendations for later analyses

The analysis as explained in this chapter has provided with some interesting results. In comparison to the results as provided in appendix B.2, some significant changes were observed. The largest was the tendency to develop deep slip surfaces which has some severe implications for the sensitivity of the stability (or FoS) to rainfall. This sensitivity was significantly reduced and there was no need to impose deep slip surfaces as was the case for phase 1.

With respect to the modelling of rainfall infiltration, some significant improvements were made. There is one final step to be implemented in order to consider most influences which is modelling the full capillary rise by the use of the air entry value. It is advised to impose this as it will most likely cause larger changes in the FoS due to rainfall as also given in Zandarín et al. (2009). Furthermore, in this phase, the drainage effect was not yet considered and should be implemented as to allow for more accurate results. Reflecting on the main research question, these results almost allow for the answering of it. There is however one significant aspect which is not treated: the slope does not represent a tailings dam. The slope is still homogeneous and there is no clear modelling of the actual shell of the tailings impoundment as was shown in Figure 3.4. In order to shift focus from general slopes to tailings dams it is recommended to implement such a shell of fully drained material in the analysis. As may be clear, these aspects were implemented in the final phase 3 of the analysis which is extensively treated in the main matter.

Table B.8

Specified levels of analysis for the general framework

Part	Level	Description
I: Spatial characterisation	1	Fully homogeneous dam for strength, stiffness and permeability
	2	Two material-type dam with liquefiable and non-liquefiable tailings by the application of a non-liquefiable shell
	3-1	Spatial characterisation based on a coupling with a consolidation analysis for full modelling of changing void ratio and corresponding strength, stiffness and permeability
	3-2	Spatial model based on in-situ data of tailings
II: Consolidation analysis	1	Assumption of fully consolidated tailings dam (no excess pore pressures)
	2	Application of a constant excess pore pressure factor over layer or tailings dam
	3	Analysis of consolidation problem with loading only
	4	Analysis of consolidation by loading with additional influences such as drying/desaturation, freeze-thaw cycles, etc. Examples of such models are: Vardon et al. (2015) and Stark et al. (2005)
III: Geohydrological analysis	1	Rules of thumb for construction of flow net or a parabolic solution to the Laplace equation (Blight, 2010)
	2	Solving Laplace equation with direct recharge modelling of rainfall (Harbaugh, 2005)
	3	Rainfall infiltration with unsaturated flow modelling based on wetting front concept (Green and Ampt, 1911)
	4	Unsaturated flow modelling based on Richards equation - use of capillary rise (Zandarín et al., 2009)
	5	Fully coupled flow-deformation analysis of rainfall infiltration. This uses the Richards equation with a constitutive model for the soil like the Barcelona Basic Model (Alonso et al., 1990)
IV: Stability/liquefaction analysis	1	Undrained strength analysis which applies undrained strength parameters for the modelling of liquefaction based on the in-situ effective stress. (Ladd, 1991; Brett et al., 2012; Brown and Gillani, 2016; Olson, 2001)
	2	Undrained strength analysis with anisotropic strength based on the mode of shearing (Sadrekarimi, 2016)
	3	Numerical stress field calculation with evaluation of the stress state with respect to the instability zone (Chu et al., 2003; Chu and Wantowski, 2012)
	4	Fully coupled liquefaction constitutive models such as Nor-sand (Jefferies, 1993)

APPENDIX C

Miscellaneous analyses

In addition to the results as provided in chapter 6, a few additional studies were performed. These studies supplied with valuable information but did not contribute significantly to the conclusion of this thesis. Or, the analyses were considered to go beyond the limitations that the methodology has imposed. In order to give an overview of the studies that have been done, and allow for the reader to critically reflect on the studies conducted, the results of these analyses will now be shown in this appendix. The following has analyses were performed:

- MA01: numeric analyses of the effect of rainfall infiltration on the slice forces.
- MA02: analysis of the effect of drainage blankets.
- MA03: analysis of the FoS for a large shell width (effective stress analysis).

These will be treated in the following sections.

C.1. MA01

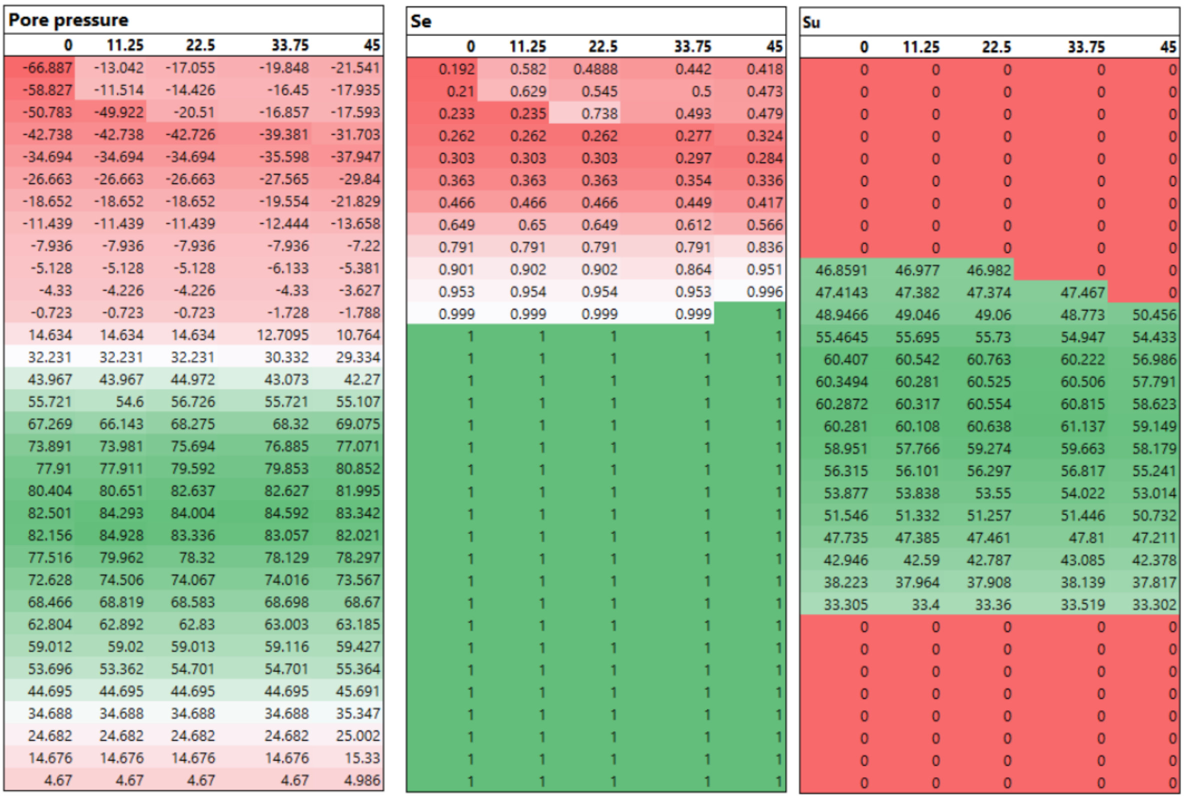
It has been found that the effect of rainfall on the FoS is low. Especially for dams with a large shell width this decrease was minimal. In order to gain more insight into the process and assess why this is the case, a special study was performed. In this study, the slice variables of the final critical failure surface were extracted from Python for a rainfall period of 5 days for three different intensities: 0.05, 0.1 and 0.2 m/day. The results were evaluated at 0, 11.25, 22.5, 33.75, and 45 days after the start of rainfall. The various other variables are shown in Table C.1.

Table C.1
Analysis input parameters of MA01

Goal	Numerical verification of the FoS for a large shell width dam
Geometric parameters	Height of 20 meters; slope of 3H:1V; shell width = 25 m
Strength parameters	$\varphi'_{liquefiable} = 40°$; $\varphi'_{non-liquefiable} = 35°$; $s_u/\sigma'_v = 0.27$
Geohydrologic parameters	$\theta_s = 0.4$; $\theta_r = 0.05$; $k_{sat} = 0.1$ m/day; $n_{mvg} = 1.7$;$\alpha_{mvg} =$ 2.0 1/m; $AEV = 0.5$ m

The data has been processed in Excel to provide with a clear overview and will be presented here as figures. This allows for the use of colour scaling for additional clarity on the different numbers. In Figure C.1, the pore pressure, relative degree of saturation and undrained shear strength are given for an intensity of 0.05 m/day.

What can be seen in the top of the provided tables over the days is the wetting front passing. It is visible that the wetting front only reaches the top 4 slices. It must be highlighted that this slice formulation did not yet introduce more slices to the top part as was done later. The wetting front is clearly a passing phenomena as somewhere between 11.25 and 22.5 days, the FoS starts to increase again. This can also be seen in the degree of saturation. It is however noted that the highest degree of saturation is



Pore pressure				
0	11.25	22.5	33.75	45
-66.887	-13.042	-17.055	-19.848	-21.541
-58.827	-11.514	-14.426	-16.45	-17.935
-50.783	-49.922	-20.51	-16.857	-17.593
-42.738	-42.738	-42.726	-39.381	-31.703
-34.694	-34.694	-34.694	-35.598	-37.947
-26.663	-26.663	-26.663	-27.565	-29.84
-18.652	-18.652	-18.652	-19.554	-21.829
-11.439	-11.439	-11.439	-12.444	-13.658
-7.936	-7.936	-7.936	-7.936	-7.22
-5.128	-5.128	-5.128	-6.133	-5.381
-4.33	-4.226	-4.226	-4.33	-3.627
-0.723	-0.723	-0.723	-1.728	-1.788
14.634	14.634	14.634	12.7095	10.764
32.231	32.231	32.231	30.332	29.334
43.967	43.967	44.972	43.073	42.27
55.721	54.6	56.726	55.721	55.107
67.269	66.143	68.275	68.32	69.075
73.891	73.981	75.694	76.885	77.071
77.91	77.911	79.592	79.853	80.852
80.404	80.651	82.637	82.627	81.995
82.501	84.293	84.004	84.592	83.342
82.156	84.928	83.336	83.057	82.021
77.516	79.962	78.32	78.129	78.297
72.628	74.506	74.067	74.016	73.567
68.466	68.819	68.583	68.698	68.67
62.804	62.892	62.83	63.003	63.185
59.012	59.02	59.013	59.116	59.427
53.696	53.362	54.701	54.701	55.364
44.695	44.695	44.695	44.695	45.691
34.688	34.688	34.688	34.688	35.347
24.682	24.682	24.682	24.682	25.002
14.676	14.676	14.676	14.676	15.33
4.67	4.67	4.67	4.67	4.986

Se				
0	11.25	22.5	33.75	45
0.192	0.582	0.4888	0.442	0.418
0.21	0.629	0.545	0.5	0.473
0.233	0.235	0.738	0.493	0.479
0.262	0.262	0.262	0.277	0.324
0.303	0.303	0.303	0.297	0.284
0.363	0.363	0.363	0.354	0.336
0.466	0.466	0.466	0.449	0.417
0.649	0.65	0.649	0.612	0.566
0.791	0.791	0.791	0.791	0.836
0.901	0.902	0.902	0.864	0.951
0.953	0.954	0.954	0.953	0.996
0.999	0.999	0.999	0.999	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1

Su				
0	11.25	22.5	33.75	45
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
46.8591	46.977	46.982	0	0
47.4143	47.382	47.374	47.467	0
48.9466	49.046	49.06	48.773	50.456
55.4645	55.695	55.73	54.947	54.433
60.407	60.542	60.763	60.222	56.986
60.3494	60.281	60.525	60.506	57.791
60.2872	60.317	60.554	60.815	58.623
60.281	60.108	60.638	61.137	59.149
58.951	57.766	59.274	59.663	58.179
56.315	56.101	56.297	56.817	55.241
53.877	53.838	53.55	54.022	53.014
51.546	51.332	51.257	51.446	50.732
47.735	47.385	47.461	47.81	47.211
42.946	42.59	42.787	43.085	42.378
38.223	37.964	37.908	38.139	37.817
33.305	33.4	33.36	33.519	33.302
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0

Figure C.1: The pore pressure, relative degree of saturation and undrained shear strength for the slice midpoints over time for an intensity of 0.05 m/day.

not recorded just after rainfall, but at the 22.5 days mark. It is not entirely sure why this is the case, but it is mostly attributed to the interpolation that is performed over the grid. It may also be because of the different failure surface at this moment and the DE algorithm getting stuck in a local minima. As the error is relatively small, this was not further investigated. Clearly, the decrease in strength because of a loss of suction is only minimal due to the fact that the slip surface is deep. The model is also not able to catch the effect of the rise of the phreatic surface well. There is a small decrease in the pore pressure over time for some nodes, but these may also be attributed to a shift in the failure surface.

This becomes more evident when looking at the development of the undrained shear strength over time as also provided in Figure C.1. What can be seen is that actually less nodes are being modelled as undrained which may only be explained when the results are compared to the actual position of the critical failure surface. The failure surfaces at the start of rainfall and at the end of the simulation period are shown in Figure C.2.

The shell has severely affected the position of the failure surface. The wetting front has not yet reached the phreatic surface. Hence the failure surface has only minimally shifted to take full advantage of the wetting front passing. Since the wetting front has not diffused yet, the failure surface has only slightly shifted to the right. Less nodes will therefore be modelled as undrained. The shift is only small, but since there will always be a slice imposed at the 90% degree saturation line in the code, some nodes have turned into a drained state.

It is visible that loss of suction at the first days after rainfall will be governing. The wetting front passes so slowly that the water content has fully diffused when the wetting front hits the phreatic surface. As recharge will be slow, a mixed mechanism cannot be initiated. In addition, any recharge to the water

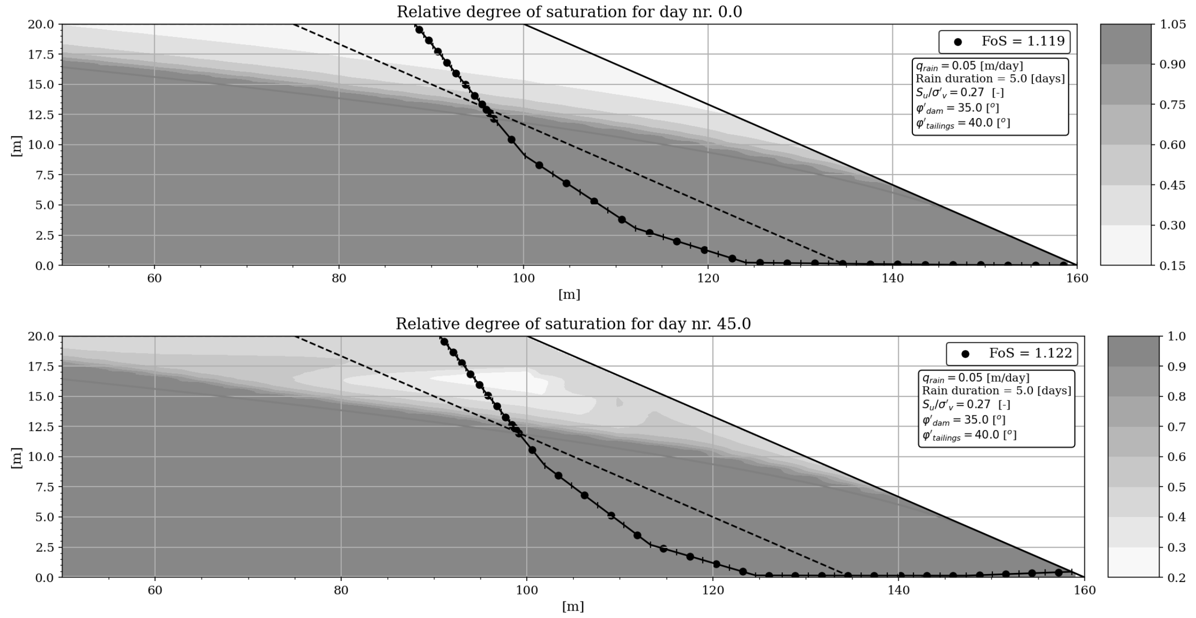


Figure C.2: The failure surfaces at the start of rainfall and at the end of the simulation period at 45 days.

table will not turn the soil into a potentially liquefiable soil. This in contrast to the smaller shell widths were this was the case. Also for the other intensities of 0.1 and 0.2 m/day, the effect of rainfall was only minimal. The mixed mechanism cannot develop due to the large shell width and any recharge to the water table will not lead to a significant weakening. Based on this numeric example it may be proven that the width of the drained/dilative zone of tailings is essential in its stability and sensitivity to rainfall. This has however also been concluded in analysis GA01.

C.2. MA02

Analysis MA02 considered the effect of a drainage blanket. Following the design guidelines as mentioned in section 3.1, it is well known within industry that having a seepage front of the phreatic surface near the toe will cause instability of the tailings dam (Klohn, 1979). In order to avoid a high phreatic surface near the toe, drainage blankets are installed beneath the starter dam. These drainage blankets cause a significant lowering of the phreatic surface and the degree of saturation near the toe significantly. Although these drainage blankets are not always properly maintained and become clogged, it is however worth studying the effects of these blankets with respect to the FoS when considering rainfall infiltration. In this line of thought, a short analysis was performed to estimate these effects. The drainage blanket was mimicked by setting the boundary condition of a head of 0 meter around 20 meters left of the toe. It is unknown whether this is a correct representation of such a drainage blanket. Furthermore it is unsure what the degree of saturation is above the blanket and whether concepts like the SWCC and the AEV still hold above the blanket. This was however assumed in the analysis and effectively the phreatic head was reduced to 0 meters for the first 20 meters upstream of the toe. The analysis was performed for a rainfall intensity of 0.1 m/day for a period of 5 days. The strength combination applied was 0.37-30-40. The shell width was arbitrarily set at 10 meters. The results of the simulation by normalised FoS and that of a comparison analysis with normal BC can be seen in Figure C.3.

The slope with the drainage blanket is influenced in the same manner during the first few days as the impoundment with a regular imposed phreatic surface and actually shows a lower normalised FoS than

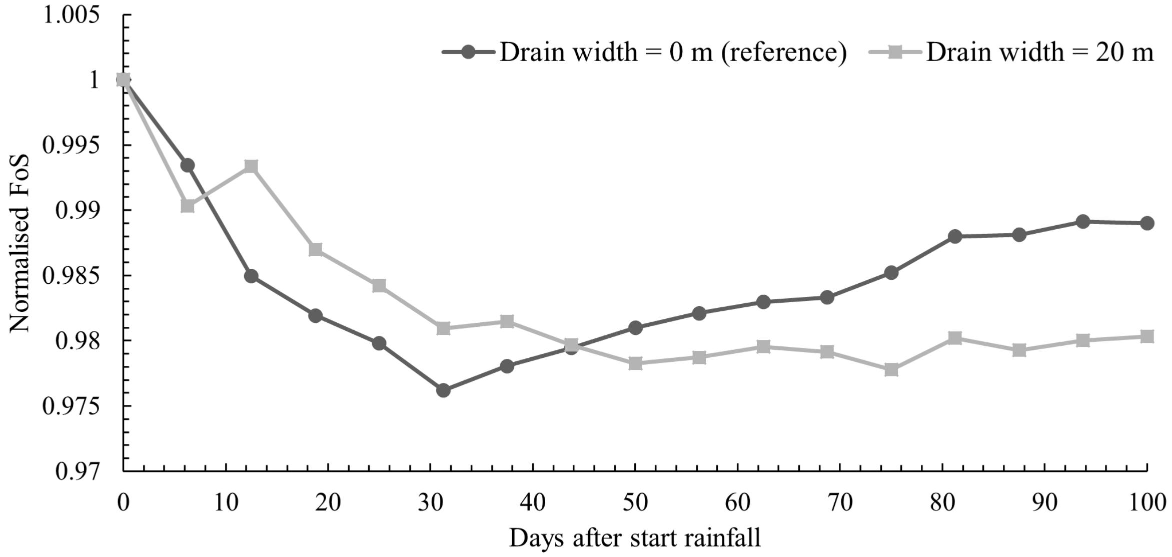


Figure C.3: The normalised FoS for the simulation of a drainage blanket of 20 meters at the toe of the impoundment. Also shown is the reference study in which such a blanket was not modelled.

for the reference case indicating the effect of rainfall is larger for such dams. However starting from 10 days onwards, the FoS is higher. Although this is not by a significant amount. The reference case has its lowest FoS at around 30 days. For the case with the drainage blanket however, the FoS keeps decreasing and will have a low value between 70 and 80 days. It must however be noted that the FoS tends to stay low after 50 days after the start of rainfall and that the change in FoS can be attributed to the accuracy of the DE algorithm. It is of interest to see that it takes long for the FoS to recover and the FoS is only minimal after a period of 100 days. In addition, the lowest FoS recorded has nearly the same normalised value as for the reference case. The wetting front and corresponding failure surface are shown in Figure C.4.

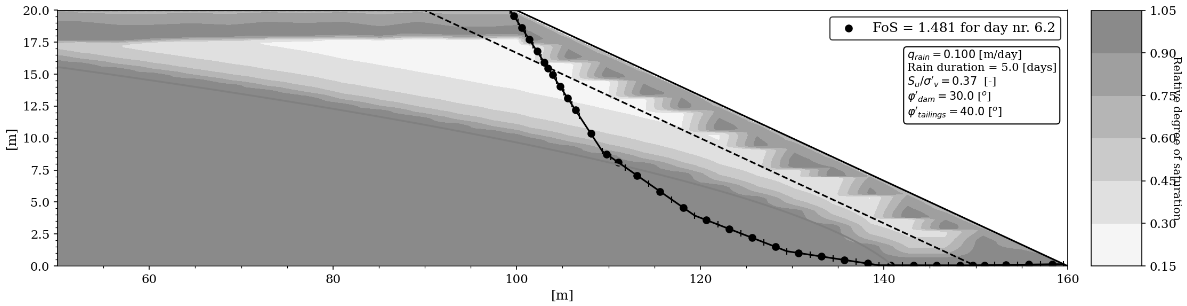


Figure C.4: The wetting front of the tailings impoundment with drainage blanket after the end of rainfall.

The wetting front behaves the same way as was seen in earlier analyses that have been treated in section 6.1.2. There is however a large difference in the behaviour near the toe. What can be seen is that for the part near the toe, the phreatic surface shoots up shortly after rainfall infiltration. This water will quickly drain away, but there is a moment where the phreatic surface is high. This causes an instantaneous decrease of strength over this part near the toe as at the bottom the pore pressure was originally zero and now has a positive pressure head equal to the slope height. This effect is not possible for the reference case as then the part near the toe is fully saturated already during regular simulations. It is however highlighted that the initial FoS because of the drainage blanket is far higher than for the reference case. The FoS at the start of rainfall for the reference case is 1.18, whereas it is 1.50 for the case with the drainage blanket. The blanket therefore severely increases the safety of the impoundment and may therefore be regarded effective. However, the sensitivity to rainfall does not decrease.

The lowest recorded FoS is at day 75 after the start of rainfall. The wetting front and failure surface are shown in Figure C.5.

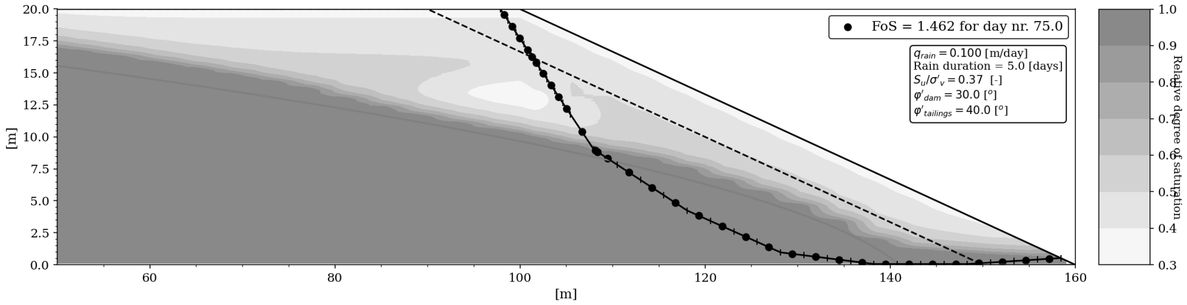


Figure C.5: The degree of saturation and corresponding failure surface for the lowest FoS in the drainage blanket simulation.

Essentially the decrease in the FoS is solely due to a loss of suction. This effect has spread over the entire unsaturated zone. As the wetting front has to advance deep into the impoundment, diffusion has caused the wetting front to be spread and hence recharge to the water table is low. Drainage will therefore be governing and any mounding of the water table will not occur.

In conclusion, drainage blankets have shown to be effective in the increase of the FoS for a given impoundment in contrast to the situation where seepage at the toe occurs. However, the sensitivity to rainfall is the same for both cases. With a drainage blanket, the initial drop in FoS is significant, due to the toe being affected by rainfall. The lowest FoS will occur due to a loss of suction above the phreatic surface. Mounding of the water table is not possible, as the highest possible drainage rate outperforms the occurring recharge rate.

C.3. MA03

In analysis MA03, the effect of rainfall on tailings dams with a large shell width was investigated. This analysis may truly be regarded as a side-step study. The shell width was set to a large value, essentially to investigate the drop in FoS for a full effective stress approach of the non-liquefiable shell. The parameter set applied in this analysis can be seen in Table C.2. The study was performed for an intensity of 0.1 m/day for a duration of 5 days.

Table C.2
Analysis input parameters of MA03

Goal	Analysis of the FoS for a large shell width and hence full effective stress analysis.
Geometric parameters	Height of 20 meters; slope of 17 degrees; shell width = 100 m
Strength parameters	$\varphi'_{liquefiable} = 35°$; $\varphi'_{non-liquefiable} = 32°$; $s_u/\sigma'_v = 0.32$
Geohydrologic parameters	$\theta_s = 0.4$; $\theta_r = 0.05$; $k_{sat} = 0.1$ m/day; $n_{mvg} = 1.7$;$\alpha_{mvg} = 2.0$ 1/m; $AEV = 0.5$ m

The calculated factors of safety are given in Figure C.6. The simulation was run for only 20 days as applying this ESA causes the factor of safety to be low shortly after the start of rainfall. Upon an earlier simulation done for a simulation period of 100 days, it was decided to reduce this time significantly in order to obtain results at a closer spaced interval for a period of 20 days.

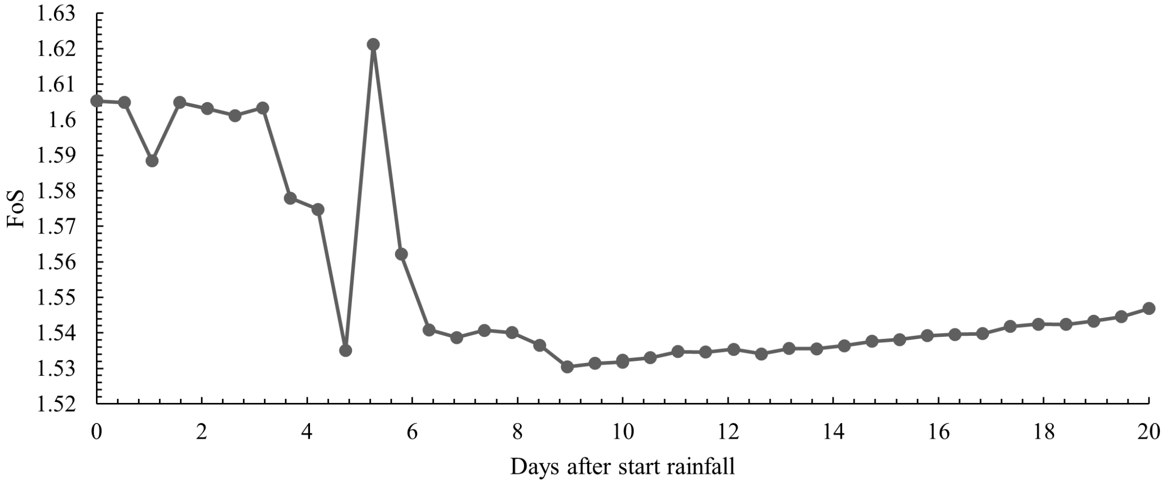


Figure C.6: The FoS for an ESA analysis of the outer shell for a period of 20 days.

The starting FoS is high and equal to a value of 1.61 at the start of rainfall. This is far higher than in the undrained strength analysis that was performed in the main matter. Where, for this combination of phreatic surface, height, strength parameters, etc. a value of around 1 is to be expected. This means that performing an ESA will substantially increase the FoS. Although here only the shell is modelled and one has to be cautious in extrapolating these results to say something about the analysis procedure. The FoS is stable for a number of days and then significantly drops at around 5 days after the start of rainfall. A huge spike is observed just after and followed by a new decrease in FoS. The FoS is lowest at a period of around 9 days where a value of approximately 1.53 is shown. The FoS is hence reduced with 5%. This is a little bit more than in the undrained analysis. Hence, in an ESA the effect of rainfall is larger. Although this effect is only marginally increased.

In view of the observations done, three time instances at which the FoS is calculated will now be reviewed in further detail. This is at the start of rainfall, at the moment the spike occurs and at the moment of lowest FoS. These will now be treated subsequently.

The degree of saturation over the impoundment as well as the critical failure surface at the start of rainfall can be seen in Figure C.7.

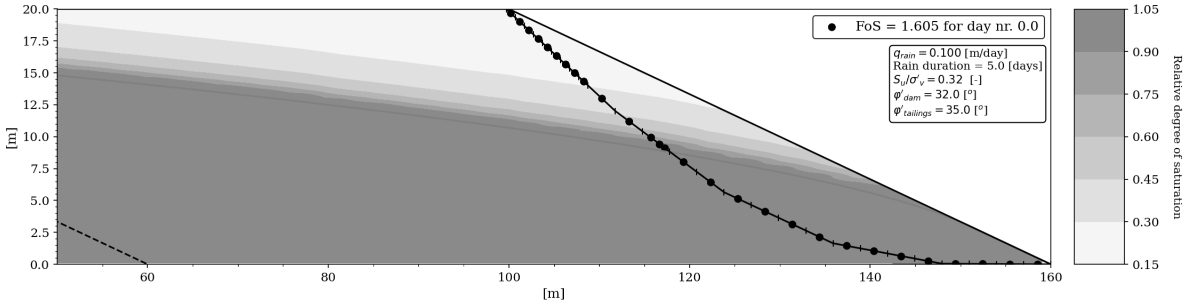


Figure C.7: Failure surface at the start of rainfall for the analysis with the large shell width

Due to the set range on the entry variable, the slip surface is only allowed to start at the top of the impoundment. What can be seen is that there is still the tendency to develop a deeper slip surface. This fact is attributed to the phreatic surface and the resulting suction above the groundwater table. Because of the higher suction, the path through it is minimised and hence some deeper slip surface is generated. When the suction would not be considered, or the phreatic surface will be at zero, then possibly a slip surface parallel to the slope is formed.

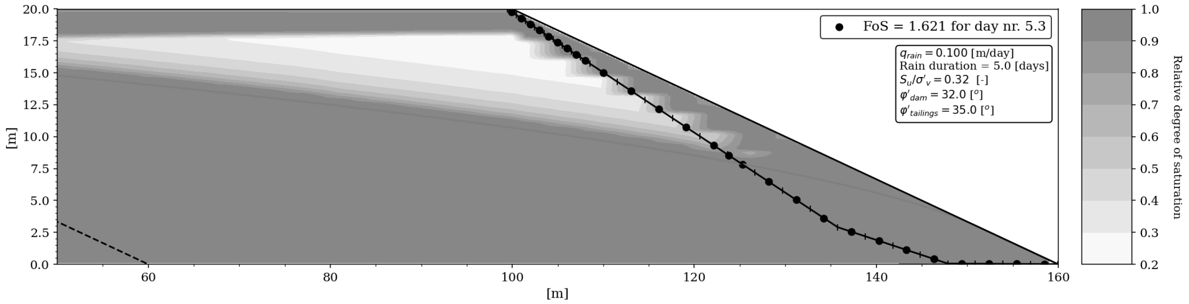


Figure C.8: The degree of saturation and corresponding failure surface for the observed spike in the FoS.

Then moving on to the spike observed earlier, the degree of saturation and corresponding failure surface are shown in Figure C.8. What may first be observed is that with reduction of suction along the slope, the failure surface tends to go towards a slip surface that is parallel to the slope. There are no certain aspects to be observed that explain this oddly high FoS. When analysing failure surfaces just before and after, it is assumed that there is no physical effect in this and the DE algorithm got stuck in a local optimum.

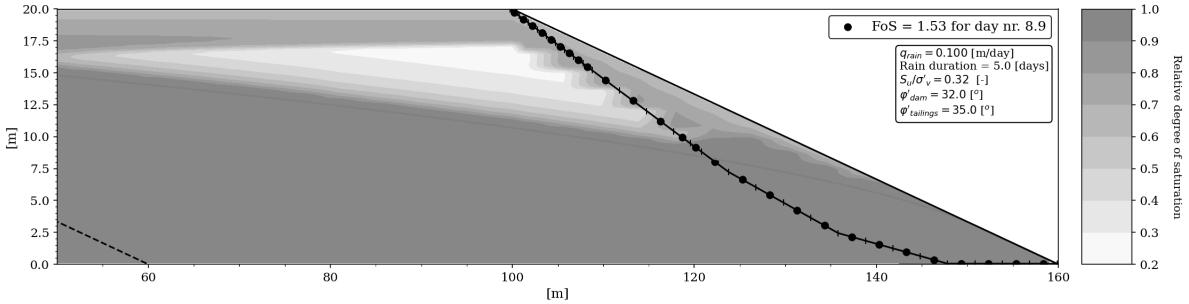


Figure C.9: The degree of saturation and corresponding failure surface for the lowest FoS in the analysis of the larger shell width.

This leaves with the analysis of the lowest FoS. The corresponding failure surface is displayed in Figure C.9. What may be apparent is that the mixed mechanism is not only governing in an USA, but in this case also in the ESA. Clearly, the failure surface seems to connect a loss of suction as well as the rise in the phreatic surface and again a deeper slip surface is obtained. It has been observed that the slip surface does not run exactly through that part for which the rise of the phreatic surface is higher. Manually, a slip surface was attempted to run through that, but because of increasing suction after passage of the wetting front, this did not yield a lower FoS. Furthermore, along a larger stretch of the failure surface the strength is now reduced, rather than only at the first part passing the phreatic surface. In an USA additional material would become liquefiable here. However, this is not the case in this effective stress approach.

A conclusion would be that the effect of rainfall is increased when applying an ESA in contrast to the USA for the tailings dam. There is the tendency to form slightly shallower slip surfaces, but due to the phreatic surface, some depth in the slip surface is still maintained. Even in an ESA analysis, the governing failure mechanism is a mixed mechanism of loss of suction in the unsaturated zone as well as a loss of strength due to mounding of the water table.

APPENDIX D

Source code: documentation and elaboration

This appendix will elaborate on the Python code as was implemented in the final version for the calculation of the return period of failure. Only the final code will be presented. A lot of different modules have been coded as well in the process but these will not be treated here. Only the code presented in this appendix is needed to obtain similar results as provided in chapter 6, apart from some plotting statements which have been commented or deleted.

Apart from providing the source code, special explanation will be provided to allow the reader to understand the code and why certain aspects were implemented. The focus therefore also lies with the computational behaviour of the code. The following parts/libraries will be treated:

- Main Code: this script will need to be run to calculate the return period of failure for different parameter sets. All functions needed are imported from libraries.
- Flopy/Modflow initial conditions: the initial conditions for the geohydrological model are based on a solution to the Laplace equation which is obtained by the functions in this library.
- Unsaturated flow analysis: implementation of PySWAP. It consists of two libraries where one is the main program PySWAP and the other holds the functions needed for PySWAP.
- Slope stability analysis: the library for the differential evolution critical failure surface.
- Gumbel analysis of precipitation data: library for obtaining the parameters of the Gumbel and GEV fits on the precipitation data.

D.1. Main code

The main code is the single script which addresses the various functions as applied in the different libraries. Its goal is to: import needed external Python libraries; initialise the needed parameters as specified by the user; modification of these parameters as to allow them as input for the different functions; perform a loop over the intensity to find the intensity for which the critical FoS of 1.1 is achieved. These different parts will now be treated.

So, at first the needed Python libraries are loaded. Table D.1 mentions the use of each of the different libraries. Also the other libraries needed for analysis are loaded, and the standard font of the Python plots is changed to Times New Roman in this part.

```
    #%% Importing libraries and packages

#Importing time package
import time
ticall = time.perf_counter()

#Import basic Python packages
```

Table D.1

Needed Python libraries for use of the main code and libraries

ibrary	Used for:
Numpy	Various array modifications and passing of values
pyplot	plotting of various (intermediary results)
math	mathematical manipulation of values or equations
griddata	function needed to get a 2D-grid from the column results of PySWAP (Quasi 2-D approach)
namedtuple	collection item in Python to pass a series of values by a single name to and from functions
cm, Listedcolormap	alteration of colourmaps for plotting
cprofile,re	allow for speed checks within the code. Provides list of needed computational time for each function
rcParams	change of standard fonts for pyplot plots

```python
import numpy as np
import matplotlib.pyplot as plt
import math as mt
from scipy.interpolate import griddata
from collections import namedtuple
from matplotlib import cm
from matplotlib.colors import ListedColormap


#Import the made libraries
import Modflow_library as Mdfl              # library for the initial conditions

import FunctionLibraryVDE_V13 as fl         # library for the DE slip surface finder
import UnsaturatedSWAP as swap              # libary to initiate the PySWAP module

#additional packages for plotting and changing standard font within plots
from matplotlib import rcParams
rcParams['font.family'] = 'serif'
rcParams['font.sans-serif'] = ['Times New Roman']
from matplotlib import rc
rc('xtick', labelsize = 11)
rc('ytick', labelsize = 11)

#Python libaries to evaluate the computational efficiency of the written code
import cProfile
import re
```

After that, the user may enter the various parameters. These are divided into different sections: geometry and strength, time parameters, unsaturated flow parameters, modflow parameters and slip surface parameters. These parameters have already been treated extensively in chapter 5. There are however some additional ones which have not been mentioned before. Their use is provided in Table D.2. It is advised to the user to not change these values unless really necessary.

Table D.2

Additionally needed parameters for analysis not treated in chapter 5

Variable	Use
layerData	artefact from earlier versions of the main code. Serves no use, but is rather embedded in the code.
tEndFlowArray	array of rainfall lengths to be treated. In this case from 1 to 6 days, not including 6.
magX	magnification factor for the 2D grid generation of the pressure head and the relative degree of saturation. Effectively every 0.1 meters a value is now defined, i.e. 10 values per meter.
sy	specific yield for the Modflow analysis (no longer needed, but applied in phase 1).
InitC	Initial conditions value for Modflow analysis. The value of 12 works fine if the BC is at 20. However, sometimes varying the parameter is needed as to assure convergence.
axisPoint	Coordinates of the point around which the moment balance in the Spencer LE formulation is calculated.

```
#------------------Geometry and strength--------------------

topPoints = np.array([[0,20],[100,20],[160,0]]) # Geometry of the impoundment

layerData = np.array(0) # artefact of earlier versions, may be removed

# tuple for the liquefiable layer material properties
LLProp = namedtuple('LL', ['gammaWet','SuSigmaV', 'phiSuc'])

# tuple for the drained layer material properties
DLProp = namedtuple('DL', ['gammaWet','phiDrained', 'damWidth'])

# here the user may provide values for both tuples. gammawet is the variable for the
# material unit weight. SuSigmaV the peak strength ratio. phiSuc is the friction angle
# to be applied for the liquefiable tailings. phiDrained is the friction angle to be used
# for the shell. damWidth is the same as the width of the drained shell.
LL = LLProp(gammaWet = 23., SuSigmaV = 0.27, phiSuc = 40.)
DL = DLProp(phiDrained = 35., damWidth = 25.0)

#-------------------Time parameters --------------------------------

tEndFlowArray = np.arange(1,6)  # An array of the durations to be evaluated in analysis.
                                # In this case, that is 1-5 days

#-------------------Unsaturated flow parameters---------------------
#grid parameters
magX = 10   # parameter for the magnfication factor applied in the interpolation


#Geohydrological parameters
nNode = 100     # number of nodes/cells for a column in PySWAP

thetas = 0.4    # Porosity [-]
thetar = 0.05   # Residual water content [-]
kSat = 0.1      # Saturated hydraulic conductivity [-]
alpha = 3.0     # Mualem-van Genuchten equation parameter
n = 1.7         # Mualem-van Genuchten equation parameter
m = 1-1/n       # Mualem-van Genuchten equation parameter based on n
```

```
h_enpr = -0.5   # Air entry value


FlSucMax = False    # Flag if the hydrostatic condition of suction above the phreatic surface
                    # is bound to a maximum
sucMax = -7.5       # If FlSucMax == True, then this is the maximum suction allowed.

# Assembly of the dictionary for the Mualem-van Genuchten equation.
mvgn = {'thetas':thetas,
            'thetar':thetar,
            'kSat': kSat,
            'alpha': alpha,
            'n': n,
            'm': m,
            'h_enpr': h_enpr}

#-------------Modflow model----------------------------------------------
modelName = 'SimRound'  # File name which will be given to the Modflow calculation

# tuple for the geohydrological parameters in the Modflow calculation.
GHProp = namedtuple('GH', ['hk', 'vka', 'InitC', 'Sy', 'bndHead'])

anisotropy = 5          # Ratio of horizontal over vertical permeability
vka = kSat              # Vertical permeability
hk = anisotropy*vka     # Horizontal permeability calculation
kh = hk
Sy = 0.15               # Specific yield (only of value in direct recharge model)
bndHead = 20.           # BC of the head at upstream side of impoundment
InitC = 12.             # Initial condition to be used in Modflow calculation


GH = GHProp(hk = hk, vka = vka, InitC = InitC, Sy = Sy, bndHead = bndHead)

#-------------------Slip surface parameters--------------------------

#parameters for non-circular slip surface
xStartRange = [50,topPoints[1,0]]   # Start range for the slip surface (x-coordinate)
xEndRange = [140,topPoints[-1,0]]   # End range for the slip surface (x-coordinate)
axisPoint = np.array([100.,50.])    # Point of calculation for moment balance
noSlicesNonCirc = 5                 # Number of slices for non-circular failure surface
                                    # Actually the number of degrees of freedom

#Differential evolution parameters
popSize = 18            # Population size in DE algorithm
MPoolSize = popSize
numGenerations =  700   # Maximum number of generations allowed in DE analysis
```

Now the initial conditions for the geohydrological calculations in PySWAP can be set up. This is done by running the function in the Modflow library. Running this function is initialised by the following code snippet. A function of the DE library is borrowed (slopeData) to calculate the height of the various columns as need to be passed to Modflow.

```
# Transforming the geometry into a set of lines with associated mathematical variables
# being the slope and the y-intercept
slopeData = fl.FuncSlopeData(topPoints)

# Rearranging the slope information to columns to be applied in the modflow analysis
# A column has a width of 1 meter and height associated with the slope geometry
xTopModflow = np.arange(0,topPoints[-1,0]+1)
zTopModFlow = np.zeros((1,len(xTopModflow)))
```

```
for i in range(len(xTopModflow)):
    idxMF = int(xTopModflow[i])
    section = fl.FuncSectionFinder(slopeData,idxMF)
    zTopModFlow[0,i] = xTopModflow[i]*slopeData[section,0]+slopeData[section,1]

#Running of the Modflow calculation. The input is the given geometry and a set of
#geohydrological variables. The output is the head per column and the final calculation
#report.
head,mfOutput = Mdfl.ModFlow(modelName, topPoints,GH,zTopModFlow)
```

After the initial conditions have been established, further preparation is done for the unsaturated flow simulations. First, some of the variables are printed to the Python kernel as to show the user the various input variables. This is of convenience when running multiple analyses simultaneously. This can be done by starting various kernels. In order to avoid slowing down of these kernels, in the taskmanager it is advised to set the priority of these kernels to ‘realtime’. The computer may then freeze when doing other extensive work but will mostly unfreeze within a couple of seconds. It however avoids severe delays in the simulations.

For the preparatory works, the x-positions of the columns are determined for calculation. This is done over a linear space ranging from 40 meters to the toe of the impoundment. Thirty columns are initiated in this case. Then an additional column is appended which considers exactly the top of the slope. This will avoid any numerical inconsistencies for the slip surface. The other variables *zRef, gwlIni* and *hCol* are needed for the unsaturated flow simulation. *zRef* will provide with the depth of the groundwater table. This is essentially the height of the slope minus the head as calculated in Modflow. *gwlIni* is a copy of that for numerical purposes. *hCol* is the height of the column.

This part is concluded with the initiation of the grid for the 2D interpolation and the changing of the colormap for plotting purposes.

```
# Printing of variables to be displayed in the Python kernel as user information
print(LL)
print(DL)
print('Thetas is,', thetas)
print('Thetar is,', thetar)
print('ksat is,', kSat)
print('alpha is,', alpha)
print('n is', n)
print('bndHead is ', bndHead)

# Intialising the arrays for the initial conditions in the unsaturated flow model
xHead1 = np.linspace(40., topPoints[-1,0]-1,30) # Position of columns for calculation
xHead2 = np.array(topPoints[1,0]) # Additional column at outer tip of slope
xHead = np.unique(np.sort(np.hstack((xHead1,xHead2)))) # Assembly of upper two lines
zRef = np.zeros(len(xHead)) # Depth of the water table
gwlIni = np.zeros(len(zRef)) # Like zRef, but used for analysis
hCol = np.zeros(len(zRef)) # Height of the column
ySlope = np.zeros(len(xHead)) # height of the slope at position x
delh = np.zeros(len(zRef)) # cell thickness

for i in range(len(xHead)):
    idx = int(mt.ceil((xHead[i])))
    section = fl.FuncSectionFinder(slopeData,idx)
    ySlope[i] = xHead[i]*slopeData[section,0]+slopeData[section,1]
    zRef[i] = head[0,idx]-ySlope[i]

    gwlIni[i] = zRef[i]

    hCol[i] = ySlope[i]
    delh[i] = hCol[i]/(nNode)
```

```
# The intiation of the meshgrid to be used later in the interpolation of the 1D data
# onto the 2D-grid. Note that magX parameter is used to refine the grid
x= np.linspace(0,topPoints[-1,0],topPoints[-1,0]*magX)
y = np.linspace(0,topPoints[0,-1],topPoints[0,-1]*magX)
grid_x, grid_y = np.meshgrid(x,y)

# Changing the colormap for plotting. The binary colormap is used but limited so as to
# only appy a grey-scale

binary = cm.get_cmap('binary',512)
newcmp = ListedColormap(binary(np.linspace(0.,0.5,256)))
```

The next section of the main code consists of two functions which need to be performed and are an integral part of the calculation. To minimise the amount of lines in the main code, in future implementations this could be moved to another library as this will result in cleaner coding. However, for various different simulations, sometimes adjustments have to be made to these functions. That will however only be for expert users.

The first function *gridXY* is a function needed to establish two arrays. These arrays are the coordinates within the grid on which the various state parameters are calculated. Basically, these are the coordinates of the nodes of the various columns in the quasi-2D approach. The function *IntensityCalculator* is truly the main function of the programme. It will perform the unsaturated flow calculation for the different columns. This is followed by an interpolation of the data over the 2D-grid. Then the FoS calculations are performed and the minimum FoS is returned. Lastly, plots are made of the degree of saturation and the resulting failure surface.

When zooming in on the code itself, first a loop is initiated over the various columns. When the depth of the water table is less than 1 meter deep, the unsaturated flow calculation will not be performed. This because it will lead to a nearly insolvable differential equation and it will slow down computation significantly, while only a minimal gain is obtained in quality. If the water depth is below this one meter, the calculation will be performed. Depending on whether the column calculation is performed for a column within the slope or along the flat part of the impoundment, different ponding heights are available. Here it is set to 0.1 and 0.5 meters respectively. Calculations have shown however that ponding never tends to exceed 0.05 meters because of the larger permeability of tailings compared to e.g. clays and the fact that no antecedent rainfall is considered. The calculation is then performed and the various variables are appended to a large horizontal array. If the groundwater level is not at sufficient depth, full saturation is assumed over the column height and a hydrostatic pressure distribution is applied.

After that the interpolation can take place for each time step. Since different columns are used and no event time discretisation is applied in PySWAP yet, the solution time steps are different for each simulation. Therefore, a loop has to be constructed that for each column applies the correct time step. It is important to note therefore that between each column, the exact time step of calculation may differ. This is however hardly of influence to the results. The correct index is retrieved and applied to the arrays which hold the various solutions for all of the columns. Then using the embedded scipy function *griddata*, the data is interpolated over a preassembled 2D grid. This provides with an interpolated grid of the degree of saturation as well as the head.

A small loop is then started over the array elements, that will reduce any value to NaN if it is above the slope. This loop was imposed after studying a similar geometry like the Feijão dam in which there is a set-back of the tailings dam. Applying the linear interpolation mechanism, values were interpolated along sections where there is no slope. The resetting of the interpolation scheme is followed by calculation of the factor of safety. As all necessary code is implemented in a library this takes just a single line. The FoS of each time step is archived into an array *FoSt*. Finally, plots can be made to show the development of the wetting front and failure surface over time. The last part of the code consists of a small loop that checks whether the FoS of a new time step is actually lower than the lowest recorded till then. If this is not the case for two times in a row, then the code breaks out of the loop as any additional computations are useless.

```
def gridXY(xHead, nIN, ySlope):
'''For the interpolation scheme, all datapoints at which the head and relative degree of
saturation are to be evaluated are calculated here. This essentially comprises all
x- and y-coordinates of the nodes within the finite difference cells '''
    phreaticX = np.array(())
    phreaticY = np.array(())
    for i in range(len(xHead)):
        phreaticX = np.hstack((phreaticX,np.array(np.ones(nIN-1)*xHead[i])))
        phreaticY = np.hstack((phreaticY,np.array(np.linspace(ySlope[i],0,num = nIN-1))))
    return phreaticX, phreaticY

#Implementing the various functions
def IntensityCalculator(qRain,tOut, tEnd,xHead, nNode, hCol, tRain, gwlIni,
                        FlSucMax,sucMax,kh, ySlope, grid_x,grid_y,topPoints,
                        noSlices,layerData,LL,DL,xStartRange,xEndRange,magR,
                        popSize,numGenerations,delh,magX,noSlicesNonCirc,
                        axisPoint, BestNonCirc):

    '''This function encompasses all needed calculations to calculate the factor of safety
    of a slope over a given amount of days for a specific intensity and rainfall duration.
    The following variables are used as input:
        - qRain: rainfall intensity
        - tOut: moment at which to evaluate the FoS
        - tEnd: end of simulation period (usually the maximum entry in tOut)
        - xHead: the positions of the column within the slope for calculation
        - nNode: number of cells per column
        - hCol: height of the columns as defined in xHead
        - tRain: duration of rainfall
        - gwlIni: depth of the groundwater level for each column
        - FlSucMax, sucMax: variables for the maximum suction allowed in the slope
        - kh: horizontal permeability
        - ySlope: similar to hCol
        - grid_x, grid_y: grid used for interpolation
        - topPoints: geometry of the slope
        - noSlices: number of slices to be used in circular failure surface
        - layerData: not needed remnant of earlier versions
        - LL, DL: material properties of liquefiable and non-liquefiable material
        - xStartRange, xEndRange: ranges of coordinate for entry and exit point of the
        slip surface
        - magR: only needed for circular failure surface
        - popSize: population size for DE algorithm
        - numGenerations: maximum number of allowed generations in DE algorithm
        - delh: cell thickness
        - magX: magnification factor of grid
        - noSlicesNonCirc: number of degrees of freedom for non-circular failure surface
        - axisPoint: coordinate of point for calculation of moment balance of slip
        surface
        - BestNoncirc: potentially to be used if one wants to evaluate a given slip
        surface. Otherwise, 0 can be used as input

    Then, for each column, the PySWAP module is applied to obtain the head, and relative
    degree of suction. These are then interpolated onto a 2D-grid. These grids are then
    used together with the other variables to evaluate the factor of safety in the
    separate library that has been built for the evaluation of the slip surface.

    The only thing returned here is the minimum factor of safety of all evaluated points
    that are part of tOut, as well as the details of the critical failure surface.
```

```
Furthermore, for each moment defined in tOut, a plot is made of the degree of
saturation, as well as the critical failure surface. Information of the calculated
FoS, moment in time, as well as some of the important strength and rainfall pattern
variables are also provided in the figure, which has proven to be useful in
storing the figures of different analyses in an archive.
'''
#Starting timer to record computational time
ticFunc = time.perf_counter()

#Resetting some of the overview variables
headOverview = []
SeOverview = []
sigmaOverview = []
gridSeOverview = []
gridheadOverview = []
tCumOldOverview = []
phreaticX = np.array(())
phreaticY = np.array(())

#tic = time.perf_counter()
# Computing the solution to the Richards equation with SWAPPython for the
given rain intensity for all given columns.

print('\nModel is initialised with a rainfall intensity of', -1*qRain,'[m/day]')
print('Solving Richards equation for the slope...')
for i in range(len(xHead)):

    # if the groundwater level has a depth of less than 1 meter, the calculation
    # in SWAP is not performed to limit computational effort
    if gwlIni[i] < -1.0:

        #distance from the toe for the drainage package
        LDrain = max(topPoints[:,0])-xHead[i]

        # Setting the ponding criterion based on the position in the slope. The
        # sloping part is not able to have as much ponding as the non-sloping part.
        if xHead[i] < topPoints[1,0]:
            pondmax = 0.5
        else:
            pondmax = 0.1

        # Initiation of the PySWAP calculation. Returned are the head at all
        # calculation times, the relative degree of saturation, node coordinates
        # and the cumulative time at which the calculation is performed.
        headInt,SeInt,zN,tCumOld =
        swap.SWAP(nNode,hCol[i],tRain,tEnd,gwlIni[i],qRain,mvgn,LDrain,kh,
        FlSucMax,sucMax,pondmax)

        phreaticX = np.hstack((phreaticX,np.array(np.ones(nNode)*xHead[i])))
        phreaticY = np.hstack((phreaticY,hCol[i]+zN))

        # Values are appended to a list for later use
        headOverview.append(headInt)
        SeOverview.append(SeInt)
        tCumOldOverview.append(tCumOld)
        sigmaOverview.append(sigmaInt)
    else:
        # Performed if the groundwater level is only at very limited depth. The
        # head distribution is assumed to be hydrostatic and the degree of saturation
        # is equal to 1. Calculation times are mimicked to be able to use these
        # results later.
```

```
            phreaticX = np.hstack((phreaticX,np.array(np.ones(nNode)*xHead[i])))
            phreaticY = np.hstack((phreaticY,np.linspace(hCol[i],0,nNode)))

            tCumOld = np.vstack(np.linspace(0,tEnd,1000))
            tCumOldOverview.append(tCumOld)
            sigmaInt = np.linspace(0,hCol[i],nNode)*LL.gammaWet
            sigmaOverview.append(sigmaInt)
            headInt = np.vstack(np.linspace(0,hCol[i],nNode))*np.ones((nNode,
            len(tCumOld[:,0])))

            headOverview.append(headInt)
            SeInt = np.ones((nNode,len(tCumOld[:,0])))
            SeOverview.append(SeInt)


    # Initiating the array for storing the calculated FoS. Entries are set to 100 by
    # default to avoid a wrong minimum to be used.
    FoSt = np.ones(len(tOut))*100.

    # print status to kernel
    print('Computing the factor of safety for the given heads...')

    counter = 0 # reset iterator

    #Here the factor of safety is to be computed for all entries of tOut
    for i in range(len(tOut)):

        # Resetting Pres, Se and sigma which are the arrays for interpolating the head
        # and the degree of saturation.
        Pres = np.array(())
        Se = np.array(())

        # Here the correct output arrays that are assembled in headOverview and
        # SeOverview have to be found. Based on the value of tOut, the correct index
        # with a nearby cumulative computation time has to be used. That index
        # is found through the following loop.

        for j in range(len(xHead)):
            for l in range(2,len(tCumOldOverview[j])-1):
                if tOut[i] == tEnd:
                    l = -1
                    break

                tmin1 = tCumOldOverview[j][l-1]
                tplus1 = tCumOldOverview[j][l+1]

                if tOut[i] == 0.:
                    l = 1
                elif tOut[i]>tmin1 and tOut[i]<tplus1:
                    break
                elif l == len(tCumOldOverview[j])-1:
                    # exception if no time has been found
                    raise Exception('No suitable time was found for which to evaluate h')
            #Appending the correct data of the given time to Se and Pres
            Se = np.hstack((Se,np.array(SeOverview[j][:,l])))
            hwSlope = headOverview[j][:,l]
            Pres = np.hstack((Pres,np.array(hwSlope*10.)))


        # Performing the 2D interpolation with the griddata function and archiving
        # in gridSeOverview and gridheadOverview. These can possibly be returned from
```

```
# the function.
grid = griddata((phreaticX,phreaticY), Pres, (grid_x,grid_y), method = 'linear')
gridSe = griddata((phreaticX,phreaticY),Se, (grid_x,grid_y), method = 'linear')
gridSeOverview.append(gridSe)
gridheadOverview.append(grid)

# loop to have all values from the interpolation scheme that rise above the slope
# geometry set to NaN. This is needed if e.g. using a stepped embankment.
slopeData = fl.FuncSlopeData(topPoints)
for k in range(len(grid_x[0,:])):
    for j in range(len(grid_y[:,0])):
        xPoint = grid_x[j,k]

        sectionPoint = fl.FuncSectionFinder(slopeData,xPoint)

        ySlopePoint= xPoint*slopeData[sectionPoint,0]+slopeData[sectionPoint,1]

        # Conditionally setting the values to NaN
        if grid_y[j,k]>ySlopePoint:
            grid[j,k] = np.nan
            gridSe[j,k] =  np.nan

# The differential evolution algorithm is started with the additional input
# of the grid and gridSe for the heads and relative degree of saturation.
# some keyword arguments such as mut and crossp are embedded to not be changed
# graphic = True will return various parameters such as the x- and y-coordinates
# of the slicecuts and slice midpoints to be used for plotting later.
BestNonCirc, FoSNonCirc, xSliceCut, ySliceCut, xMidSlices, yMidSlices =
fl.FuncOwnDe(topPoints,grid,gridSe,layerData,LL, DL,
xStartRange,xEndRange,noSlicesNonCirc,popSize,numGenerations,axisPoint,
magX,gridData = True, mut = 0.8, crossp = 0.7, graphics = True)

# Found FoS is stored in the array
FoSt[i] = FoSNonCirc

# The following lines will create the plot
fig1,(ax1,cax) = plt.subplots(1,2,figsize= (15,4),
gridspec_kw={"width_ratios":[1, 0.05]})

# The contour plot. The label for the legend is the FoS.
gridShowCirc = ax1.contourf(gridSe, extent =  [0,topPoints[-1,0],0,topPoints[0,-1]],
cmap = newcmp)
labelFoSNonCirc = str(round(FoSNonCirc,3))

# Plotting the head from Modflow very lightly as well as slope geometry
ax1.plot(head[0,:], color = 'gray')
ax1.plot(topPoints[:,0],topPoints[:,1], color = 'black')

# Plotting the extent of the shell
ax1.plot(topPoints[:,0]-DL.damWidth,topPoints[:,1], linestyle = 'dashed',
color = 'black')

# Plotting the failure surface: slice cuts and slice midpoints
ax1.scatter(xMidSlices,yMidSlices, marker = 'o', color = 'black',
label = ('FoS = '+labelFoSNonCirc))
ax1.plot(xSliceCut,ySliceCut, linestyle= 'solid', marker = '|', color = 'black')

# Setting axis labels and axis limits
ax1.set_xlabel('[m]', fontsize = 12)
ax1.set_ylabel('[m]', fontsize = 12)
ax1.set_xlim(50,topPoints[-1,0])
```

```
        # Text to be used in the text box. Holds some major parameters for the simulation.
        textstr = '\n'.join((
        r'$q_{rain} = %.2f$' ' [m/day]' % (-1*qRain, ),
        r'Rain duration = %.1f' ' [days]' %(tRain, ),
        r"$S_{u}/\sigma'_{v} = %.2f$ " ' [-]' % (LL.SuSigmaV, ),
        r"$\varphi'_{dam} = %.1f$" ' [$^{o}$]' % (DL.phiDrained, ),
        r"$\varphi'_{tailings} = %.1f$" ' [$^{o}$]' %(LL.phiSuc, )))

        # Plot the textbox
        ax1.text(0.841, 0.85, textstr, transform=ax1.transAxes, fontsize=10,
        verticalalignment='top', bbox = dict(facecolor = 'white',boxstyle = 'round'))

        # Plot the ticks, legend and colorbar at the right positions
        ax1.minorticks_on()
        ax1.legend(fontsize = 12, edgecolor = 'black')
        fig1.colorbar(mappable = gridShowCirc,cax = cax)
        plt.subplots_adjust(wspace=0.02)
        plt.tight_layout()

        # If the FoS is not further decreasing for a number of evaluation times, the
        # computation is stopped as new computations will be additional computational
        # time.
        if FoSt[i] > np.min(FoSt):
            counter +=1
        elif FoSt[i] < np.min(FoSt) and i > 0.:
            counter = 0

        if counter ==2 :
            break

    # Taking the minimum found FoS
    minFoS = np.nanmin(FoSt)

    # Final ouput to the user kernel
    print('The minimum FoS is', minFoS)
    print('The timestep at which this occurs is', tOut[np.argmin(FoSt)])
    print('Total FoS output array is', FoSt)
    tocFunc = time.perf_counter()
    #Resetting the intermediate lists
    print("Computational time is ", tocFunc-ticFunc)

    return minFoS, BestNonCirc
```

The final part consists of calling the *IntensityCalculator* function in the correct way for analysis. The code that is shown below is needed to obtain the FoS at different intensities to calculate the return period. It is noted that still some manual calculations are needed after. A small interpolation code was used for this purpose. In earlier phases, a Newton-Raphson algorithm was applied. This however proved to be relatively unstable and a lot of iterations were needed. Since the computational time of each cycle to calculate the minimum FoS for each intensity is relatively large, this was not performed anymore. Now, it calculates the FoS for different intensities in an incremental manner for each duration of rainfall. When the FoS is below 1.1 it does another step back in between the final two intensities to obtain more information on the possible position of the intensity for which the FoS is 1.1. The final part of the code causes a notification to show when the simulation is finished. This concludes for the main code.

```
#%% New implementation that is based on incremental intensity


FoS = []
```

```
BestNonCirc = 0
for i in range(len(tEndFlowArray)):
    print('\nSequence is started for', tEndFlowArray[i], 'days of rainfall')
    tRain = tEndFlowArray[i]
    tEnd = tRain +40.
    tOut = np.linspace(15.,tEnd,6)

    if tRain == 1:
        qRainArray = np.array([0.001,0.25])
        FoSInt1 = np.zeros(len(qRainArray))
        for j in range(len(qRainArray)):
            qRain = -1*qRainArray[j]
            FoSInt1[j],BestNonCirc = IntensityCalculator(qRain,
                              tOut,tEnd, xHead, nNode, hCol, tRain, gwlIni,FlSucMax,
                              sucMax, kh, ySlope, grid_x,grid_y, topPoints,
                              noSlices,layerData,LL,DL,xStartRange,xEndRange,magR,
                              popSize,numGenerations,delh,magX, noSlicesNonCirc,axisPoint,
                              BestNonCirc)
            if FoSInt1[0]<1.1:
                break
        if FoSInt1[0]<1.1:
            break
        FoS.append(FoSInt1)

    if tRain > 1:
        qRainArray = np.linspace(0.05,0.2,4)
        FoSInt = np.zeros(len(qRainArray)+1)
        for j in range(len(qRainArray)):
            qRain = -1*qRainArray[j]
            FoSInt[j],BestNonCirc = IntensityCalculator(qRain,
                              tOut,tEnd, xHead, nNode, hCol, tRain, gwlIni,FlSucMax,
                              sucMax, kh, ySlope, grid_x,grid_y, topPoints,
                              noSlices,layerData,LL,DL,xStartRange,xEndRange,magR,
                              popSize,numGenerations,delh,magX, noSlicesNonCirc,axisPoint,
                              BestNonCirc)
            if FoSInt[j] < 1.1 and j > 0:
                qRain = -1*(qRainArray[j]+qRainArray[j-1])/2
                FoSInt[j],BestNonCirc = IntensityCalculator(qRain,
                              tOut,tEnd, xHead, nNode, hCol, tRain, gwlIni,FlSucMax,
                              sucMax, kh, ySlope, grid_x,grid_y, topPoints,
                              noSlices,layerData,LL,DL,xStartRange,xEndRange,magR,
                              popSize,numGenerations,delh,magX, noSlicesNonCirc,axisPoint,
                              BestNonCirc)
                break
            elif FoSInt[j] < 1.1 and j == 0:
                qRain = -1*0.025
                FoSInt[j],BestNonCirc = IntensityCalculator(qRain,
                              tOut,tEnd, xHead, nNode, hCol, tRain, gwlIni,FlSucMax,
                              sucMax, kh, ySlope, grid_x,grid_y, topPoints,
                              noSlices,layerData,LL,DL,xStartRange,xEndRange,magR,
                              popSize,numGenerations,delh,magX, noSlicesNonCirc,axisPoint,
                              BestNonCirc)
                break
        FoS.append(FoSInt)

MessageBox = ctypes.windll.user32.MessageBoxW
MessageBox(None, 'Simulation is finished', 'Simulation update', 0)
```

D.2. Flopy/Modflow initial conditions

The initial conditions to the analysis are formed with the help of the MODFLOW software. MODFLOW is a general finite difference solver to the Laplace equation and is a well-established software package in the field of groundwater modelling. It is lacking a user interface and setting up the necessary files is notoriously complex. Therefore, lately a Python library called FloPy has been written that serves as an interface to Modflow (Bakker et al., 2016). This Python library has been implemented and used in the analysis. A specific function called *ModFlow* was written in Python to control the FloPy library. The input for this function is the geometry of the slope as well as some geohydrological parameters as contained in the tuple *GH*. The desired output is the head according to the boundary conditions imposed along the entire impoundment. The source code can be seen below.

```python
def ModFlow(modelName, topPoints,GH,zTopModFlow):
    mf = flopy.modflow.Modflow(modelName, exe_name = 'mf2005')
    Lx = topPoints[-1,0]
    Ly = 1.
    nrow = 1
    ncol = len(zTopModFlow[0,:])
    nlay = 1

    ztop = np.ones((1,ncol))
    ztop = zTopModFlow
    zbot = -3.0

    delr = Lx / ncol
    delc = Ly / nrow
    delv = (ztop - zbot) / nlay
    botm = np.linspace(ztop, zbot, nlay + 1)

    nper = 1
    perlen = [1]
    nstp = [1]
    steady =[True]

    hk = GH.hk
    vka = GH.vka
    Sy = GH.Sy
    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)

    laytyp = np.ones(nlay)
    laywet = np.ones(nlay)

    ibound = np.ones((nlay, nrow, ncol), dtype=np.int32)
    ibound[-1, :, -1] = -1
    ibound[:,:,0] = -1
    strt = GH.InitC*np.ones((nlay, nrow, ncol), dtype=np.float32)
    strt[-1, :, -1] = 0
    strt[:,:,0] = GH.bndHead
    bas = flopy.modflow.ModflowBas(mf, ibound=ibound, strt=strt)

    lpf = flopy.modflow.ModflowLpf(mf, hk = hk, vka=vka, sy = Sy, hdry = -1,
    laytyp = laytyp, laywet = laywet, ipakcb=53)

    Drains = []
    for i in range(len(ztop[0,:])):
        Drains.append([0,0,i,ztop[0,i],100])
    stress_period_drain = {0:Drains}
    drn = flopy.modflow.ModflowDrn(mf,stress_period_data = stress_period_drain)
    #rch = flopy.modflow.ModflowRch(mf, nrchop = 3, rech = rech)

    stress_period_data = {}
    for kper in range(nper):
        for kstp in range(nstp[kper]):
            stress_period_data[(kper, kstp)] = ['save head',
                                                'save drawdown',
                                                'save budget',
                                                'print head',
                                                'print budget']
    oc = flopy.modflow.ModflowOc(mf, stress_period_data=stress_period_data,
                                 compact=True)


    pcg = flopy.modflow.ModflowPcg(mf)

    mf.write_input()

    success, mfoutput = mf.run_model(silent=True, pause=False, report=True)
    if not success:
        print(mfoutput)
        raise Exception('MODFLOW did not terminate normally.')


    hds = bf.HeadFile(modelName + '.hds')
    times = hds.get_times()
    headXSectionSum = 0
    timeMax = 0
    headXSectionOutput = []

    for i in range(len(times)):
        head = hds.get_data(totim=times[i])
        headXSectionInter = head[:,0,:]
        headXSectionOutput.append(headXSectionInter)

        if np.sum(headXSectionInter)> headXSectionSum:
            headXSection = headXSectionInter
            headXSectionSum = np.sum(headXSectionInter)
            timeMax = times[i]

    return headXSection, mfoutput
```

First, initialisation of the Modflow model is needed. The model file name will get the name as provided in the function. Then the length of the considered geometry is defined in horizontal and vertical direction. In this case, the 2D application of Modflow is used. The y-coordinate is therefore used for the vertical direction. It has been decided not to account for vertical flow to ease calculations. It is expected that the effect is only minimal (also proven essentially with the Plaxis comparison) and it makes Modflow stable. Modflow in itself is rather unstable and will raise exceptions and errors because of various parameters not clearly defined or not defined within the correct range. The latter is however not provided when that is the case and the error message will just read that convergence was not reached. It is therefore advised not to apply too many changes to this function.

The number of rows and layers is in this sense equal to one, as only a single layer is defined. The number of columns concerns the cell width in horizontal direction. Because of the accuracy to which a solution is needed, this is defined at every meter. Higher resolutions were not necessary or did not provide more accurate results. After that, the top and bottom coordinates of each column are defined. The top of the columns are an input to the function. The bottom is set at -3.0 meters as to not influence the results because of the reducing transmissivity towards the toe. *delr, delc, delv and botm* are parameters needed to define the cell widths and heights.

The next lines of code denote the number of stress periods to be calculated and in accordance the number of timesteps. This is however a steady-state analysis and therefore the number of periods is only one. The geohydrological parameters (*hk, vka and Sy*) are initialised from the input tuple which allows for the *dis* file (discretisation) to be set-up by FloPy, which is then used in Modflow. The layertype and layer wetting are both set to 1 and indicate that the layer can be wetted and that the transmissivity is calculated to be in accordance with the phreatic table (unconfined acquifer). More on this can be found in Harbaugh (2005). Then two rather important arrays are initialised. These are the *ibound* and *strt* arrays. The *ibound* array will hold only ones except for those cells for which the head is already defined (boundary conditions) those values are changed to -1. The *strt* array is essentially the starting/initial condition array. However, the value defined for those cells that have an ibound value of -1 will need to have the boundary conditions as these will no longer change. After that, the *bas* and *lpf* files are initialised.

In order to avoid the head to rise above the slope, drains are positioned on the top of each column. A list is made to which these drains are appended. The values resemble: layer number, row number, column number, vertical position with respect to zero and the conductivity. The conductivity of the drain is set high as to take away any additional head rising above the surface immediately. The stress period is set to the first and only stress period as a steady-state analysis is run. The *drn* file may now also be implemented.

Next is the definition of the output characterisation. Essentially for all periods and timesteps, the head, drawdown and water budget are saved and the head and budget are also printed. The *pcg* file will initialise the solver, which in this case is the parallel conjugate gradient solver. The model is run and if it is successful, no exception will be raised. The last part consists of retrieving the heads from the *hds* output file and adds some alteration of the array as to ease later calculations. The final head for each column as well as the output results are returned to the main code.

D.3. Unsaturated flow analysis

As stated in the main text in chapter 4, the unsaturated flow analysis was to a large extent based on the program SWAP which was written in FORTRAN 90. Although the documentation on SWAP is rather extensive and no large changes have been made to this source, other than omitting a substantial portion, the code as adopted in PySWAP will be treated here. The first reason being, the completeness of the documentation and offering the reader sufficient background information within this text. Second, the source code of SWAP deviates at some points from the formulae mentioned in its documentation. The author has found this merit confusing and would therefore prefer to offer some comments to the code in this section. The following will follow the same structure as the code, meaning that it starts with the main function that is run when using PySWAP. Then all secondary functions (PySWAP library) will be treated.

D.3.1. Main PySWAP function

PySWAP starts with calling the function SWAP, which is listed below. At first, some additional variables are being intialised. These have been treated before in chapter 5. All other values are being set to zero. Then, the initial conditions are formed with the *calcGrid*, *timeControl* and *InCo* functions. These will form the grid, the initial time step for calculation and the initial conditions of the state variables: pressure head, water content and (unsaturated) permeabilities. After that, a while loop is started until the end of the simulation period has been reached. Within the loop, the state variables and calculation time steps are stored. The pressure heads (and other state variables) are calculated by using the *soilWater* function. Then also the change of water content per cell and the drainage package are implemented. The last lines (commented here) allow for plotting of the state variables at different time steps. This is the main part of the PySWAP function

```
def SWAP(nNode,hCol,tRain,tEnd,gwlIni,qRain,mvgn,LDrain,kh,FlSucMax,sucMax, pondmax):

    dT = 0.                 # Timestep size for current time step

    dTMin = 1e-6          # Minimum allowed timestep in days
    dTMax = 0.2            # Maximum allowed timestep in days
    tCumOld = 0.            # Total cumulative time after previous time step
    tCumOldArchive = np.zeros(1)
    tCum = 0.               # Total cumulative time after current time step

    pond = 0.               # Pond level parameter
    pondm1 = 0.           # Pond level at previous timestep
    numbit = 0


    gwlDra = hCol+gwlIni

    draindic={'gwlDra': gwlDra,
              'LDrain': LDrain,
              'kh': kh}

    #Numerical and convergence parameters
    MaxIt = 75
    maxBackTr = 8

    CritDevBalCp = 1e-2
    CritDevh2Cp = 5e-2
    CritDevh1Cp = 1e-2
    CritDevBalTot = 1e-4

    #Variables for the Mualem-van Genuchten equations


    #Variables for runoff and ponding
    rsro = 1
    pondmax = pondmax

    #Variables for top and bottom flux boundary

    qBot = 0.0

    #Variables for the state parameters
    h = np.zeros(nNode)         # Initialising the head profile
    hold = np.zeros(nNode)
    theta = np.zeros(nNode)     # Initialising the water content profile
    relSat = np.zeros(nNode)
    dimoca = np.zeros(nNode)    # Initialising the array for the differential moisture
                                #    capacity
    kNode = np.zeros(nNode)     # Initialising the array for the node conductivities
    kMean = np.zeros(nNode+1)   # Initialising the array for the internode conductivities
                                #based on the weighted arithmic mean
    hArchive = np.zeros((nNode,1))  # List to which the heads will be saved
    hmin1 = np.zeros(nNode)     # Immediate array for the heads of the previous timestep
    thetaArchive = np.zeros((nNode,1))  # List to which the water contents will be saved
    thetamin1 = np.zeros(nNode) # Immediate array for the water content of the previous
                                # timestep
    relSatArchive = np.zeros((nNode,1))
```

```
q = np.zeros(nNode+1)

dFdhM = np.zeros(nNode)
dFdhU = np.zeros(nNode)
dFdhL = np.zeros(nNode)
F = np.zeros(nNode)
hGrad = np.zeros(nNode)

#Needed flags
FlRain = True                # Flag for stating that there is rainfall
Fltoph = False               # Flag whether top boundary condition changes to
                             #    a head based one
FlNonConv = False            # Flag for non-convergence of the solution
Fldecdt = False
FlNonConv = False



pondVar= {'rsro':rsro,
          'pondmax':pondmax}

bound = {'qRain':qRain,
         'qBot': qBot}

#%% Setting the grid formulation, initial time step, initial conditions of state parameters

zN,dZ,disnod = usl.calcGrid(nNode,hCol)

dT,tCumOld,FlRain = usl.timeControl(dT,dTMin, dTMax,
tCumOld,tCum,tRain,tEnd,numbit,Fldecdt,FlRain)

h,theta,relSat,dimoca,kNode,kMean =usl.inCo(h,theta,relSat,dimoca,kNode, kMean,
gwlIni,zN,nNode,mvgn,dZ,dT,FlSucMax,sucMax)

#%% Running the main program
# fig1,ax1 = plt.subplots()
# fig2,ax2 = plt.subplots()
# fig,ax3 = plt.subplots()

#loop trough the simulation period until end is reached
counter = 0
while tCumOld < tEnd:
    counter += 1

    hvec = np.vstack(h)
    thetavec = np.vstack(theta)
    relSatvec = np.vstack(relSat)
    hArchive = np.append(hArchive,hvec,axis = 1)
    thetaArchive = np.append(thetaArchive,thetavec,axis = 1)
    relSatArchive = np.append(relSatArchive,relSatvec,axis = 1)
    tCumOldArchive = np.append(tCumOldArchive,tCumOld)

    #Calculate the level of the groundwater
    gwl, gwlsur, compGwl = usl.calcgwl(h,nNode,hCol)

    #Call the drainage package for the lateral drainage
    qdra = usl.drainage(gwl,compGwl, draindic,nNode)

    #Reset, save and calculate new soil water state variables
    h, theta, relSat, kNode, kMean, Fldecdt,numbit  = usl.soilWater(
                        h,hmin1,hold,hArchive,theta,thetaArchive, thetamin1,
                        qdra,relSat,dimoca,dFdhM, nNode,dFdhU,dFdhL,
                        F,hGrad,kMean,kNode, bound,mvgn,pondm1,pond,
                        pondVar,dZ,disnod,dT, dTMin ,MaxIt,maxBackTr,
                        CritDevBalCp, CritDevh2Cp,CritDevh1Cp,CritDevBalTot,
                        FlNonConv, Fldecdt,FlRain)


    #update the hydraulic conductivities and calculate the fluxes/soilWater
    # rate variables
    kNode, relSat, kMean, q =
    usl.soilWaterRate(h,theta,relSat,thetamin1,q,kNode,kMean,nNode,dZ,dT,mvgn,bound)

    #Compute the new timestep
    dT, tCumOld,FlRain = usl.timeControl(dT,dTMin, dTMax,
    tCumOld,tCum,tRain,tEnd,numbit,Fldecdt,FlRain)


    #Plotting and testing
#     ax3.plot(tCumOld,gwl, 'o')
#     if counter%10 == 0:
#             ax1.plot(h,zN)
#             ax2.plot(theta,zN)

# ax1.grid()
# ax2.grid()


return hArchive, relSatArchive,zN, tCumOldArchive
```

At first, some additional parameters are initialised. These equate to zero to be input arguments for the functions to be used. There are however a number of additional numerical parameters as also treated in chapter 5. It is advised for further use to not change these numerical parameters too much. That is why they have also not been included as main input parameters for this SWAP function, but are embedded within the function itself.

D.3.2. PySWAP function library

The PySWAP function library is an extensive function library which is the main computational heart of the PySWAP module. The structure follows roughly that of SWAP but is all assembled in a single file. The functions in this library will now be treated in their coding order, which follows simulation order to most extent. The first function is the *calcGrid* function which initialises the grid or cells:

```
def calcGrid(nNode, hCol):
''' Creates a linearly spaced grid for the given height of the column.
The z-coordinate ofthe node
starts at 0 at the top and increases linearly with the depth '''
disnod = hCol/nNode*np.ones(nNode)
disnod[0] = 0.5*disnod[0]
zN = np.arange(-0.5*disnod[1],-hCol,-disnod[1])
dZ = hCol/nNode*np.ones(nNode)
return zN, dZ, disnod
```

As provided in the function documentation, a linearly spaced grid is provided for the given height of the column. The top starts at zero and increases towards more negative values. These indicate the height of the various nodes. The *disnod* array represents the nodal distance, i.e. the distance from a node to its upper node. The *zN* is the nodal z-position and the *dZ* is the cell thickness. Although the SWAP documentation advises to apply smaller cells near the infiltration boundary, a uniform grid has proven to be more stable, whilst the loss in quality was low for the given number of nodes in Chapter 5. The three arrays are returned to the main PySWAP code.

The following function is the *timeControl* function and it essentially computes the needed step size in time for the next calculation. It does this on the basis of the computational performance of the previous time step. It may be understood that when many iterations are needed in order to converge to a solution for each time step, that a smaller time step is desired for the next one. And, in the other way around, when the computational performance is very good and little iterations are needed, the time step can be increased as to minimise the computational time needed for solution. This function does exactly this. It analyses computational performance based on the number of iterations and changes the time step when needed. The time step is decreased when it exceeds 20 iterations and is increased when less than 3 iterations are needed. Here the programme diverges from that of SWAP as SWAP will only increase the time step when convergence in the time step has failed. This does however not anticipate well on already increasing numbers of iterations and it has been found that when decreasing the time step earlier in the process excessive iterations can be avoided. The *decreaseTime* function is needed to decrease the time step.

There are two additional events that need an exception. Those are the starting time step and the time step to an event. The starting time step is equal to: $\sqrt{dTMin \cdot dTMax}$ as also provided in the SWAP documentation. This time step works rather well, although it is noticed that the time step decreases rapidly after. This may have to do with the fact that high intensities are applied and that small time steps are needed as heads quickly increase. The second event is the fact that a time step needs to be implemented exactly at that moment where rainfall stops in order to avoid any discrepancies in the rainfall depth that is applied. The programme checks continuously for the residual time to this event and will limit the time step if necessary. Additional work to the programme could include more of such events.

```
def timeControl(dT,dTMin, dTMax, tCumOld, tCum,tRain,tEnd,numbit,Fldecdt,FlRain):
    '''This function computes the next timestep size from:
        -The start
        -based on numerical performance of the previous timestep
        -based on the residual time left for the event (either end of
        precipitation or end of timeline)
        '''
    tCumOld = tCumOld+dT
    dTprevious = dT
    # Starting time step
    if tCumOld == 0:
        dT = 0.5*np.sqrt(dTMin*dTMax)

        return dT,tCumOld,FlRain

    #When running the function
    #Calculate time till events
    #Time till end rainfall
    dTRain = tRain-tCumOld
    if dTRain <= 0.:
        FlRain = False
    else:
        FlRain = True

    #Time till end of simulation
    dTEnd = tEnd - tCumOld

    #Compare the two critical ones
    if FlRain == True and dTRain !=0:
        dTEvent = dTRain
    if FlRain == False:
        dTEvent = dTEnd

    #Check for numerical performance
    #First with the flag for decrease (see where this one comes from)
    if Fldecdt == True:
        dT = decreaseTime(dT)
        Fldecdt = False
        return dT, tCumOld,FlRain

    #Then for the numerical performance
    if numbit <= 3:
        dtNum = min(dT*2.0, dTMax)

    elif numbit >=20:
        dtNum = max(dT*0.5, dTMin)
    else:
        dtNum = dT

    #Taking the minimum
    dT = min(dTEvent,dtNum)

    return dT, tCumOld,FlRain
```

The *decreaseTime* function is embedded in the *timeControl* one and calculates the time step if convergence has not been reached and a smaller time step is needed. It divides the given time step by three if needed so. Except for when the current time step is smaller than three times the minimum time step, then it is limited to this minimum timestep.

```
def decreaseTime(dT,dTMin):
    if dT > 3.0*dTMin:
        dT = dT/3.0
    else:
        dT = dTMin
    return dT
```

The next function in the library is the *inCo* function which is short for ‘initial conditions’. As provided within the function documentation, it provides with the initial head profile, initial water content profile, initial hydraulic conductivities at the nodes and at the internodes/cell borders and lastly the initial differential moisture capacity. These values are needed in order to start calculations. The initial pressure head profile is provided as to be linear with depth. Increasing depth will mean increasing pressure head. It is noted that when in the provided user input, the *FlsucMax* is true, the suction will be limited to the given maximum value. For the other values, specific functions are used that are treated further in this section.

```
def inCo(h,theta,relSat,dimoca,kNode, kMean, gwlIni,zN,nNode,mvgn,dZ,dT,FlsucMax,sucMax):
    ''' Function should form the initial conditions of the program. This means that it
    calculates:
        -The initial head profile from the groundwater level (linear head over distance)
        -The initial water content profile
        -The initial hydraulic conductivities and mean hydraulic conductivities
        -The initial differential moisture capacities
    '''
    #Calculation of the initial head profile
    for i in range(nNode):
        h[i] = gwlIni - zN[i]
        if FlsucMax:
            h[i] =  max(h[i], sucMax)
    #Calculation of the initial water content profile
    theta = watCon(h,theta,nNode,mvgn)

    #Calculation of the initial moisture capacity
    dimoca = moisCap(h,theta,dimoca,nNode,dT,mvgn)

    #Calculation of the permeabilities at the nodes
    kNode,relSat = hConduc(theta,relSat,kNode,nNode,mvgn)

    #Calculation of the permeabilities at the internodes
    kMean = hConducMean(kNode, kMean, nNode,dZ)

    return h,theta,relSat,dimoca,kNode,kMean
```

The *watCon* function calculates the water content based on the pressure head. This version is the more extended one which also considers the effect of the air-entry value (AEV). At first the parameter h_{crit} is given which is equal to 1 cm essentially. If the AEV is greater (less negative) than this value, it is neglected in the analysis. Then the standard Mualem-van Genuchten equations are applied in order to calculate the water content. If the head is positive, the water content is limited to the saturated water content (i.e. porosity). Otherwise the Mualem- van Genuchten equations are used. This is only the case if the suction head is greater than the h_{crit}. Otherwise, a simplified equation is used.

When the AEV is larger than this critical value, a cubic spline is applied. This sequence has been provided with in SWAP and the details are not treated here. This part works, but it must be said that it does not look efficient when applying it in Python. SWAP is programmed in the lower level language of FORTRAN 90 and in this way embedded functions in the language were not used/possible. Python has however very extended libraries and it is argued that Python could fit a cubic spline and apply these values. This will probably be more accurate and should be applied in later studies with this code. it was noted that the calculated water content with the splines could rise above that of the saturated water content. It was unsure why this was the case, but the values were limited as to avoid also the relative degree of saturation to rise above 1. As stated, there is some significant room for improvement here. It may then also be investigated why the results with the AEV deviate from that in literature as also provided in section 4.3.1.

```
def watCon(h,theta,nNode,mvgn):
    '''Calculates the water content from pressure head'''

    h_crit = -1e-2

    for i in range(nNode):
        if mvgn['h_enpr'] > h_crit:
            if h[i] >= 0.0:
                theta[i] = mvgn['thetas']
            elif h[i] > h_crit:
                a = (abs(mvgn['alpha']*h_crit))**mvgn['n']
                a = (1+a)**mvgn['m']
                a = mvgn['thetar']+(mvgn['thetas']-mvgn['thetar'])/a
                watcon = a +(mvgn['thetas']-a)/(-1*h_crit)*(h[i]-h_crit)
                theta[i] =  min(watcon,mvgn['thetas'])
            else:
                a = (abs(mvgn['alpha']*h[i]))**mvgn['n']
                a = (1.0+a)**mvgn['m']
                theta[i] = mvgn['thetar']+(mvgn['thetas']-mvgn['thetar'])/a
        else:
            h105 = 1.05 * mvgn['h_enpr']

            if h[i] >= h105:
                t105 = mvgn['thetar']+(mvgn['thetas']-mvgn['thetar'])*                         \
                    ((1.0+(abs(mvgn['alpha']*mvgn['h_enpr']))**mvgn['n'])**mvgn['m'])/      \
                        ((1.0+(abs(mvgn['alpha']*h105))**mvgn['n'])**mvgn['m'])

                c105 = (mvgn['thetas']-mvgn['thetar'])*mvgn['alpha']*mvgn['m']*mvgn['n']*    \
                    (abs(mvgn['alpha']*h105)**(mvgn['n']-1))*                                \
                        ((1.0+abs(mvgn['alpha']*mvgn['h_enpr'])**mvgn['n'])**mvgn['m'])/     \
                            ((1.0+abs(mvgn['alpha']*h105)**mvgn['n'])**(mvgn['m']+1))
                a = (t105-mvgn['thetas']-c105*h105)/(c105*h105**2)
                b = (t105**2-2*t105*mvgn['thetas']+mvgn['thetas']**2)/(t105-mvgn['thetas']
                -c105*h105)
                theta[i] = mvgn['thetas']+b*a*h[i]/(1.0+a*h[i])
            else:
                a = abs(mvgn['alpha']*h[i])**mvgn['n']
                a = (1.0+a)**mvgn['m']
                s_enpr = (1.0+(abs(mvgn['alpha']*mvgn['h_enpr']))**mvgn['n'])**mvgn['m']
                theta[i] = mvgn['thetar']+(mvgn['thetas']-mvgn['thetar'])/a*s_enpr

            #Reducing theta so that it cannot go above thetas. not sure why it is allowed
            #to do so.
            theta[i] = min(theta[i],mvgn['thetas'])

    return theta
```

In function *moisCap*, the differential moisture capacity is calculated. The differential moisture capacity describes the derivative of the SWCC for water content and head. The structure of the function is similar to that for the water content. However, it has some additional terms. It is noted that also here an advantage may be gained by implementing Python-based cubic splines.

```
def moisCap(h,theta,dimoca,nNode,dT,mvgn):
    '''Calculates the differential moisture capacity'''
    h_crit = -1e-2
    for i in range(nNode):
        if mvgn['h_enpr'] > h_crit:
            if h[i]>=0.0:
                dimoca[i] = dT*1e-7
            elif h[i] > h_crit:
                a = (abs(mvgn['alpha']*h_crit))**mvgn['n']
                a = (1+a)**mvgn['m']
                a = mvgn['thetar']+(mvgn['thetas']-mvgn['thetar'])/a
                watcon = a +(mvgn['thetas']-a)/(-1*h_crit)*(h[i]-h_crit)
                term1 =  min(watcon,mvgn['thetas'])
                dimoca[i] = (mvgn['thetas']-term1)/(-1*h_crit)
            else:
                alphah = abs(mvgn['alpha']*h[i])
                term1 = alphah**(mvgn['n']-1.)
                term2 = term1*alphah
                term2 = (1.+term2)**(mvgn['m']+1.)
                term2 = (mvgn['thetas']-mvgn['thetar'] )/term2
                dimoca[i] = abs(-1.*mvgn['n']*mvgn['m']*mvgn['alpha']*term2*term1)
        else:
            h105 = 1.05 * mvgn['h_enpr']

            if h[i] >= h105:
                t105 = mvgn['thetar']+(mvgn['thetas']-mvgn['thetar'])*                      \
                    ((1.0+(abs(mvgn['alpha']*mvgn['h_enpr']))**mvgn['n'])**mvgn['m'])/      \
                        ((1.0+(abs(mvgn['alpha']*h105))**mvgn['n'])**mvgn['m'])

                c105 = (mvgn['thetas']-mvgn['thetar'])*mvgn['alpha']*mvgn['m']*mvgn['n']*    \
                    (abs(mvgn['alpha']*h105)**(mvgn['n']-1))*                                \
                        ((1.0+abs(mvgn['alpha']*mvgn['h_enpr'])**mvgn['n'])**mvgn['m'])     \
                            ((1.0+abs(mvgn['alpha']*h105)**mvgn['n'])**(mvgn['m']+1))
                a = (t105-mvgn['thetas']-c105*h105)/(c105*h105**2)
                b = (t105**2-2*t105*mvgn['thetas']+mvgn['thetas']**2)/(t105-mvgn['thetas']
                -c105*h105)
                dimoca[i] = b*a/((1.0+a*h[i])**2)
            else:
                alphah = abs(mvgn['alpha']*h[i])
                term1 = alphah**(mvgn['n']-1.)
                term2 = term1*alphah
                term2 = (1.+term2)**(mvgn['m']+1.)
                term2 = (mvgn['thetas']-mvgn['thetar'] )/term2
                s_enpr = (1.0+(abs(mvgn['alpha']*mvgn['h_enpr']))**mvgn['n'])**mvgn['m']
                dimoca[i] = abs(-1.*mvgn['n']*mvgn['m']*mvgn['alpha']*term2*term1)*s_enpr

        if h[i] > -1.0 and dimoca[i] < dT*1e-7:
            dimoca[i] = dT*1e-7

    return dimoca
```

The function *hConduc* calculates the (unsaturated) hydraulic conductivity based on the water content and the head. It follows the classical formulation of the Mualem-van Genuchten equation for the hydraulic head. Note that sometimes the inverse of the formula is applied and that might change some of the powers. The flow path tortuosity is set to 0.5 implicitly in this formula. The function also calculates the value of the relative degree of saturation. Although in the original implementation of SWAP this does not play a very significant value, the importance of this degree of saturation is stressed in the implementation here as it is essentially the effective stress parameter in Bishop's formula for effective stress.

```
def hConduc(theta, relSat, kNode,nNode,mvgn):
    '''Calculate hydraulic conductivity as a function of the water content and head.
    Here a different formulation for the hydraulic conductivity is used.
    Which basically means that the value for the parameter lambda
    (which is the value for the tortuosity of the flow path)
    has a default value of 0.5.'''

    h_crit = -1e-2

    for i in range(nNode):
        relSat[i] = (theta[i]-mvgn['thetar'])/(mvgn['thetas']-mvgn['thetar'])
        if mvgn['h_enpr'] > h_crit:
            if relSat[i] < 0.001:
                kNode[i] = 1e-10
            elif relSat[i]>(1-1e-6):
                kNode[i] = mvgn['kSat']
            else:
                # term1 = (1.0-relSat[i]**(1/mvgn['m']))**mvgn['m']
                # kNode[i] = mvgn['kSat']*(relSat[i]**0.5)*(1.0-term1)*(1.0-term1)
                permN = relSat[i]**0.5*(1-(1-relSat[i]**(1/mvgn['m']))**mvgn['m'])**2
                kNode[i] = mvgn['kSat']*permN

        else:
            thetam = mvgn['thetar']+(mvgn['thetas']-mvgn['thetar'])*(1.0+abs(mvgn['alpha']*
            mvgn['h_enpr'])**mvgn['n'])**mvgn['m']

            if relSat[i] < 0.001:
                kNode[i] = 1e-10
            else:
                if theta[i] >= thetam:
                    kNode[i] = mvgn['kSat']
                else:
                    relsatm = (theta[i]-mvgn['thetar'])/(thetam-mvgn['thetar'])
                    relsat1 = (mvgn['thetas']-mvgn['thetar'])/(thetam-mvgn['thetar'])
                    term1 = (1.0-relsatm**(1/mvgn['m']))**mvgn['m']
                    term2 = (1.0-relsat1**(1/mvgn['m']))**mvgn['m']
                    kNode[i] = mvgn['kSat']*relSat[i]*((1.0-term1)/(1.0-term2))**2

    return kNode, relSat
```

The next formula calculates the hydraulic conductivity to be applied in the solution mechanism. As explained, the hydraulic conductivity is computed in the nodes as these hold the values of water content and pressure head. However, applying the finite difference scheme, the in- and outflow are defined at the borders of each cell (internodes). *hConducMean* is a function that calculates the hydraulic conductivity at these internodes. Within SWAP there is a variety of methods to be chosen. Here only the weighed arithmic mean is implemented. Note however that with the given function *calcGrid* the value changes to the mean of the hydraulic conductivity in the neighbouring nodes as all cell sizes are equal.

```
def hConducMean(kNode, kMean, nNode,dZ):
    '''Calculates the permeability at the internodes, meaning that it takes some algorithm to
    go from the conductivities at the nodes to the conductivities in between. In this case,
    only the weighted arithmic mean is programmed. This was advised in the SWAP manual'''

    for i in range(1,nNode):
        kMean[i]=(dZ[i-1]*kNode[i-1]+dZ[i]*kNode[i])/(dZ[i-1]+dZ[i])
    kMean[-1] = kNode[-1]
    return kMean
```

The *soilWater* function in the PySWAP library does not serve a lot of added value. It is a general overhead function for the computational phase in which some of the values are stored, and a new phase is initiated. Within SWAP the function is much more extensive and it resets some of the variables. Within Python this is not necessary and will only lead to confusion and more overhead time. This function is therefore rather rudimentary and could be eliminated possibly in future implementations. Main important thing is that it stores the state variables of the previous timesteps (needed in the calculation) and that it initiates the *headCalc* function.

```
def soilWater(h,hmin1,hold,hArchive, theta, thetaArchive, thetamin1,qdra,relSat,
        dimoca,dFdhM, nNode,dFdhU,dFdhL,F,hGrad,kMean,kNode, bound,mvgn,pondm1,pond,
        pondVar,dZ,disnod, dT, dTMin,MaxIt,maxBackTr,CritDevBalCp,
        CritDevh2Cp,CritDevh1Cp,CritDevBalTot, FlNonConv, Fldecdt,FlRain):

    '''Global overhead function for saving, resetting variables and initialising the
    calculation of the new state variables'''

    #And call the 'SoilWaterStateVar'
    for i in range(nNode):
        hmin1[i] = h[i]
        thetamin1[i] = theta[i]
    pondm1 = pond

    #Calculating the new variables
    h, theta,kNode, kMean, Fldecdt,numbit = headCalc(
                    h,hmin1,hold, theta,relSat,thetamin1,qdra,dimoca,dFdhM,
                    nNode,dFdhU,dFdhL,F,hGrad,kMean,kNode, bound,mvgn,pondm1,pond,
                    pondVar,dZ,disnod, dT,dTMin,MaxIt, maxBackTr,CritDevBalCp,
                    CritDevh2Cp,CritDevh1Cp,CritDevBalTot, FlNonConv, Fldecdt,FlRain)

    return h, theta,relSat, kNode, kMean, Fldecdt, numbit
```

The *headCalc* function is the main function within the computational part of PySWAP. It will calculate pressure heads, water contents, degree of saturation, etc. for the next time step and will call upon earlier mentioned functions to do so. However, the assembly of the Jacobian and water balance are also done within this function. The function diverges from SWAP as relatively a lot has been left out, essentially belonging to the macropore module in SWAP or the use of implicit hydraulic conductivities. Both are not applied here. The latter is something that is recommended to use in future studies as to improve the calculation quality. Essentially the hydraulic conductivity is then reiterated every calculation/iteration step and not left constant.

At first, the hydraulic conductivities are calculated based on the heads and water content from the previous time step. This allows for assembly of the jacobian to some extent. The jacobian resembles the derivative of the water balance with head (dF/dh) and will consist of three diagonals. This as a jacobian value can be calculated for each cell itself and for the relation with the upper and lower cell. Calculation of the hydraulic conductivities allows for the assembly of the upper and lower diagonal as these are related to the in- and outflow to the neighbouring cells. After that, the *hGrad* vector is needed. This basically formulates the gradient of the head between each cell. If the difference between the two is equal to the nodal distance, the gradient reduces to zero. This is essentially the hydrostatic equilibrium. The 1 that is added in this line is therefore a 'correction' for the fact that the state variable is the pressure head rather than the piezometric level.

Then the F vector is initiated. The F vector basically comprises the water balance of each cell. First, the water balance of the top cell is calculated. After that, the *boundTop* function is applied to formulate the boundary condition. If *Fltoph* is true, it means that the top boundary is head based and some further calculations based on the ponding head are needed. The additional inflow is then added to the water balance. If the boundary condition is discharge-based *q1* which is equal to the precipitation intensity is added to the water balance.

Then the water balance is assembled for all cells. Visible is that, in the water balance, the change of total water volume in each cell over time is a function of the inflow, the outflow and the amount of water that is extracted by the drainage system. For the lower cell this is different as here there is a possible outflow called *qbound*. This is however set to zero in the analysis. SWAP holds a large library of possible boundary conditions that could also be implemented here, such as a head-based outflow.

Following SWAP, the iteration loop on the solution of the head at t+1 is started. It is noted here that, although a mixed formulation is used, the system is solved for the heads. Other state variables will follow from that. The number of allowed iterations is increased by a factor 2 if the the time step is equal to the minimum allowed time step. Otherwise the default value is applied. The loop over the iterations is started by saving the head from the previous solution step and it calculates the differential moisture capacity. After that the jacobian can be finished and the middle diagonal is calculated. First for the upper cell, than for all other cells finishing with the lower cell. The formulation of the jacobian is also treated in the documentation of SWAP. After the respective vectors/arrays have been calculated, the *np.diag* Python function is used to transform the vector to a diagonal in the matrix. The jacobian is then assembled by summing the different diagonals. In order to avoid any unnecessary computational effort, the scipy.sparse.csc_matrix is applied to store it as a sparse matrix (the matrix is sparse as it is only consisting of three diagonals). The *spsolve* function will solve for the jacobian and the water balance, to obtain the heads at the new solution step. This is a significant and more stable improvement to SWAP. The solution is the *difh* and as the name suggests, is the difference between the head in this time step and that of the previous one.

Now, as the solution will not likely have converged yet (iterative solution procedure), a Newton-Raphson solution scheme is applied. Possible back-tracing is implemented to avoid that the scheme will diverge from the solution. This is something that often happens in Newton-Raphson schemes with unknown values of the derivative function. The first part of the new loop will calculate the reduction factor of *difh* to be applied to the solution scheme. A new value of the head is calculated together with the corresponding values of all other state variables. As the hydraulic conductivity is calculated to be an explicit function of the head in the previous time step, this state variable is not updated here. This could however be implemented in future applications of the study and code. The new water balance is calculated and back-tracing is applied as long as that leads to better solutions of the water balance (*sumIt<sumold*) or an overall convergence parameter is satisfied. Otherwise the difference in head is divided by three again.

Once the back-tracing has been performed to satisfaction, it is checked whether the solution has converged. This is done for the water balance deviation per compartment, the change of head with time step (this may not be too large), and the total error in the water balance. If the criteria are satisfied, the solution has converged, otherwise a new solution iteration is started. If the maximum number of iterations has been performed and the solution has not yet converged, the soil state variables are reset and the time step is started again, but then with a smaller value according to the function *decreaseTime*. If the solution has not converged, and the minimum time step is applied already, an exception will be raised that convergence could not be reached and it will stop the simulation. Then alteration of the convergence parameters, increase in the number of iterations or a decrease of the minimum time step is needed.

```
def headCalc(h,hmin1,hold, theta, relSat,thetamin1,qdra,dimoca,dFdhM,
             nNode,dFdhU,dFdhL,F,hGrad,kMean,kNode, bound,mvgn,pondm1,pond,
             pondVar,dZ,disnod, dT, dTMin,MaxIt,maxBackTr,CritDevBalCp,
             CritDevh2Cp,CritDevh1Cp,CritDevBalTot, FlNonConv, Fldecdt,FlRain):
    ''' Calculation of the pressure heads, water contents and conductivities
    for the next time step'''

    # Reset conductivities to time level t
    kNode,relSat = hConduc(theta, relSat,kNode, nNode, mvgn)
    kMean = hConducMean(kNode, kMean, nNode, dZ)
    kMean[nNode] = kMean[nNode-1]


    #Calculation of the dFdhU and dFdhL
    for i in range(1,nNode):
        dFdhU[i] = -kMean[i]/disnod[i]
        dFdhL[i-1] = dFdhU[i]

    #Calculation of the gradient vector
    for i in range(1,nNode):

        hGrad[i] = (h[i-1]-h[i])/disnod[i]+1.

    #First entry in the F vector
    F[0] = (theta[0]-thetamin1[0])*dZ[0]/dT+kMean[1]*hGrad[1]+qdra[0]

    #Call the function for the top boundary
    Fltoph,q1,pond,runots,hsurf = boundTop(h,kNode,kMean,dZ,disnod,dT,bound,mvgn,pondm1,pond,
    pondVar,FlRain)
    #print('Pond level is', pond)
    #print('Runoff is', runots)
    # Further assembly of the F vector
    if Fltoph == True:
        hGrad[0] = (hsurf-h[0])/disnod[0]+1.0
        F[0] = F[0]-kMean[0]*hGrad[0]
    else:
        F[0] = F[0]+q1

    for i in range(1,nNode-1):
        F[i]=(theta[i]-thetamin1[i])*dZ[i]/dT-kMean[i]*hGrad[i]+kMean[i+1]*hGrad[i+1]
```

```
            +qdra[i]

    F[-1] = (theta[-1]-thetamin1[-1])*dZ[-1]/dT-kMean[-1]*hGrad[-1]-bound['qBot']
            +qdra[-1]

    #Initial estimate of F inner product
    sumold = 0.0
    for i in range(nNode):
        sumold = sumold+F[i]*F[i]
    sumold = 0.5*sumold

    #Start iteration loop, maximum number of iterations is specified in the input
    if dT == dTMin:
        MaxIt1 = 2*MaxIt
    else:
        MaxIt1 = MaxIt

    sumIt = 0.0 # 'For Forcheck'???

    for numbit in range(1,MaxIt1):

        for j in range(nNode):
            hold[j] = h[j] # save values of h, probably not in the final list
            dimoca = moisCap(h,theta,dimoca,nNode,dT,mvgn)

        #Formulation of the Jacobian

        #Main diagonal
        dFdhM[0] = dimoca[0]*dZ[1]/dT-dFdhL[0] #dZ[1] is chosen as the compartment
                                                #thickness is to be used here
        if Fltoph == True:
            dFdhM[0] = dFdhM[0]+kMean[0]/disnod[0]

        for k in range(1,nNode-1):
            dFdhM[k]= dimoca[k]*dZ[k]/dT-dFdhU[k]-dFdhL[k]

        dFdhM[-1] = dimoca[-1]*dZ[i]/dT-dFdhU[-1]

        #Further formulation of the entire matrix
        adiag = np.diag(dFdhM,0)
        bdiag = np.diag(dFdhU[1:],-1)
        cdiag = np.diag(dFdhL[:-1],1)

        jac = adiag +bdiag+cdiag
        jac = scipy.sparse.csc_matrix(jac)

        #Solving the matrix system
        difh = spsolve(jac,F)



        #Running the Newton Raphson system with backtrace
        factor = 1.0
        iBackTr = 0.0
        for itry in range(maxBackTr):
            iBackTr = iBackTr+1
            #'Factor reduces the change of h (difh) calculated as a full Newton Raphson step'

            if (dT == dTMin and numbit>MaxIt):
                factmax = 0.0
                for i in range(nNode):
                    if (abs(hold[i])<1.):
                        factmax = max(factmax,abs(difh[i]))
                    else:
                        factmax = max(factmax,abs(difh[i]/hold[i]))

                for i in range(nNode):
                    h[i]=hold[i]-difh[i]*min(1.0,1.0/factmax)
            else:
                for i in range(nNode):
                    h[i]=hold[i]-factor*difh[i]

            for i in range(nNode):
                theta=watCon(h,theta,nNode,mvgn)

            for i in range(1,nNode):
                hGrad[i]=(h[i-1]-h[i])/disnod[i]+1.0

            #Calculate the F function again
            F[0] = (theta[0]-thetamin1[0])*dZ[0]/dT+kMean[1]*hGrad[1]+qdra[0]

            #Call the function for the top boundary
            Fltoph,q1,pond,runots,hsurf =boundTop(
            h,kNode, kMean,dZ,disnod,dT,bound,mvgn,pondm1,
                                pond,pondVar,FlRain)

            # Further assembly of the F vector
            if Fltoph == True:
                hGrad[0] = (hsurf-h[0])/disnod[0]+1.0
                F[0] = F[0]-kMean[0]*hGrad[0]
            else:
                F[0] = F[0]+q1

            for i in range(1,nNode-1):
                F[i]=(theta[i]-thetamin1[i])*dZ[i]/dT-kMean[i]*hGrad[i] \
                    + kMean[i+1]*hGrad[i+1]+qdra[i]

            F[-1] = (theta[-1]-thetamin1[-1])*dZ[-1]/dT-kMean[-1]*hGrad[-1] \
                    -bound['qBot']+qdra[-1]

            #Calculate maximum deviation per compartment and new inner product
            FMax = 0.
            sumIt = 0.
            sum1 = 0.

            for i in range(nNode):
                if(abs(F[i])>FMax):
                    FMax = abs(F[i])
                sumIt = sumIt+F[i]*F[i]
                sum1 = sum1+F[i]
            sumIt = 0.5*sumIt

            if sumIt < sumold or FMax < CritDevBalCp:
                break
            else:
                factor = factor/3.0

        #Continuing after the Newton Raphson loop
        #Check on convergence of solution

        #Apply performance criteria per compartment
        for i in range(nNode):

            #Test for water balance deviation per soil compartments
            if (abs(F[i])>CritDevBalCp):
                FlNonConv = True
            elif (abs(hold[i])<1.0 and abs(h[i]-hold[i])>CritDevh2Cp):
                FlNonConv = True
            elif(abs(h[i]-hold[i])/abs(hold[i]) > CritDevh1Cp):
                FlNonConv = True
            elif abs(sum1) > CritDevBalTot:
                FlNonConv = True
            else:
                FlNonConv = False

        # Save sum for next iteration and store number of iterations needed
        sumold = sumIt
        nIt = numbit

        if FlNonConv == False:      # Convergence has been reached
            Fldecdt == False
            return h, theta, kNode, kMean, Fldecdt, numbit

    # Maximum number of iterations has been reached
    if FlNonConv == True and dT != dTMin:
        #Reset soil state variables:
            for j in range(nNode):
                h[j]=hmin1[j]
                theta[j] = thetamin1[j]
            kMean[-1] = kMean[-2]
            pond = pondm1

            #Continue with smaller timestep
            Fldecdt == True

            return h, theta, kNode, kMean, Fldecdt, numbit
    if FlNonConv == True and dT == dTMin:
        raise Exception('Convergence could not be reached, programme was stopped')
```

When the heads at the new time step are known, the *soilWaterRate* function can be applied. This function will calculate the flux or change of water content inside each cell. This rate does however not serve any function in the rest of the programme and does not describe the flux from cell to cell (essentially the discharge). A function could be implemented that does this, but was not done here. Basically, some values of the *headCalc* function should be extracted for that. These are the gradient of the head and the mean hydraulic conductivity.

```
def soilWaterRate(h,theta,relSat,thetamin1,q,kNode,kMean,nNode,dZ,dT,mvgn,bound):

    #Update hydraulic conductivities to time level t+1
    kNode,relSat = hConduc(theta, relSat, kNode, nNode, mvgn)
    kMean = hConducMean(kNode, kMean, nNode, dZ)
    kMean[-1] = kNode[-1]

    #Calculate the fluxes from changes in volume per compartment

    i = nNode
    q[i]= bound['qBot']
    for i in range(nNode-1,1,-1):
        q[i] = -(theta[i]-thetamin1[i])*dZ[i]/dT

    return kNode,relSat, kMean, q
```

The *boundTop* function will determine the type of boundary condition that is applied at the infiltration boundary. As stated many times, the infiltration of rainfall with intensities higher than the saturated hydraulic conductivity needs the switching between discharge- and head-based boundary conditions. At first, it calculates the discharge *q1* based on the value of the pondlevel in the previous time step. Then, assuming maximum saturation and maximum conductivity at the top, it performs a guess value for the head in the first cell. If this value is smaller than a threshold value of 1e-6, the q1 with rainfall discharge is returned. The parameter FlRain denotes whether it rains or not and will be 'True' in the period of rainfall and otherwise 'False' which returns 0 discharge.

If the value of *h0* is however larger than this threshold value, ponding occurs. It then calculates the new ponding level based on some simple water balance calculations. If this value *h0max* is larger than the maximum pond level, runoff will need to take place. This run-off function is a simplified version of what is implemented in SWAP. Here, no resistance to run-off is applied as to make the pond level truly the maximum pond level. Otherwise it will rise above. Again, the values are based on simple water balance and hence some discrepancies can occur, allowing the pond level to be higher than the maximum or be lower and still have run-off. This is however believed to be of little importance to the final solution.

```python
def boundTop(h, kNode, kMean, dZ,disnod, dT, bound,mvgn, pondm1,pond,pondVar,FlRain):
    '''Determine soil profile top boundary condition, calculates top flux,
    potential ponding height and runoff'''
    # General procedure for calculation of whether runoff will occur or not
    if pondm1 == 0:
        q1 = bound['qRain']*FlRain
    elif pondm1 != 0:
        q1 = bound['qRain']*FlRain-pondm1/dT


    #Calculate maximum conductivity assuming saturation at ground surface (z=0)
    k0max = (dZ[0]*mvgn['kSat']+dZ[0]*kNode[0])/(dZ[0]+dZ[0])
    #print(k0max)
    h0 = h[0]-disnod[0]*(q1/k0max+1)


    if h0 < 1e-6:
        kMean[0] = 0
        pond = 0
        hsurf = 0
        runots = 0
        Fltoph = False
        return Fltoph, q1, pond,runots,hsurf
    else:
        kMean[0] = k0max
        Fltoph = True

    #Calculation of runoff and ponding level
    if Fltoph == True:
        p1 = k0max/disnod[0]*dT
        p2 = 1./(p1+1.)
        h0max = p2*(pondm1-bound['qRain']*dT*FlRain-k0max*dT+p1*h[0])


        if h0max <= pondVar['pondmax']:
            runots = 0.
            pond = h0max
            hsurf = pond
            q1 = 0
            return Fltoph, q1, pond, runots, hsurf
        else:
            # Only chosen to program the situation for which the exponent in the runoff
            #equation is equal to
            # one. This means the equation simplifies and no iterative
            #procedure is needed.
            p1 = k0max/disnod[0]*dT
            p2 = 1.0/(p1+1.0+dT/pondVar['rsro'])

            pond = p2*(pondm1-bound['qRain']*dT*FlRain-k0max*dT+p1*h[0]+dT/
            pondVar['rsro']*pondVar['pondmax'])

            runots = dT/pondVar['rsro']*(pond-pondVar['pondmax'])
            hsurf = pond
            q1  = 0

            return Fltoph, q1, pond, runots, hsurf
```

As stated in the main matter, a drainage package is applied in order to mimic horizontal flow of water. First, the total drainage will need to be calculated. It does this on the basis of some calculated value of the groundwater level and applying the formulae as earlier provided in this thesis, the total drainage needed to lower the phreatic surface is computed. After that, the drainage is equally distributed over the cells below the groundwater level.

```
def drainage(gwl,compGwl, draindic,nNode):
    resEntry = 0.

    #Specifying the head difference
    dh = (gwl-draindic['gwlDra'])
    dhres = (gwl)

    totres = draindic['LDrain']**2/(4*draindic['kh']*dhres*(dh>0))+resEntry


    if dh > 0:
        qDrain = dh/totres
    else:
        qDrain = 0


    #Building the discharge vector
    qdra = np.zeros(nNode)

    #Calculation of total compartments:
    totDrainComp = nNode-compGwl

    for i in range(compGwl,nNode):
        qdra[i] = qDrain/totDrainComp

    return qdra
```

The final function is that of the calculation of the groundwater level. It is a standard loop which moves from the bottom of the column upwards. Once the pressure head inside a cell is not positive anymore, the loop is stopped and the groundwater level is calculated on the basis of the pressure head value in that compartment. The compartment which holds the groundwater level is also returned.

```
def calcgwl(h,nNode,hCol):

    ThickComp = hCol/nNode

    for i in range(len(h)-2,0,-1):
        if h[i] < 0.:
            compGwl = i
            gwlsur = ThickComp*i+ThickComp-h[i+1]+0.5*ThickComp
            gwl = hCol-gwlsur
            break
        elif i ==1:
            compGwl = 0
            gwlsur = 0
            gwl = hCol-gwlsur

    return gwl, gwlsur, compGwl
```

This concludes the treatment of the PySWAP code.

D.4. Slope stability analysis

The final slope stability algorithm implemented was a limit equilibrium calculation of a non-circular failure surface, applying the Spencer formulation of the equilibrium of slices. Furthermore, the unsaturated strength was taken into consideration. The critical failure surface finder was implemented as a differential evolution algorithm. The following issues are thought of to be important to elaborate upon:

- generating kinematically feasible slip surfaces;
- the differential evolution algorithm applied;
- calculation of the factor of safety for unsaturated/saturated soils;

D.4.1. Generating kinematically feasible slip surfaces

One of the more common issues for application of non-circular slip surfaces is the fact that the generated slip surfaces are kinematically not feasible. This means that the slip surfaces holds extremely sharp corners, are concave rather than convex or that the wedges have slope angles at the entry and exit points which are respectively larger than the active wedge or passive wedge of a soil. The latter means that the normal force at the slice base may become a tension force which is not appropriate for soils. Implemented in the Python code is the algorithm as proposed by Cheng (2003). For completion purposes, a summary of this algorithm is provided. The algorithm by Cheng (2003) is perfectly suitable for the application in genetic algorithms, by the use of the so-called control variables. The workings of the algorithm are largely visualised in Figure D.1.
The first two important variables of the failure surface are the entry and exit points. In the aforementioned figure noted as respectively B and A. The algorithm will now be explained in a step-wise manner:

1. Typical in limit equilibrium analysis is the fact that the horizontal width of the slice is a constant, meaning that at first, the x-coordinates of all of the interior slice points can be calculated based on the horizontal distance between B and A. In this case the x-coordinates of C, D, E and F.

2. The entry and exit point, B and A, are connected with each other by a line. The point on this line, directly above C is named C1. C1 is an upper bound to the y-coordinate of point C. However, to make this kinematically feasible, C1 is the lower coordinate of either the slope profile or the point on the line between B and A. In this case, the slope is governing.

3. The lower bound of C is determined by Cheng (2003) to be equal to a distance of AB/4 below point C1. It is noted that this may also e.g. be AB/5 and does not show large influences on the results.

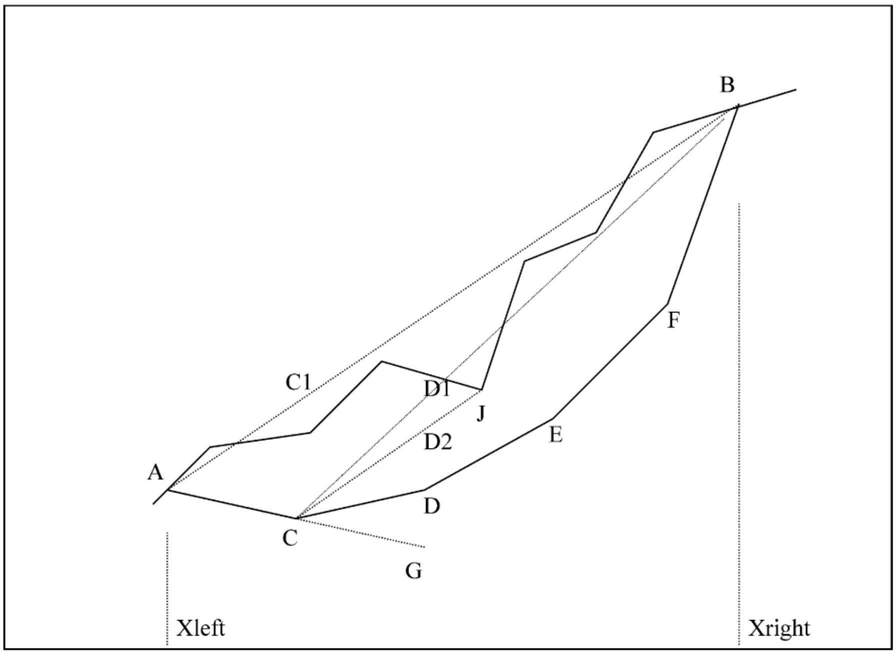


Figure D.1: The formation of the critical failure surface based on the Cheng algorithm (Cheng, 2003)

4. Once a position is chosen of C along the upper and lower bound that was mentioned before, A and C are connected to each other by a line. When extrapolating this to the x-coordinate of point D, we obtain a lower bound for the slice point of D. In this way, a convex shape is always maintained. This point is point G in the figure.
5. The upper bound (D1) is determined in exactly the same way as was done for point C1.
6. This procedure can be repeated for all of the slice points in order to provide bounds to all of the slice points. This can then be used in the control variables of the differential evolution algorithm.

This algorithm was fully implemented in Python and part of the differential evolution library in the source code. The algorithm is given in the functions *FuncSlicexCoord* and *FuncSliceyCoord*. There are a few additional aspects that are needed in order to have this algorithm fully function. These aspects are mainly related to the slope geometry and the position of any changes in the slope angle. It is necessary to determine the y-coordinate of the slope for the x-coordinate of the slice. This needs the functions *FuncSlopeData* and *FuncSectionFinder*. The slope geometry is given by the coordinates of the vertices of this geometry. The first function computes the angle and y-intercept of each section which then allows for a mathematically accurate description of the slope. The second function evaluates for an x-coordinate to which section it belongs so that, given the slope and y-intercept, the y-coordinate of the slope for the given x-position can be defined.

There are a few additional things which were incorporated into the *FuncSliceyCoord* function which had mainly to do with the number of slices within a LEM computation. As stated, a differential evolution scheme is employed. This differential evolution scheme optimises for the control variables the lowest factor of safety (i.e. the ‘highest’ fitness). These control variables were discussed above and consists of the entry and exit point of the failure surface, as well as a number for each slice cut between 0 and 1 to be defined along the kinematically feasible vertices. One may imagine that increasing the number of control variables will also increase the needed number of generations for optimisation, and the needed population size as the number of dimensions has increased. The solution space grows each time a new control variable is used. However, the application of limit equilibrium method needs a larger amount of slices in order to converge to the true factor of safety. Geo-slope international Ltd. (2015), recommends even a number of 30 slices to always be used. It is however impractical to optimise for 30 control variables as the factor of safety has to be computed a significant amount during the entire calculation process. In that sense, for each control variable slice a certain number of additional slices are added. These are slices that is not optimised for, but in this way, the number of slices can be increased by a significant factor without needing a large change to the optimisation process. The amount of 'interslices' was varied between 4 and 7, depending on the amount of control variable slices that was used.

In addition to that, there is the issue of different materials within the slope. Limit equilibrium methods (at least most of them) do not observe a material change along the bottom boundary of a slice, but assume that the material at the midpoint of the slice is present along the entire base. Within optimisation methods such as DE, this results in the fact that the optimiser always converges to having midpoints of slices along boundaries just at the weaker material side. In order to solve for this issue, the material boundaries were modelled as well. This meant that the dam boundary as well as the boundary for which the degree of saturation equals 0.9 were modelled. Then for each slice it was checked if the slice crosses such a boundary. If that was the case, an additional slice cut was implemented at the boundary. This effectively means that along each slice base, the material is always the same. Especially for the boundary along the degree of saturation, this proved to be an important aspect in finding the correct factor of safety.

The following sections will discuss the functions as mentioned above and provide with the source code.

FuncSliceXCoord

This function simply returns the x-coordinates of the slice cuts from the control variables only. In this case, *x1* and *xn* refer to the entry and exit position of the failure surface respectively. The input variables are the *controlVar* which is the entire vector which resembles the failure surface. *noSlicesNonCirc* is the number of slices for a non circular failure surface. This refers to the amount of control variables implemented rather than the total number of slices. By using the linear space function from the *numpy* library, the function returns equally spaced slice cuts with a number of *noSlicesNonCirc+1*, as the number of slice cuts equals that of the number of slices plus one.

```python
def FuncSlicexCoord(controlVar, noSlicesNonCirc):
    x1 = controlVar[0]
    xn = controlVar[-1]
    xSliceCut = np.linspace(float(x1),float(xn),noSlicesNonCirc+1)

    return xSliceCut
```

The function *FuncSliceyCoord* is the core of the entire slice formulation. Throughout the various versions it has grown substantially in size and future improvements of the code could define various functions for the separate parts. The code starts with finding the y-coordinates of the entry and exit points along the slope. First, the function *FuncSectionFinder* is used to find the 'section' which holds the x-coordinate in its domain. A section is defined as that part of the slope in between the vertices of the slope geometry. In most cases, only two sections were used. The flat part of the top of the impoundment and its sloping part. After the section was found, the mathematical description of these lines was used to find the y-coordinates of the entry and exit points.

Following this is the penultimate vertex as described previously with the algorithm of Cheng (2003). The maximum and minimum kinematically feasible y-coordinate is described and the control vector is then used to find the exact point. After that, a loop is initiated for all the other slices. The *polyfit* function of the *numpy* library is used in order to describe the line segment between two respective vertices. After this part, the *ySliceCut* vector holds the y-positions of the control vertices.

Then a small loop is initiated in order to divide these larger slices into smaller slices. In this case a single slice is cut into four new slices. This may however be changed. The insert function is used to provide with the new x- and y- coordinates based on simple linear interpolation.

Then the material boundaries will need an additional slice cut. This is done in a single loop for which first, the dam boundaries are checked, after which the boundary for liquefaction or no liquefaction of the tailings is defined and used. The *shapely* package is used for this as it allows for defining lines and can easily calculate their intersection, with verification if the intersection lies within the domain of the two stretches. The loop iterates between all slice cuts and defines whether the material boundary is crossed. If this is the case, at this intersection, a new cut is inserted. For the dam this procedure is easy. For the material boundary belonging to the degree of saturation, this material boundary has to be found. For this the *gridSe* is used which holds the relative degree of saturation. The height for which the degree of saturation crosses the value of 0.9 is defined and it is observed whether the slice base crosses this line. If this is the case, a new insertion point is used. The material boundary truly has to be defined in this case. First, it was implemented that the values of the degree of saturation at the vertices could be directly used to calculate the position of the additional cut, but as the degree of saturation is capped at 1, this delivered poor results.

Finally, the *slicecut* coordinates are used to calculate the midpoints, which will be used in the factor of safety calculation.

```
def FuncSliceyCoord(controlVar,xSliceCut,slopeData,noSlicesNonCirc,topPoints, DL,gridSe,grid):
    magX = 10

    #Defining the y-coordinates of the entry and exit points on the slope
    sectionCut = np.zeros(len(xSliceCut), dtype = int)
    for i in range(len(xSliceCut)):
        sectionCut[i] = FuncSectionFinder(slopeData,xSliceCut[i])

    ySliceCut = np.zeros(len(xSliceCut))
    ySliceCut[0] = xSliceCut[0]*slopeData[sectionCut[0],0]+slopeData[sectionCut[0],1]
    ySliceCut[-1] = xSliceCut[-1]*slopeData[sectionCut[-1],0]+slopeData[sectionCut[-1],1]


    # Control variable for size of the possible solutions
    diffY = (ySliceCut[0]-ySliceCut[-1])/4

    #Define y coordinate of penultimate vertex
    lineAB = np.polyfit((xSliceCut[0],xSliceCut[-1]),(ySliceCut[0],ySliceCut[-1]), deg =1 )
    yMaxPenAlg = xSliceCut[-2]*lineAB[0]+lineAB[1]
    yMaxPenSlope = xSliceCut[-2]*slopeData[sectionCut[-2],0]+slopeData[sectionCut[-2],1]
    yMaxPen = min([yMaxPenAlg,yMaxPenSlope])

    yMinPen = max([yMaxPen-diffY,0])

    ySliceCut[-2] = (yMaxPen-yMinPen)*controlVar[-2]+yMinPen

    #Define y-coordinate for all other vertices in a loop
    for i in range(noSlicesNonCirc-2,0,-1):
        lineI = np.polyfit((xSliceCut[0],xSliceCut[i+1]),(ySliceCut[0],ySliceCut[i+1]),
        deg =1 )

        #Calculating maximum of control variable
        yMaxIAlg = xSliceCut[i]*lineI[0]+lineI[1]
        yMaxISlope = xSliceCut[i]*slopeData[sectionCut[i],0]+slopeData[sectionCut[i],1]
        yMaxI = min([yMaxIAlg,yMaxISlope])

        #Calculating minimum of control variable
        linePrev = np.polyfit((xSliceCut[i+1],xSliceCut[i+2]), (ySliceCut[i+1],ySliceCut[i+2]),
        deg =1)
        yMinDiffPrev = xSliceCut[i]*linePrev[0]+linePrev[1]
        yMinI = max([yMaxPen-diffY,0,yMinDiffPrev])

        ySliceCut[i] = (yMaxI-yMinI)*controlVar[i]+yMinI

    #Copy as to not change the original cut function for loops
    xSliceCutBetween = np.copy(xSliceCut)
    ySliceCutBetween = np.copy(ySliceCut)

    #Create interslices between control variables. Now, one slice is split into four slices.
    for i in range(len(ySliceCut)-1):
        xNewSlice = np.linspace(xSliceCut[i],xSliceCut[i+1],5)
        yNewSlice = np.linspace(ySliceCut[i], ySliceCut[i+1],5)

        xSliceCutBetween = np.insert(xSliceCutBetween,1+i*4,xNewSlice[1:4])
        ySliceCutBetween = np.insert(ySliceCutBetween,1+i*4,yNewSlice[1:4])

    pADam = (topPoints[1,0]-DL.damWidth,topPoints[1,1])
    pBDam = (topPoints[-1,0]-DL.damWidth,topPoints[-1,1])
    damLine = LineString([pADam,pBDam])
    xSliceCutFinal = np.copy(xSliceCutBetween)
    ySliceCutFinal = np.copy(ySliceCutBetween)
    counter = 0

    #Create additional slices along material boundaries.
    for i in range(len(ySliceCutBetween)-1):
        pASlice = (xSliceCutBetween[i],ySliceCutBetween[i])
        pBSlice = (xSliceCutBetween[i+1],ySliceCutBetween[i+1])
        sliceLine = LineString([pASlice,pBSlice])

        intPt = damLine.intersects(sliceLine)
        intPtLine = damLine.intersection(sliceLine)
        if intPt:
            xSliceNew = np.array([float(intPtLine.x)])
            ySliceNew = np.array([float(intPtLine.y)])

            if xSliceNew[0] != xSliceCutBetween[i] and xSliceNew[0] != xSliceCutBetween[i+1]:
                xSliceCutFinal = np.insert(xSliceCutFinal,i+1+counter,xSliceNew)
                ySliceCutFinal = np.insert(ySliceCutFinal,i+1+counter,ySliceNew)
                counter +=1

    #for i in range(len(ySliceCutFinal)-3):

        if i < len(ySliceCutFinal)-3:
            idxxi = int(round(xSliceCutFinal[i],1)*magX)
            idxyi = min(int(round(ySliceCutFinal[i],1)*magX),topPoints[0,1]*magX-1)

            idxxplusi = min(int(round(xSliceCutFinal[i+1],1)*magX), topPoints[-1,0]*magX-1)
            idxyplusi = min(int(round(ySliceCutFinal[i+1],1)*magX),topPoints[0,1]*magX-1)

            aSe = gridSe[idxyi,idxxi]
            bSe = gridSe[idxyplusi,idxxplusi]
            ySei = 0
            ySeplusi = 0
            if aSe <0.9 and bSe> 0.9:
                for j in range(len(gridSe[:,idxxi])-1):

                    if gridSe[j,idxxi]>=0.9 and gridSe[j+1,idxxi] <0.9:
                        ySei = j
                        break
                for j in range(len(gridSe[:,idxxplusi])-1):
                    if gridSe[j,idxxplusi] >= 0.9 and gridSe[j+1,idxxplusi]<0.9:
                        ySeplusi = j
                        break

                pASlice = (xSliceCutFinal[i],ySliceCutFinal[i])
                pBSlice = (xSliceCutFinal[i+1],ySliceCutFinal[i+1])
                sliceLine = LineString([pASlice,pBSlice])

                pSei = (idxxi/magX,ySei/magX)
                pSeplusi = (idxxplusi/magX,ySeplusi/magX)
                sliceSeLine = LineString([pSei,pSeplusi])

                intPtSe = sliceSeLine.intersects(sliceLine)
                intPtSeLine = sliceSeLine.intersection(sliceLine)

                if intPtSe:
                    xSliceNew = np.array([float(intPtSeLine.x)])
                    ySliceNew = np.array([float(intPtSeLine.y)])

                    xSliceCutFinal = np.insert(xSliceCutFinal,i+1,xSliceNew)

                    ySliceCutFinal = np.insert(ySliceCutFinal,i+1,ySliceNew)
                break

    #Calculate the midpoints of the final slices
    yMidPoints = np.zeros(len(xSliceCutFinal)-1)
    xMidPoints = np.zeros(len(xSliceCutFinal)-1)
    for i in range(len(xSliceCutFinal)-1):
        yMidPoints[i] = (ySliceCutFinal[i]+ySliceCutFinal[i+1])/2
        xMidPoints[i] = (xSliceCutFinal[i]+xSliceCutFinal[i+1])/2
    return ySliceCutFinal , yMidPoints,xSliceCutFinal,xMidPoints
```

The following function was used for determining the correct slope section.

```
def FuncSectionFinder(slopeData, x):
    ''' Function for finding the correct section to be used for computing y-values of
    coordinates with corresponding x-values'''
    for i in range(len(slopeData[:,0])):
        if x>=slopeData[i,2] and x<=slopeData[i,3]:
            break
    return i
```

D.4.2. The differential evolution algorithm

Differential evolution is part of the family of the genetic/evolutionary algorithms. These algorithms are commonly applied in optimisation problems and are therefore of use for finding the critical slip surface. This method was favoured over more common algorithms such as the grid-search algorithm, also employed in slope stability, as this requires continuous updating by the user and was therefore not favoured by the author. The differential evolution will represent the variables as some 'gene' for which the function (in this case the FoS calculator) is evaluated. A initial 'population' is formed consisting of multiple randomly generated genes. For all of these genes the FoS is calculated and the best performing genes will move to the next generation. The idea is that the population converges to an optimal performing gene for which the factor of safety is lowest. By the means of randomly occurring mutations and cross-overs of the genes, one avoids that the solution converges to a local optimum rather than a global optimum.

The definition of the gene is in this case simple. The gene should hold two values for the x-coordinate of the entry and exit point. Moreover, it should hold a value between 0 and 1 defining the position of each slice point along the upper and lower bound as explained in section D.4.1. This means that the number of values in the gene is the number of slices plus one.

There is a wide variety of possibilities to apply the mutations and cross-overs. The code is largely inspired by the work of Rodriguez-Mier (2017), which is a very simple yet effective form of the differential evolution with the rand/1/bin scheme. The rand/2/bin scheme has also been applied, as was suggested in Rickard and Sitar (2012). This scheme did however not perform better than the rand/1/bin scheme and was neglected.

The function starts with defining the first population. This is done with the function *FuncFirstPopulation* which will be treated afterwards. Important to note already is that any differential evolution algorithm is stochastic of nature, meaning that each new optimisation will yield different results because of the different starting population. Following the generation of the first population, the fitness of each population member is calculated by starting the *FuncFullFSCalculator* function. This function holds the process from the population member or control vector towards the factor of safety. Returned and stored are the factors of safety for each of the population members. The best performing member is the one with the lowest factor of safety.

After the FoS of the initial population has been calculated, the optimisation in the differential evolution algorithm is started. For each of the members, first a new list of indices of the population is made (*idxs*) excluding the index of the population member *j*. Then from this new list, three indices are chosen at random. The members corresponding to these indices are then used to make a new mutant vector following this rand/1/bin scheme. The name differential evolution is directly inherited from this procedure as the difference between chosen members is used. After the mutant has been formed, the *np.clip* function is used to make sure that the new values of the mutant are within the ranges set for the control variables. This means, for the x-values of the entry and exit point of the failure surface and the other variables between 0 and 1.

Before the new trial vector is imposed, first it is seen whether the new mutant vector will cross-over with the original population member. This is based on the probability defined by the variable *crossp* and the index at which cross-over may take place is chosen at random. Either way, the mutant changes to a trial vector for which the FoS is calculated. If this fitness is better, i.e. lower, than the new trial vector will replace the old member within the population. If this is not, the original member is maintained. After this procedure has been done for all of the members, a new generation is started.

An important issue within any optimisation algorithm is when to stop the process. The true factor of safety is unknown and in this sense it is hard to set this endpoint. Most slope stability software, including that of Rickard and Sitar (2012), impose a maximum number of generations to be used. However, this does not state anything about the quality of the solution. There are various ways in which other stopping criteria can be used. Zielinski and Laur (2008) provides with different stopping criteria of which the *MovPar* method was used as a movement-based criterion, based on this text. This means that the average movement of the population needs to decrease below a certain treshold for a set number of generations. Using this convergence based criterion makes sense. As with increasing number of generations, the population should have a lower FoS. When there are hardly any trial vectors that perform better, or the improvement is only very low, the solution has basically converged and new generations are no longer needed. An interpretation of this method has been implemented in the function.

Following the function, there are some implicit parameters that are used here. These parameters are *mut, crossp, avgMovement* and *stopCounter*. The values of these parameters are defined within the function. Although this may be reckoned as to not be neat programming, some sensitivity analyses were performed for these parameters and changing is not advised. The parameter *mut* defines the way in which the mutant vector is formulated and has a usual range between 0.2 and 0.6. For the cross-over probability in this case a value of 0.7 is used, but could be a bit lower. The *avgMovement* value was adopted from Zielinski and Laur (2008) and in a sensitivity analysis, a value of 22 generations was found to be a good optimum between number of generations needed and the quality of the solution.

```
def FuncOwnDe(topPoints,phreaticSurface,gridSe,layerData,LL, DL,
xStartRange,xEndRange,noSlicesNonCirc,popSize,numGenerations,axisPoint, magX, gridData,
mut = 0.5, crossp = 0.7, graphics = False):
    #Initiating the first population
    minGen = 0
    Population = FuncFirstPopulation(xStartRange,xEndRange,noSlicesNonCirc,popSize)
    dimensions = noSlicesNonCirc+1
    # plt.figure()
    #Computing the factor of safety for all of the genes for first trial vector
    fitness = np.zeros(popSize)
    for j in range(popSize):
        fitness[j] = FuncFullFSCalculator(topPoints,layerData,phreaticSurface,
        gridSe,LL,DL,axisPoint, noSlicesNonCirc, Population[j,:], magX,gridData,
        graphics = False)
    bestIndex = np.nanargmin(fitness)
    best = Population[bestIndex]

    stopCounter = 0
    popOld = np.zeros((popSize,noSlicesNonCirc+1))

    for i in range(numGenerations):
        #print('Generation number is ', i+1)
        # plt.plot(i,np.nanmin(fitness), 'o')
        # plt.plot(i,np.nanmean(fitness), '+')
        # plt.ylim([0.9,2.0])


        #plt.ylim([0,2])

        popOld = Population.copy()
        for j in range(popSize):

            idxs = [idx for idx in range(popSize) if idx != j]
            a,b,c = Population[np.random.choice(idxs,3,replace = False)]
            mutant = a+mut*(b-c)
            mutant[0] = np.clip(mutant[0], xStartRange[0], xStartRange[1])
            mutant[-1] = np.clip(mutant[-1], xEndRange[0],xEndRange[1])
            mutant[1:-1] = np.clip(mutant[1:-1], 0.001, 0.999)

            cross_points = np.random.rand(dimensions) < crossp
            if not np.any(cross_points):
                cross_points[np.random.randint(0, dimensions)] = True

            trial = np.where(cross_points, mutant, Population[j])

            FoS = FuncFullFSCalculator(topPoints,layerData,phreaticSurface,
            gridSe,LL,DL,axisPoint, noSlicesNonCirc, trial, magX,gridData,
            graphics = False)

            if FoS < fitness[j]:
                fitness[j] = FoS
                Population[j] = trial
                if FoS < fitness[bestIndex]:
                    bestIndex = j
                    best = trial
         #yield best, fitness[bestIndex]
        #Termination criterion based on MovPar: the average movement of the
        #population members is below treshold t for a given number of generations g,
        movement = abs(np.subtract(popOld,Population))

        avgMovement = np.nanmean(movement)

        #print(avgMovement)
        if avgMovement < 0.001:
            stopCounter +=1
        elif avgMovement >=0.001:
            stopCounter = 0

        if stopCounter == 22:
            break
        elif i == numGenerations-1:
            print('Convergence was not reached in DE, more generations needed for
            this day')
            break
    #print('Final generation number is ', i)
    if graphics == True:
       FSSpencer, xSliceCut, ySliceCut, xMidSlices, yMidSlices =
       FuncFullFSCalculator(topPoints,layerData,phreaticSurface,gridSe,LL,DL
       ,axisPoint,noSlicesNonCirc,best, magX, gridData, graphics = True)
       return best, FSSpencer, xSliceCut, ySliceCut, xMidSlices, yMidSlices
    elif graphics == False:
        return best, fitness[bestIndex]
```

The function *FuncFirstPopulation*, as stated, initiates the first population. Its code is given by:

```
def FuncFirstPopulation(xStartRange,xEndRange,noSlicesNonCirc,popSize):
    Population = np.zeros((popSize,noSlicesNonCirc+1))
    Population[:,0] = np.random.uniform(low = xStartRange[0], high = xStartRange[1],
    size = popSize)
    Population[:,-1] = np.random.uniform(low = xEndRange[0], high = xEndRange[1],
    size = popSize)
    Population[:,1:-1] = np.random.uniform(low = 0, high = 1, size
    =(popSize,noSlicesNonCirc-1))
    return Population
```

Basically a large array of the population size times the number of control variables is initiated. This means for the entry and exit point of the failure surface and the values along the kinematically possible y-position. Returned is this population to the differential evolution algorithm.

D.4.3. Calculation of the factor of safety

The calculation of the factor of safety is done through various functions, of which the function *FuncFullFSCalculator* is the main overlapping one that is initiated by the differential evolution algorithm. This function is given by:

```
def FuncFullFSCalculator(topPoints,layerData,phreaticData,gridSe,LL,
DL,axisPoint, noSlicesNonCirc, Population,magX, gridData,graphics = True):
    controlVar = Population

    xSliceCut = FuncSlicexCoord(controlVar, noSlicesNonCirc)
    slopeData = FuncSlopeData(topPoints)
    ySliceCut, yMidSlices,xSliceCut,xMidSlices =
    FuncSliceyCoord(controlVar,xSliceCut,slopeData,noSlicesNonCirc,topPoints,
    DL,gridSe,phreaticData)
```

```
FSBishop,FSSpencer,FSJanbu = FuncOlsonStark(topPoints,slopeData,layerData,
phreaticData,gridSe,LL,DL,xMidSlices,yMidSlices,xSliceCut,ySliceCut,
axisPoint, magX, gridData)

FSFound = FSSpencer

if FSFound <0.5:
    return 1000.

if graphics == True:
    return FSFound, xSliceCut, ySliceCut, xMidSlices, yMidSlices
elif graphics == False:
    return FSFound
```

Various functions are called upon. This includes the functions *FuncSlicexCoord* and *FuncSliceyCoord* that have been treated before. This section will elaborate on two other functions, the *FuncSlopeData* and the *FuncOlsonStark*. Note that in this functions, the factor of safety according to Bishop, Spencer and Janbu are provided, but only the Spencer factor of safety is passed back to the differential evolution algorithm. This avoids any serious problems with the optimisation.

Function *FuncSlopeData* provides with mathematical descriptions of the slope geometry:

```
def FuncSlopeData(SlopePoints):
''' For the slope points are given where the angle of the slope changes.
This accounts for the entire impoundment, meaning that it has to be
specified for the entire cross section from left to right. The
function will return the slope and intercept of the
linear interpolations between the points and analytically defines
the geometry '''

xPoints = SlopePoints[:,0]
yPoints = SlopePoints[:,1]
FuncSlopeData = np.zeros((len(xPoints)-1,4))
for i in range(len(xPoints)-1):
    FuncSlopeData[i,0], FuncSlopeData[i,1] =
    np.polyfit((xPoints[i],xPoints[i+1]),(yPoints[i],yPoints[i+1]),deg=1)

    FuncSlopeData[i,2] = xPoints[i]
    FuncSlopeData[i,3] = xPoints[i+1]
return FuncSlopeData
```

The more important function is the one of *FuncOlsonStark*. Although the name suggests that it is only the Olson and Stark method that may be applied, it is a more general limit equilibrium calculation method for the factor of safety. This function starts with initiation of various slice variables. Table D.3 holds the various variable names and their meaning.

Once these arrays, which are all of the same length as the number of slices, are initiated, a loop is started over the slices to calculate for each slice the corresponding values of the previously mentioned variables. The function *FuncGetGeomProp* is also used to find the values for f and R. After that the grid search is used. As known, the pressure head, as well as the effective degree of saturation are passed onto the function in grid form. Because of the resolution at which these are defined (every 0.1 m), no interpolation mechanism will be applied in between these grid points for the slice calculation. Otherwise, a lot of computational complexity would be added to the code, whilst the gain in accuracy is only very minimal. This is why the slice coordinates are rounded to a single decimal after which the index can be determined, that holds the corresponding head and saturation value.

The strength at the slice base may be one of three different types as explained in the main matter of this thesis. To differentiate between liquefiable and non-liquefiable tailings, the function *FuncStrengthLookup* is used which returns a 1 for non-liquefiable dam material and a 0 for liquefiable tailings. Then

Table D.3
Slice variables in the FoS calculation.

Variable name	Description
deltaX	width of each slice (horizontal distance)
section	to which section of the slope geometry the midslice point belongs
soilType	The soil type defined as 0 (liquefiable) or 1 (non-liquefiable)
beta	base length of the slice
f	offset from axis point (shortest perpendicular distance) to the line perpendicular to the slice base in the midpoint
R	distance from slice midpoint to the point on the line described by f
weightSlice	The weight in kN of the slice
sigmaEff	The effective stress (vertical) at the slice midpoint
Su	The undrained strength in kPa of the soil at the slice midpoint
Se	The effective degree of saturation at the slice midpoint
xDist	The horizontal distance between the slice midpoint and the axis-point for calculation
phiDrained	Friction angle at the slice midpoint
alpha	The slope of the slice base w.r.t. the horizontal
porePressure	Pore pressure in kPa at the slice midpoint
heightSlice	Height of the slice in meters.

based on the degree of saturation with a boundary of 0.9, an additional check is done whether the liquefiable tailings are saturated enough or not. The slice formulation finishes with providing the strength of each slice as either a cohesion type undrained shear strength or a friction angle at the slice base.

Finishing the slice formulation, the factor of safety can be calculated. In this case, the Bishop, simplified Janbu and Spencer FoS were implemented. Both Bishop and Janbu are determinate functions for slope stability meaning that, by some assumptions, the number of knowns is equal to the number of knowns. These stability formulations are however not complete in that sense that they do not satisfy all equilibrium conditions. This is the case for the Spencer formulation. That does however come at the cost of having an indeterminate formulation and an additional iteration procedure is therefore needed. Bishop and Janbu provide with proper input guesses for the Spencer FoS. Bishop satisfies the global moment equilibrium, whereas Janbu does this for the force equilibrium. Under varying angle of the inter-slice forces, there is one angle for which the moment and force FoS are equal. This is the Spencer FoS. This is why the Bishop and Janbu FoS are also calculated. Especially the Bishop one is important as it allows for a first starting point within the Newton-raphson scheme for finding the correct interslice force angle, as it assumes horizontal slice forces (Janbu also does that).

The explanation of calculation of the FoS has been partially treated in the main matter. Shown is that a Newton-Raphson scheme is implemented for finding the correct inter-slice force angle. Spencer is however a very instable formulation meaning that for some slip surfaces there is simply no angle for which the force and moment FoS is the same. This then results in divergence of the Newton-Raphson scheme. In that case, a high FoS is returned for the Spencer algorithm as to show that the solution is poor and does not have a good ‘fitness’ which will result in this population member being swapped with the trial version or that the trial version is not accepted.

Some other parameters are introduced here, including the maximum number of iterations that may be performed. The number of iterations seems to be low. However, in the sensitivity study it was shown that if the solution is to converge, 4 to 5 iterations are needed. Any iteration above that will only mean divergence of the solution and hence no value was seen to set a value larger than 7. This resulted in a significantly higher computational efficiency and reduced the average needed computational time for this function from 0.06 to 0.025 seconds.

```
def FuncOlsonStark(topPoints,slopeData,layerData,
phreaticSurface,gridSe,LL,DL,xMidSlices,yMidSlices,xSliceCut,ySliceCut, axisPoint,magX,
gridData):
    deltaX = np.zeros(len(xMidSlices))
    #Performing the slice building
    section = np.zeros(len(xMidSlices), dtype = int)
    soilType = np.zeros(len(xMidSlices), dtype = int)
    beta = np.zeros(len(xMidSlices))
    f = np.zeros(len(xMidSlices))
    R = np.ones(len(xMidSlices))


    weightSlice = np.zeros(len(xMidSlices))
    sigmaEff = np.zeros(len(xMidSlices))
    Su = np.zeros(len(xMidSlices))
    Se = np.zeros(len(xMidSlices))
    xDist = np.zeros(len(xMidSlices))
    phiDrained = np.zeros(len(xMidSlices))
    alpha = np.zeros(len(xMidSlices))


    porePressure = np.zeros(len(xMidSlices))
    heightSlice = np.zeros(len(xMidSlices))
    for i in range(len(xMidSlices)):
        #The geometric properties of f and R
        deltaX[i] = abs(xSliceCut[i+1]-xSliceCut[i])
        f[i],R[i] = funcGetGeomProp(axisPoint,xSliceCut[i], xSliceCut[i+1],
        ySliceCut[i],ySliceCut[i+1])

        section[i] = FuncSectionFinder(slopeData,xMidSlices[i])

        alpha[i] = -1*mt.atan(((ySliceCut[i+1]-ySliceCut[i])/
        (xSliceCut[i+1]-xSliceCut[i])))
        xDist[i] = axisPoint[0]-xMidSlices[i]
        beta[i] =  deltaX[i]/np.cos(alpha[i])


        ySlopeX = (xMidSlices[i]*slopeData[section[i],0]+slopeData[section[i],1])
        heightSlice[i] = ySlopeX-yMidSlices[i]
        weightSlice[i] = deltaX[i]*LL.gammaWet*(heightSlice[i])

        idxx = int(round(xMidSlices[i],1)*magX)
        idxy = min(int(round(yMidSlices[i],1)*magX),topPoints[0,1]*magX-1)

        if idxy < 0:
            porePressure[i] = -1000000
            Se[i] = 1
        else:
            porePressure[i] = phreaticSurface[idxy,idxx]
            Se[i] = gridSe[idxy,idxx]
        sigmaEff[i] = (weightSlice[i]/deltaX[i]-(porePressure[i]))*(porePressure[i]>0)

        #Try to find the soiltype of the element

        soilType[i] = FuncStrengthlookup(xMidSlices[i], yMidSlices[i], topPoints,DL,LL)

        if soilType[i] ==0 and yMidSlices[i]>0:
            if  Se[i]>0.9:
                Su[i] = LL.SuSigmaV*sigmaEff[i]
                phiDrained[i] = 0.
            elif Se[i]<0.9:
                Su[i] = 0
                phiDrained[i] = LL.phiSuc/180*np.pi
        elif yMidSlices[i] < 0:
            Su[i] = 100000000.
        elif soilType[i] == 1 and yMidSlices[i]>0:
            phiDrained[i] = DL.phiDrained/180*np.pi



    #Performing Bishop analysis
    Nm = np.zeros(len(xMidSlices))
    tol = 0.01
    FSm = 1.0
    itMax = 5
    for j in range(itMax):
        resistanceTotal = 0
        loadTotal = 0
        for i in range(len(xMidSlices)):
            ma =  (np.cos(alpha[i])+(np.sin(alpha[i])*np.tan(phiDrained[i]))/FSm)
            Nm[i] =  (weightSlice[i]-(Su[i]*beta[i]*np.sin(alpha[i]))/
            FSm+(porePressure[i]*beta[i]*np.sin(alpha[i])*Se[i]*np.tan(phiDrained[i]))/FSm)/
            ma

            resistanceTotal += (Su[i]*beta[i]*R[i] +
            (Nm[i]-porePressure[i]*beta[i]*Se[i])*R[i]*np.tan(phiDrained[i]))

            loadTotal += (weightSlice[i]*xDist[i]-Nm[i]*f[i])
            FScalc = resistanceTotal/loadTotal
        if abs(FScalc-FSm)<tol:
            FSm = FScalc
            break
        else:
            FSm = FScalc
    #print(j)
    FSBishop = FSm
    #print('FSBishop is,', FSBishop)

    #Performing Janbu analysis
    FSf = 1.0
    Nf = np.zeros(len(xMidSlices))
    itMax = 4
    for j in range(itMax):
            #print( 'FSf in iteration is', FSf)
            resistanceTotalf = 0
            loadTotalf = 0

            for k in range(len(xMidSlices)):
                Nf[k] =  (weightSlice[k]-(Su[k]*beta[k]*np.sin(alpha[k])-
                porePressure[k]*beta[k]*Se[k]*np.sin(alpha[k])*np.tan(phiDrained[k]))/FSf)/
                (np.cos(alpha[k])+(np.sin(alpha[k])*np.tan(phiDrained[k]))/FSf)

                resistanceTotalf +=
                Su[k]*beta[k]*np.cos(alpha[k])+(Nf[k]-porePressure[k]*beta[k]*Se[k])*
                np.tan(phiDrained[k])*np.cos(alpha[k])

                loadTotalf += Nf[k]*np.sin(alpha[k])

                FScalcf = resistanceTotalf/loadTotalf
            if abs(FScalcf-FSf)<tol:
                FSf = FScalcf
                break
            else:
                FSf = FScalcf
    FSJanbu = FSf

    LambdaOld = 0
    LambdaNew = 0.4
    FSmOld = 0.
    FSfOld = 0.
    FSf = 1.0
    itMax = 7
    ErminEl = np.zeros(len(xMidSlices))
    XrminXl = np.zeros(len(xMidSlices))

    for i in range(len(xMidSlices)):
        ErminEl[i] = -((Su[i]*beta[i]-porePressure[i]*beta[i]*Se[i]*np.tan(phiDrained[i]))
        *np.cos(alpha[i]))/FSBishop+Nm[i]*(-(np.tan(phiDrained[i])*np.cos(alpha[i]))/
        FSBishop+np.sin(alpha[i]))

        XrminXl[i] = ErminEl[i]*LambdaNew

    for i in range(itMax):
        #loop on total FoS
        #print('Total FoS in loop is,', FSmOld)
        for j in range(itMax):
            #print('FSm in iteration is', FSm)
            #loop on iteration for FSm
            resistanceTotal = 0
            loadTotal = 0
            Nm = np.zeros(len(xMidSlices))
            for k in range(len(xMidSlices)):
                ma = (np.cos(alpha[k])+(np.sin(alpha[k])*np.tan(phiDrained[k]))/FSm)

                Nm[k] = (weightSlice[k]-XrminXl[k]-(Su[k]*beta[k]*np.sin(alpha[k]))/
                FSm+(porePressure[k]*beta[k]*Se[k]*np.sin(alpha[k])*
                np.tan(phiDrained[k]))/FSm)/ma

                resistanceTotal += (Su[k]*beta[k]*R[k] +
                (Nm[k]-porePressure[k]*beta[k]*Se[k])*R[k]*np.tan(phiDrained[k]))

                loadTotal += (weightSlice[k]*xDist[k])-Nm[k]*f[k]

                FScalc = resistanceTotal/loadTotal

            if abs(FScalc-FSm)<tol:
                FSmOld = FSm
                FSm = FScalc
                break
            else:
                FSmOld = FSm
                FSm = FScalc

        for j in range(itMax):
            #print( 'FSf in iteration is', FSf)
            resistanceTotalf = 0
            loadTotalf = 0
            N = np.zeros(len(xMidSlices))
            for k in range(len(xMidSlices)):
                Nf[k] = (weightSlice[k]-XrminXl[k]-(Su[k]*beta[k]*np.sin(alpha[k])
                -porePressure[k]*beta[k]*Se[k]*np.sin(alpha[k])*np.tan(phiDrained[k]))/FSf)/
                (np.cos(alpha[k])+(np.sin(alpha[k])*np.tan(phiDrained[k]))/FSf)

                resistanceTotalf +=
                Su[k]*beta[k]*np.cos(alpha[k])+(Nf[k]-porePressure[k]*beta[k]*Se[k])*
                np.tan(phiDrained[k])*np.cos(alpha[k])

                loadTotalf += Nf[k]*np.sin(alpha[k])

                FScalcf = resistanceTotalf/loadTotalf
            if abs(FScalcf-FSf)<tol:
                FSf = FScalcf
                break
            else:
                FSf = FScalcf


        #Newton-Raphson scheme
        mM = (FSm-FSmOld)/(LambdaNew-LambdaOld)
        mF = (FSf-FSfOld)/(LambdaNew-LambdaOld)
        Lambdaint = LambdaNew-((FSm-FSf)/(mM-mF))

        if (abs(FSm-FSf)<tol and FSm>0):
            LambdaOld = LambdaNew
            LambdaNew = Lambdaint
            for i in range(len(xMidSlices)):
                XrminXl[i] = ErminEl[i]*LambdaNew

            if LambdaNew  < 0. or LambdaNew > 2.:
                FSSpencer = 1000.
            else:
                FSSpencer = FSm
                break
        elif abs(LambdaNew) > 10:
            FSSpencer = 1000. # No suitable Lambda has been found when this
                              # condition is met.
            break
        elif i == itMax-1:
            FSSpencer = 1000.
            break
        else:
            LambdaOld = LambdaNew
            LambdaNew = Lambdaint
            for i in range(len(xMidSlices)):
                XrminXl[i] = ErminEl[i]*LambdaNew
            FSmOld = FSm
            FSfOld = FSf

    return FSBishop, FSSpencer, FSJanbu
```

The other functions used within this FoS calculator are the *FuncStrengthLookup* and *funcGetGeomProp*. The first one is used to determine whether a midpoint of a slice is within the drained berm or in the potentially liquefiable tailings deposit. In this way, the proper strength can be assigned by returning a 0 for a liquefiable deposit and a 1 for a drained deposit:

```
def FuncStrengthlookup(x,y,topPoints,DL,LL):


    slopeData = FuncSlopeData(topPoints)
    for i in range(len(topPoints)-1):
        if y <= topPoints[i,1] and y >= topPoints[i+1,1]:
            section = i
            break

    xSlope = (y-slopeData[section,1])/slopeData[section,0]

    if xSlope - x < DL.damWidth:
        return 1
    else:
        return 0
```

The other function retrieves some of the geometric properties needed for the calculation for a non-circular failure surface. Among these are the offset from the axis point for the line perpendicular through the midpoint of the slice base. As well as the associated radius with that. More on these properties (variable names are maintained) can be found in Fredlund and Krahn (1977).

```
def funcGetGeomProp(axisPoint,x1, x2, y1, y2):
    if y1 == y2:
        f = axisPoint[0]-((x2+x1)/2)
        coordX = 0
        coordY = 0
    else:
        af = (x2-x1)/(y1-y2)
        bf = -1
        cf = ((y1+y2)/2)-((x2**2-x1**2)/(2*(y1-y2)))
        d = (af*axisPoint[0]+bf*axisPoint[1]+cf)/np.sqrt(af**2+bf**2)
        m = (y2-y1)/(x2-x1)
        f = d*np.sign(-m)

        coordX = (bf*(bf*axisPoint[0]-af*axisPoint[1])-af*cf)/(af**2+bf**2)
        coordY = (af*(-1*bf*axisPoint[0]+af*axisPoint[1])-bf*cf)/(af**2+bf**2)


    g = (y1-y2)*axisPoint[0]+(x2-x1)*axisPoint[1]+((x1*y2-x2*y1))
    h = np.sqrt((x2-x1)**2+(y2-y1)**2)
    R = abs(g/h)
    return f, R,coordX,coordY
```

D.5. Gumbel analysis of precipitation data

The Gumbel analysis of the data consists of a separate file. It can be integrated into the main code, but then the manual computations regarding the failure intensity will have to be eliminated and a full search algorithm for the intensity at which the FoS is 1.1 will have to be applied. The code consists of two functions and a main part. The main part holds the following:

```
PrecipData = np.array([])

for filename in os.listdir(r”D:\Users\zonj3\Documents\MSc. Thesis\2 Scripts\Phase 3\
1.0\Data Rainfall”):
    if filename.endswith('.csv'):
        file = os.path.join(r”D:\Users\zonj3\Documents\MSc. Thesis\2 Scripts\Phase 3\
        1.0\Data Rainfall”,filename)

        PrecipImport = pd.read_csv(file,index_col = 0,parse_dates = True, header = 0,
        usecols=[4], dtype = {'Precipitation':np.float64})
        PrecipImportNp = PrecipImport.to_numpy()
        PrecipData = np.append(PrecipData,PrecipImportNp)

PrecipDataNp = np.hstack(PrecipData)

PrecipDataNpf = np.zeros(len(PrecipDataNp))
for i in range(len(PrecipDataNp)):
    PrecipDataNpf[i] = float(PrecipDataNp[i])/1000.

GEVparams = []
DayLength = np.array([1,2,3,4,5])
ProbDays =  np.zeros(len(DayLength))
MaxDayInt = 365

Returns, stattest = MaxAnnua(4,MaxDayInt,PrecipDataNpf)
```

It starts by retrieving the data. In this case, the data is stored in a single folder that contains all csv files. These files are read using the *pandas* module of Python, but the data is transformed into a Python array. This is done as to apply the *scipy* package for extreme value analysis later on. What is obtained is a very large array of all recorded intensities (in m/day). As the data was assumed to be homogeneous, all of the data is assembled into a single array. This array is then used as input for the *MaxAnnua* function:

```
def MaxAnnua(N, MaxDayInt, PrecipDataNpf):
    '''Determining the maximum in MaxDayInt which is the interval for which
    the maximum is computed '''

    years = len(PrecipDataNpf)//(MaxDayInt)
    MaxAnnua = np.zeros(years+1)

    for i in range(years+1):
        if i < years:
            IndexStart = i*MaxDayInt
            IndexEnd = i*MaxDayInt+MaxDayInt
            MaxAnnua[i] = max(np.convolve(PrecipDataNpf[IndexStart:IndexEnd],
            np.ones((N,))/N,mode= 'valid'))

        elif i == years:
            IndexStart = i*MaxDayInt
            MaxAnnua[i] = max(np.convolve(PrecipDataNpf[IndexStart:],np.ones((N,))/N
            ,mode = 'valid'))

    plt.figure(figsize = (8,3))
    plt.plot(MaxAnnua)
    plt.grid()
    plt.title('Maximum intensity per annua for a single station')
    plt.xlabel('Station year')
    plt.ylabel('Intensity [m/day]')

    statTest = adfuller(MaxAnnua, regression = 'c')
    print(statTest[0],statTest[1])

    #First applying the method of moments
    muhat   = np.mean(MaxAnnua)
    sigmahat = np.std(MaxAnnua, ddof = 1)

    alphahat = np.sqrt(6)*sigmahat/np.pi
    uhat = muhat-np.euler_gamma*alphahat

    #Full Gumbel based on the method of moments
    locGumbel = uhat
    scaleGumbel = alphahat
    for i in range(10):
        if i == 0:
            locGumbelMLE, scaleGumbelMLE = gumbel_r.fit(MaxAnnua, loc = locGumbel,
            scale = scaleGumbel)
        else:
            locGumbelMLE, scaleGumbelMLE = gumbel_r.fit(MaxAnnua, loc = locGumbelMLE,
            scale = scaleGumbelMLE)

    #Full GEV on MLE
    for i in range(10):
        if i == 0:
            shape, loc,scale = gev.fit(MaxAnnua)
        else:
            shape, loc, scale = gev.fit(MaxAnnua,loc=loc,scale = scale)

    GEVparam = [shape,loc,scale]

    #Performing Kolmogorov-Smirnov test
    testResultGEV = kstest(MaxAnnua,'genextreme', args = (shape,loc,scale))
    testResultGumbel = kstest(MaxAnnua, 'gumbel_r', args = (locGumbel, scaleGumbel))
    testResultGumbelMLE = kstest(MaxAnnua,'gumbel_r', args = (locGumbelMLE, scaleGumbelMLE))
    print(GEVparam,testResultGEV)
    print(locGumbel,scaleGumbel,testResultGumbel)
    print(locGumbelMLE,scaleGumbelMLE,testResultGumbelMLE)

    plt.figure()
    plt.hist(MaxAnnua,cumulative = True, density = True, histtype = 'step', align = 'right')
    x = np.linspace(min(MaxAnnua), max(MaxAnnua), 50)
    y1 = gev.cdf(x,shape,loc,scale)
    y2 = gumbel_r.cdf(x,locGumbel,scaleGumbel)
    y3 = gumbel_r.cdf(x,locGumbelMLE,scaleGumbelMLE)

    plt.plot(x,y1, label = 'GEV MLE')
    plt.plot(x,y2, label = 'Gumbel MM')
    plt.plot(x,y3, label = 'Gumbel MLE')
    plt.legend()
    plt.title('Cumulative normalised histogram and fits for '+str(N)+' days')
    plt.xlabel('Intensity [m/day]')


    return
```

The *MaxAnnua* function starts with retrieving the annual maxima of the dataset. As all datasheets have values from 1979 to 2014, this can rather easily be done and no timestamp information is needed. The difficulty lies with the fact that e.g. for 3 days of consecutive rainfall, various steps are needed in order to perform all combinations. The *np.convolve* function from the *numpy* module is applied for this. The annual maxima (1620 in total) are returned in an array. Then the method of moments and MLE for the Gumbel and the MLE for the GEV distribution can be calculated. After which a Kolmogorov-Smirnov test is applied. The values are plotted and a histogram of the CDF is made. The function does not return any value as the values were printed and stored manually for later analysis.

The other function will return the probability of rain for a certain duration:

```
def ProbDay (PrecipDataNpf, N, PrecipTreshold = 2.5e-3):
    #Determing the probability of occurence for rainy days for a given treshold of rain
    counterPrecip = 0
    stateSequence = np.zeros(len(PrecipDataNpf))
    for i in range(len(PrecipDataNpf)):
        if PrecipDataNpf[i] > PrecipTreshold:
            stateSequence[i] = 1

    runningAvg = np.convolve(stateSequence,np.ones((N,))/N,mode = 'valid')

    for i in range(len(runningAvg)):
        if runningAvg[i] == 1:
            counterPrecip +=1
    Prob = counterPrecip/len(PrecipDataNpf)

    return Prob
```

At first, the precipitation data is transformed into states of either 1 or 0. The 1 indicates a rainy day, for which the threshold is set at 2.5 mm/day of rainfall. Then the same *np.convolve* function is performed, which returns the average of N values. If this average value is equal to 1, all N values are equal to 1 and one is added to the *counterPrecip*. Then, in the end this value is divided by the total amount of days to get to a probability of this duration. The final step is a simplification but because of the extremely large dataset, the end-effects are very small.