Implications of dredge mine design on mine optimization and discussing possible approaches

M.J.Bijmolt - 4088263

6/24/2014

TU Delft Delft University of Technology

IHC MERWEDE

PREFACE

Mining is almost as old as mankind itself and for many years, most of these mining activities took place on land, at the surface or deep under the ground. It hasn't been until the 90's that offshore mining is considered a real option. The technology for offshore mining probably goes back to ancient times, when ditches and rivers were kept open with spoon and bag dredges.

Nowadays, dredge mining is not only used for offshore mines but also for land operations with a relatively high water level. One of the more famous land based dredge mines is Richards Bay, which has been extracting heavy minerals from coastal dune sands in South Africa for more than three decades.

The increasing use of dredging equipment for the mining industry has not yet resulted in the research of more advanced optimization techniques for these operations. Standard optimization techniques developed for surface mines may not be applicable, given the unique design of a dredge mine. Therefore, a collaboration between the TU Delft and Royal IHC, a large equipment supplier and consultant for the dredging industry, was set-up to investigate possible approaches for optimizing dredge mine operations. It is believed that large steps can be taken to improve the business model for dredge mining if proper optimization techniques can be applied on the design of dredge mines.

ABSTRACT

The development of dredging as a major player for surface mine applications has led Royal IHC, a large equipment supplier and consultant for dredging and mining operations, and the TU Delft to work on more advanced optimizations techniques for the design of dredge mines. Three implications of the design of a dredge mine were found to be crucial for optimizations, namely: 1) depth control, 2) mining direction and 3) creation of multiple ponds.

The conventional approach for open pit mines, in which a series of nested pits are created to determine an optimal mining sequence, was tested using the core module of Whittle and showed not to be readily applicable on dredge mines, because 1) multiple ponds may be created, 2) the depth to be mined for a certain area changes in time and 3) the nested pits expand randomly towards high graded zones.

A new method for optimizing dredge mines was introduced as a second approach, which determines an ultimate depth per stacked block model, based on the cumulative values and finds an optimal route for a pond through these stacked blocks by using an adapted version of the Nearest Neighbour algorithm. Four limitations for this approach are recognized: 1) the depth difference between stacked blocks could become impractical, 2) full utilization of the field is not possible because it may reach a premature dead-end or it may enclose a group of non-mined blocks, 3) the blocks have to meet the same length and width requirements of a pond; therefore not incorporating the accuracy of the data and 4) it lacks the function ability to mine the area in layers.

Project Alpha indicated that the new approach finds an optimal mine design; however, the long lifetime of the mine (>60 years) results in a low recovery of 65%. Decreasing the lifetime of the mine would result in a higher recovery. The conventional approach showed to be impractical for the design of a dredge mine, while it created multiple thin deposits. The NPV of the worst case scenario of Whittle turns out slightly lower than the NPV of the optimal route determined by the second approach.

CONTENT

1 INTRODUCTION

Royal IHC is a major producer of dredging and mining vessels and equipment and houses its own mining department for dredge mining equipment and consulting services. To expand its scope of consulting services, a research project was initiated to investigate the possibility of using optimization techniques for dredge mine operations. This report is the first step of this project and serves as an introduction to dredge mine optimizations.

The goals of this report are as follows:

1. Determine the important implications of the design of a dredge mine on the optimization task
2. Discuss possible approaches for optimizing dredge mines

The two approaches chosen are 1) a conventional approach for optimizing surface mines and 2) a new approach based on the Traveling Salesman Problem.

To reach the goals mentioned above, the following objectives are recognized:

I. Evaluate the design criteria for a dredge mine
II. Determine the implications for the optimization based on these design criteria
III. Evaluate the usability of the conventional approach for optimization
IV. Develop a new optimization approach specifically designed for dredge mines
V. Discuss the usability of both approaches with a case study

Overviews of the objectives are also depicted in Figure 1.

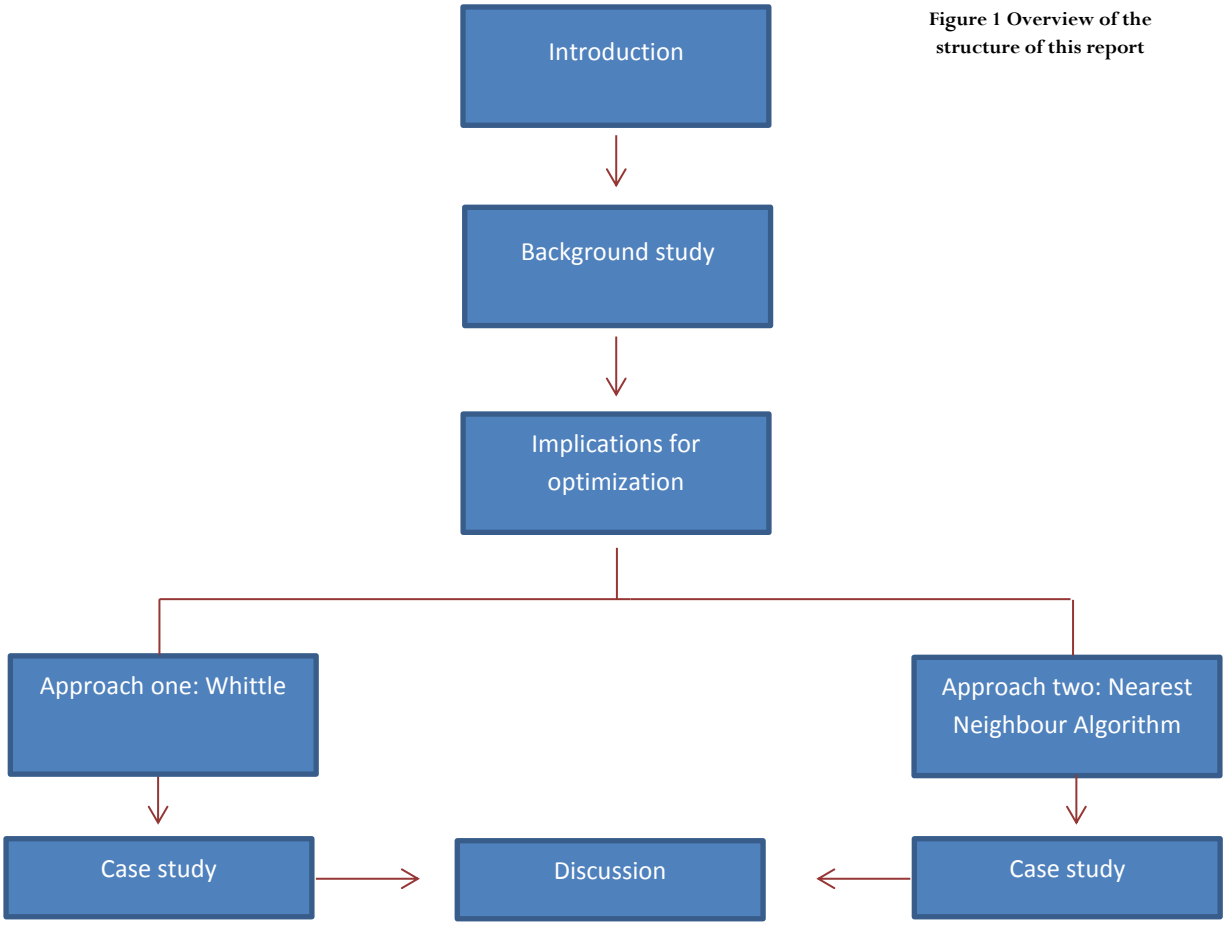


Figure 1 Overview of the structure of this report

2 BACKGROUND

DREDGE MINE DESIGN

Dredge mining is the maritime transportation of natural materials from one part of a (artificial) water environment to a processing plant and back to another water environment by specialized dredging vessels (EuDA 2014). At first, these cases seem limited by submerged deposits in offshore area's or in rivers, but nowadays, dredge mining is also used for land based operations and has become a realistic option besides the more conventional open pit method. The only real limitation is the presence of water.

A dredge mine consists of a natural or artificial pond (1), in which cutter-suction dredgers (2) accelerate the breaching of the bench (3) and, by means of hydraulic transportation, transport the material to a processing plant (4). In this processing plant, the oversize material, usually everything more than 2 mm, is separated by means of sieving from the rest of the material (Jones, unknown). The filtered ore is then processed to separate the minerals from the remaining waste, which is a combination of very fine material (<63µm), called slime and coarser material. The slimes are then mixed with the filtered oversize and reinserted into the pond (5). This last part can be challenging, while slimes settle very slowly, and recirculation may occur (Herchenhorn 2005). This can be mitigated by adding flocculants or separate filters; however, it is the experience of IHC that this is very costly. If the slime content is too high, backfill is not possible and a separate settling pond has to be designed. Therefore, slimes are preferably not mined. The entire process is drawn schematically in Figure 2.

Cutter suction dredgers come in different types and sizes, but in general all dredgers use centrifugal pumps as their main means of lifting and transporting the dredged material (Bray 2009). The material is disaggregated by a cutting device, of which there are two types: a cutter head and a cutter wheel. Cutter heads rotate around the axis of the suction pipe, whereas cutting wheels rotate perpendicular to the axis of the suction pipe. (Bray 2009). Cutting wheels are in general less selective than cutter heads (Bray 2009). It is the experience of IHC that he average selectivity for a cutter suction dredger is two meters. Figure 3 shows the different heads. Cutter suction dredgers are kept in place by two spuds anchored into the ground. When this is not possible due to poor ground conditions, steel cables to the embankments are used to fix the location of the vessel. The cutter suction dredger can orientate itself by rotating around the spud or by slackening the cables. The dredger moves forward by pushing itself away from one spud pole, while the second pole is retrieved from the ground. When it has reached its maximum distance, the second spud pole is anchored and the first pole is retrieved, repeating the entire process. When the vessel needs to be moved to a new position, it can be moved by a tugboat or by its own engine if the vessel is self-propelled. The reach of a cutter suction dredger in terms of depth is determined by its cutting arm. At IHC, the maximum reach is about 30 meters for its largest cutter suction dredger.

Given the lay-out of the mine and the description of the cutter, it is clear that a large number of parameters determine the eventual mine design. Therefore, it is important to state these design criteria before optimizing its design:

- Slope angle
  - A high slope angle support easy breaching, however, it makes the pond instable and a bench collapse on the cutter's arm should be prevented at all times. A typical value used by IHC for the slope angle is 30 degrees.
- Pond size
  - The pond size is greatly dependent on the size of the dredger and processing plant, but also depends on the place and environment. A typical pond size used by IHC for a regular operation is 350 by 350 meters. A smaller pond will decrease the impact on the environment if the pond is immediately backfilled. Another factor influencing the pond size is the reach of a dredger. It is possible to mine the pond in layers, where each layer is mined in turn to reach greater depths.
- Mining direction
  - A dredger works most efficiently when it can mine into one direction as long as possible, given that the dredger moves by using its spud poles. The length of the pond is seen as the minimal length by IHC for an efficient operation.
- Backfill
  - The advantage of a dredge mine is that the waste can be directly used to fill the gap left behind, minimizing the environmental impact. However, when slimes are abundant, it may not be efficient to backfill in the same pond, while the slimes may not settle, causing the dredger to re-mine the material, and leading to very unproductive scenarios. In these cases, a separate pond is required to store the waste or additional filters or flocculants are required to enhance the settling velocity.
- Processing
  - The processing steps can be rather complex, using multiple separators like hydro cyclones and sieves. In general, it is the experience of IHC that the processing unit works most efficient if the feed is relatively constant in terms of size distribution and grade.
- Material properties
  - Material properties like size distribution and hardness play an important role in the design of the processing plant and in the design of the dredger, but also in the choice of a slope angle. The weaker the cohesion between the materials, the higher the production can be, while the material will be breached efficiently. The material properties are not considered in this report, while these may differ for each location

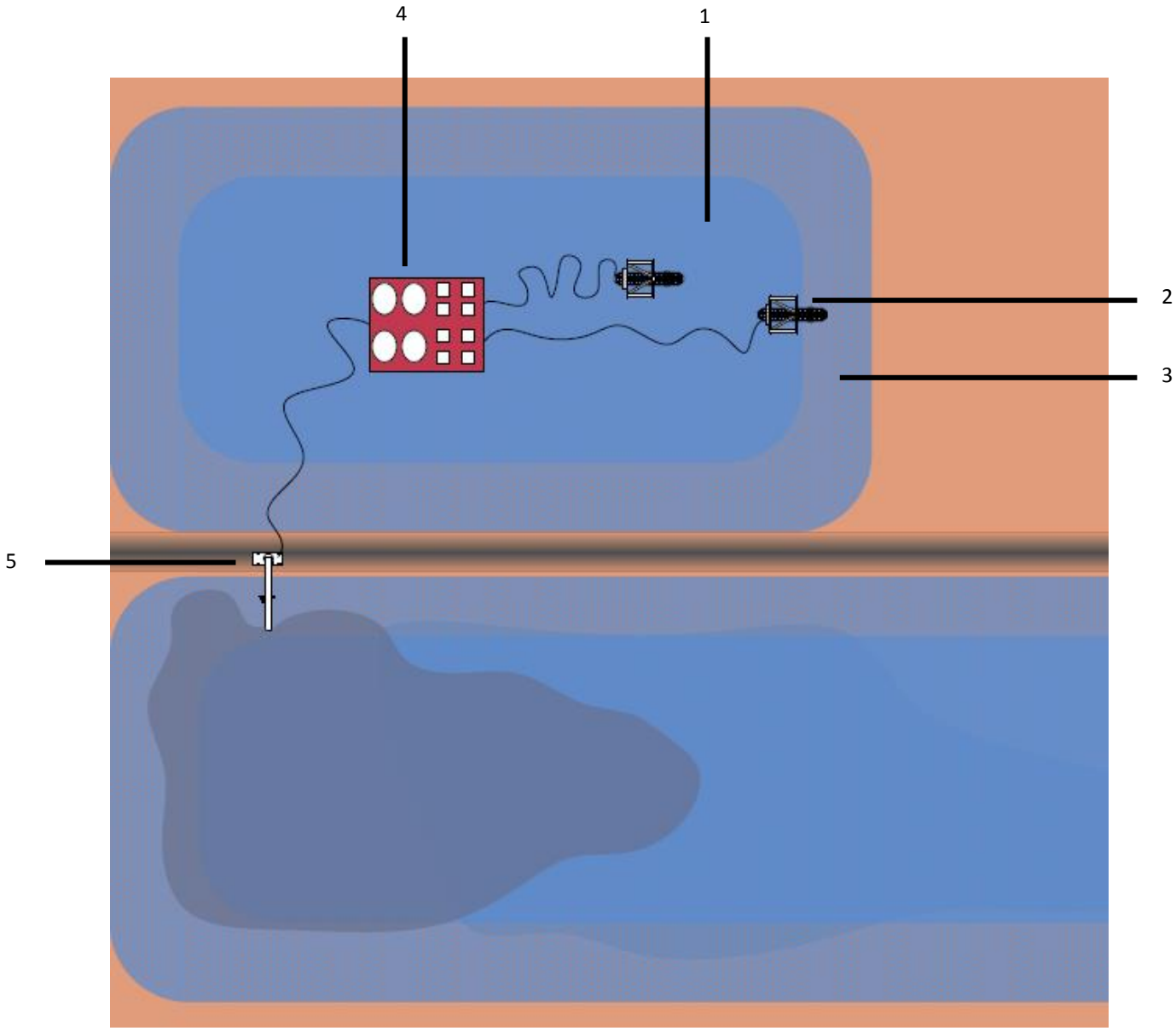


Figure 2 Schematic drawing of a land based dredge mine. The numbers correspond to the numbers in the text on page4.

Figure 3 Left: Cutter wheel. Right: cutter head

The mining industry is known as a capital intensive industry, meaning that the investment costs are relatively high compared to the labour costs (Connolly 2011). These capital costs are also called capital expenditures or Capex and are fixed, one-time expenses for the purchase of land, buildings and equipment. The operating costs include all costs that would halt if one would stop the operation (Hustrulid 2006). These costs are also called operating expenditures or Opex.

The cash flow of a mine is a good indication for the financial status of the mine. The cash flow is the real money going in and out of the company, therefore, not taking into account virtual costs like depreciation. The cash flow of a mine is calculated as follows (Hustrulid 2006):

$$F(t) = R(t) - O(t) - C(t) - I_t(t) * T \quad (1)$$

Where:

- F(t)= cash flow in year t
- R(t)= revenue in year t
- O(t)= Operating costs in year t
- C(t)= Capital costs in year t
- I_t (t)= Taxable income in year t
- T= tax rate

To take into account that revenues are made in the future, the Discounted Cash Flow (DCF) method is applied (Hall and Nicholls, 2007). The DCF method estimates the project value as the present value of expected future returns and is typically implemented under an assumption that investment policy is independent of prices (Hall and Nicholls, 2007). The DCF assumes a constant and known discount rate (Alessandri, 2004). If the calculated DCF exceeds the investment costs, revenue can be generated. The following formula may be used for calculating the DCF:

$$DCF = \frac{FV}{(1+i)^n} \quad (2)$$

Where:
DCF= discounted cash flow
FV=future cash flow
i=discount rate
n=time in years before the future cash flow occurs

To express the value of a project, the Net Present Value (NPV) is used (Alessandri, 2004). The net present value is calculated by summing the DCF's per year. One of the shortcomings of the DCF method is that it assumes that the project is always finished, while in reality, the management may decide to stop the project at any time, given a change in the business environment (Hull, 2012). The real option method takes into account this flexibility, however, the DCF method is still widely used in the mining industry, therefore, in this report, the DCF method will be used.

3 OPTIMIZATIONS

INTRODUCTION

Many optimization techniques exists for different mine phases. In this chapter, the different optimization types are set forth; of which one is selected for this research. This optimization type is then discussed thoroughly for the general surface mine applications and followed by a discussion on the implications of the design of a dredge mine on the optimization method.

The general assumption made in this report is that the ore body can be represented as a 3-D grid, made of single and constant values for each of the variables. Whittle (2004) and Godoy (2004) proposes to take into account the variability of the values, hereby incorporating the uncertainty of the interpolation method. However, this is outside the scope of this report.

OPTIMIZATION TYPES

Mine optimization can be done in many ways and at different stadia, but the general focus is towards the planning and operation phase, while these phases have a significant influence on the economical and operational viability. The phases of a mine are shown in Figure 5 and the different ways of optimization are listed below (Hajdasinski 1988).

Planning phase

- *Maximizing NPV by pit optimization*
- *Maximizing the recovery*
- *Minimizing environmental impact by pit optimization*

Operation phase

- *Extend mine life by pit optimization*
- *Improve DCF by pit optimization*
- *Extend the recovery*
- *Decrease Opex*

The optimizations in the operational phase may be conducted when new information is available about the ore body or when new techniques are available. When conducting such optimizations, one has to take into account the area that is already mined, creating more boundary conditions than optimizations in the planning phase. Therefore, this report will only consider optimizations in the planning phase.

Of the optimization possibilities in the planning phase, only the first optimization type, maximizing the NPV by pit optimizations, is considered in this report. The reason for choosing this optimization type is that it is easily quantifiable, which makes it very suitable for computer models. A description of this optimization type is given on the next page.

Maximizing the NPV in the planning phase is done by modelling an ultimate pit and by optimizing the operational schedule. The ultimate pit is defined as the maximum outline of the mineable area that is economically viable to mine (Hustrulid 2006). The most important input for the ultimate pit is an ore body, modelled in a mine modelling software using one of many estimation methods. The other parameters are market or user driven, but also include a geotechnical limitation, namely the slope angle at which the pit is stable. The market driven parameter is the price of a mineral and the user driven parameters are the costs of mining and the costs of processing the minerals.

Based on this ultimate pit, an operational schedule can be developed to mine the ultimate pit in parts, called push backs (Hustrulid 2006). The sequence is chosen to maximize the NPV, which means that the high graded zones should be mined first to generate the highest cash flow in the first years (Hustrulid). The parameters that define this optimal schedule are the same parameters that also define the ultimate pit, but also include two more user driven parameters, namely the capital expenditures and the production rate and include the mine design limitations. An overview of these processes is depicted in Figure 4.

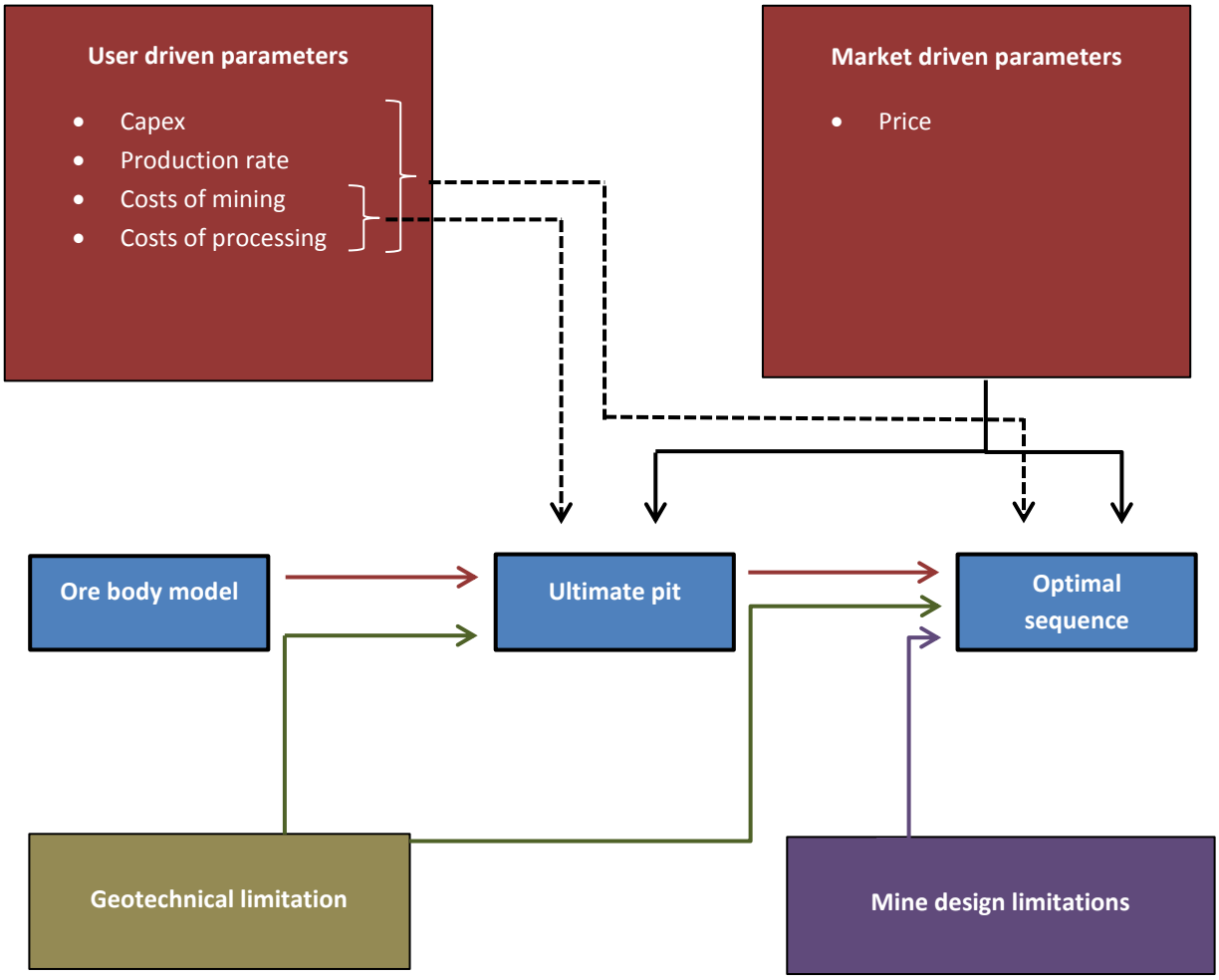


Figure 4 optimization processes and challenges for dredge mining

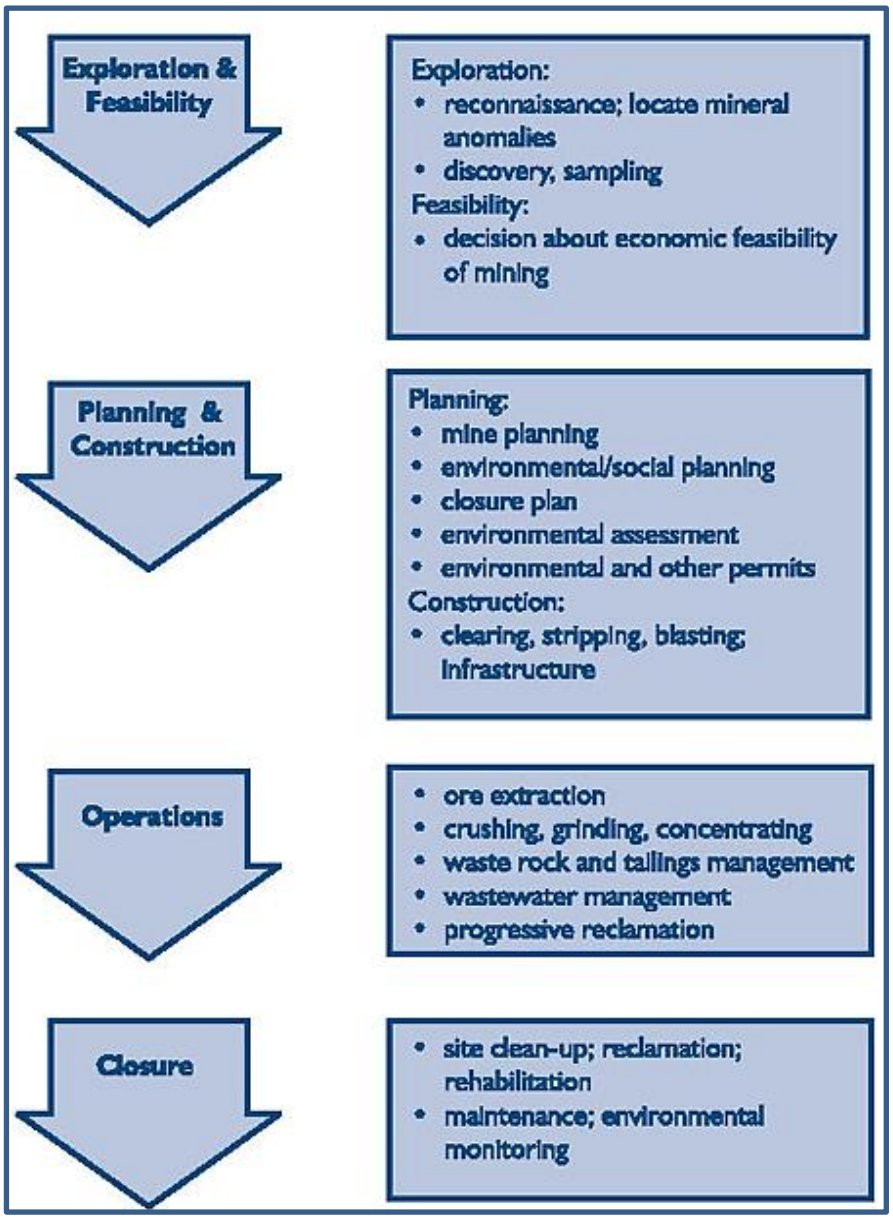


Figure 5 Activities of the mine life cycle (CEPA 1999).

When considering an ultimate pit for surface mines, the slope angle controls for a large factor the outline of the ultimate pit. The reason is that for most surface mines, the deposit lays deep under the ground or is vertically orientated, which means that the slope angle will determine the amount of waste that is required to be removed for reaching the ore and therefore, to what extend the mine can be mined economically. The ratio between waste and ore is called the stripping ratio and is illustrated in Figure 6 (Hustrulid 2006). When the ore body is more horizontally orientated, like for example most sedimentary deposits (Grotzinger 2007), the ultimate pit is less influenced by the slope angle. This can be easily seen in Figure 6: if the ore deposit would be more horizontally stretched, the strip ratio would decrease significantly.

The result of this design can be rather impractical, while the two restrictions, the slope angle and the economic viability, do not take into account any of the design limitations. This is normally resolved in the pit sequence step.

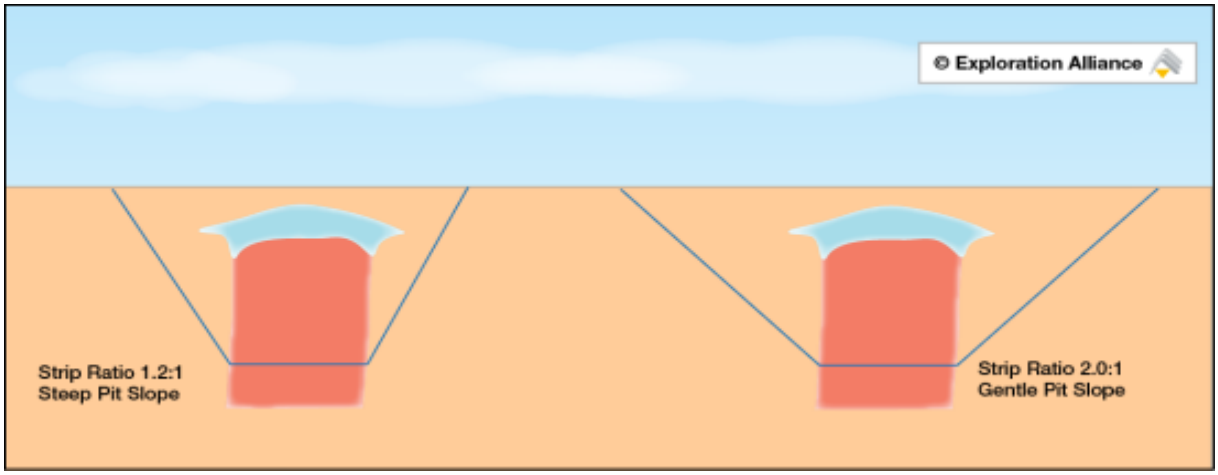


Figure 6 Stripping ratio for an ore body with different slope angles (Cook 2011)

There are a number of optimization algorithms on the market to define the ultimate pit size. All the algorithms have in common that they use a two or three dimensional block model as a representation of the ore body. The block model can be seen as a grid, for which values for different variables are estimated at each grid point. The most commonly used algorithms for surface mine applications are stated below with a short introduction:

Floating cone

The floating cone program works by repeatedly defining cones for each block and evaluating whether the blocks in the particular cone have a positive total value. The slope of the cone obeys the general slope restrictions. If the cumulative values of the blocks in the cone are positive, the cone is mined and the search continues (Hustrulid 2006). The disadvantage of this program is that it has difficulties when two ore zones are separated with an overlapping waste zone. The algorithm could then reject both ore zones, while this would still be economically viable (Whittle 2004). Therefore, this algorithm is only applied in combination with a manual optimization.

Two dimensional Lerchs-Grossmann

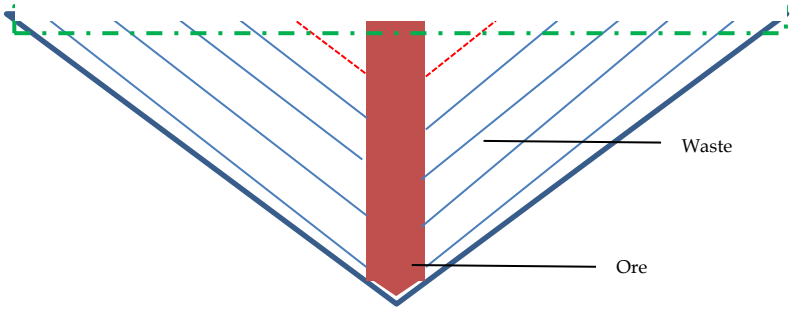
This algorithm uses a 2D block model and arcs for describing the slope restrictions to optimize the ultimate pit. The algorithm starts by cumulative summing columns from the top. It will then find the neighbouring block with the highest grade or value for each block in the pit, creating links between them. The linked block then obtains the value of the combined blocks. By doing so, it creates an ultimate route with the highest total value. Completely explaining the Lerchs-Grossmann algorithm would take multiple pages and is not done here. The author refers to the following papers and books for the interested reader: Lerchs (1965), Sainsbury (1970) and Hustrulid (2006). The two dimensional algorithm can be extended to a 2.5-D and a 3-D algorithm, of which the 3-D algorithm is mostly used and applied by at least four different program sets (Whittle 2004). It uses the restictrion that for mining a particular block, 5 or 9 of the above laying blocks are required to be mined (Hustrulid 2006).

When an ultimate pit is obtained, a pit sequence is made, which divides the ultimate pit into smaller, mineable pits, called push backs (Hustrulid 2006). In general, there are two types of pit sequences: a worst case and a best case (Whittle 2004). In the worst case sequence, one begins with removing the entire top layer and continues downwards. It is called a worst case scenario because the waste to ore ratio is initially very high and decreases in time, thus making very high costs in the beginning, which has a negative impact on the NPV. In the case of a best case scenario, the ore is mined with the smallest possible amount of waste, creating shells, which results in a higher NPV. Both cases are explained in Figure 7. To see whether pit sequencing matters to the mine economics, one can compare the NPV for the worst and best case scenario. When these figures do not differ more than a few percent, pit sequencing is considered to be of less importance (Whittle 1999).

Various techniques exist to develop these pit sequences, but the most common one is to produce a nest of ultimate pits corresponding to various mineral prices (Hustrulid 2006). The original ultimate pit is determined by the most likely mineral price and the rest of the pits are determined by lower mineral prices. These pits will then migrate towards this original ultimate pit. The result of using this technique is that the nested pits do not obey the restriction of the mine design and have to be reshaped to create pushbacks that can be mined (Whittle 2006). For example, odd corners or peaks can be removed by various programs (Whittle 2006).

One of the important limitations of maximizing the NPV in this way is that the ultimate pit is designed without taking into account the fact that the blocks are not mined at the same time. This is then compensated by optimizing the schedule for that particular pit design, creating a best and worst case scenarios, which often produce a wide range of possible pits (Hanson 2001). The best case scenario is over-optimistic, while it is unlikely that the waste is mined in the same year as the associated ore and the worst case is a very pessimistic scenario that is rarely seen in practice (Hanson 2001). It is therefore important to see the method of the nested pits as a guide to help in the choice of selecting a pit design schedule.

Figure 7 Worst case mining, in which the green block is mined first and best case mining, in which the red dotted area, called a shell, is mined first



LIMITATIONS FOR DREDGING APPLICATIONS

The before mentioned techniques are specifically designed for open pit mines and are not readily applicable on dredge mines. Therefore, the information about dredge mine designs from chapter 1 is combined with the information about optimization for surface mines to analyse the implications of the design of a dredge mine on these optimization techniques. The section below discusses what challenges are faced when designing the ultimate pit and developing an optimal sequence for a dredge mine.

Ultimate pit

The ultimate pit describes the maximum outline for which the mine is economically mineable. The outline is defined by the slope restrictions. The nature of these conditions do not differ when designing an ultimate pit for a dredge mine. Therefore, determining the ultimate pit will not affect the design of a dredge mine differently than it would have affected the design of a surface mine.

Pit sequencing

From the ultimate pit, a number of nested pits are created, from which practical pushbacks are created. The challenges for the processes are shown in Figure 9 and described in the next section.

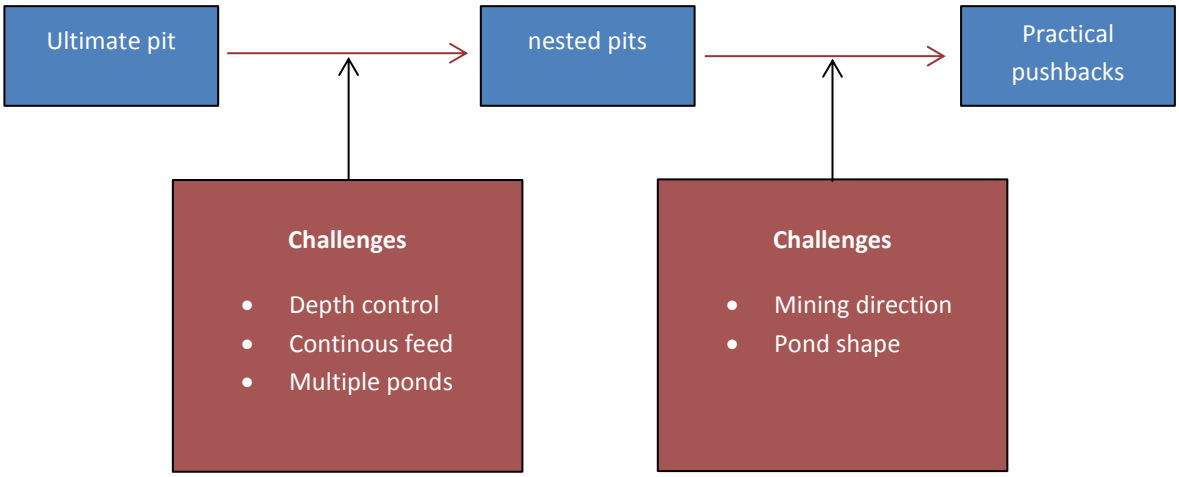


Figure 9 Challenges of a dredge mine optimization

Depth control

A dredger has a maximum depth it can reach and if one wants to mine deeper with dredging equipment, the top layer has to be mined and the groundwater level lowered before the lower layer can be mined. It is not possible to have one pit with varying depths that exceeds the maximum depth the dredger can reach. However, the nested pits do not take into account this maximum depth. A possible solution would be to remove all blocks below this maximum depth; however, this would limit the resources and is not a very practical solution.

Another challenge regarding the maximum depth is that these nested pits may extend the size of an earlier pit, while this would be the optimal choice in terms of NPV. However, this is not a practical sequence strategy for dredge mines because of two reasons: 1) dredge mines are continuously backfilled and 2) dredging vessels but especially the processing plant are not easily transported back to an earlier spot.

Continuous feed

It is the experience of IHC that a processing plant of a dredge mine works most efficiently when there is a continuous feed in terms of grade and size distribution and while most processing plants are directly connected with the dredgers, the combined feed from the dredgers need to be as continuous as possible. The nested pits do not carry information about the continuity, while it focusses on the values. It will not be crucial for a dredge mining operation to have a continuous feed, while other options are available to artificially create a continuous feed. For example, a buffer can be used to pre-mix the feed.

Multiple ponds

Another practical limitation that is expected based on the workings of the nested pit is the creation of multiple pits. This is especially true for deposits with disconnected high graded zones, while these would be mined simultaneously by the algorithm, albeit at different locations. This is not a practical design for a dredge mine, while the processing plant is also located in the pit. This would require multiple processing plants or long pipelines to other pits, which are all unattractive scenarios. Therefore, the nested pits should be required to limit to one pit that expands.

Mining direction

The nested pits grow randomly towards new ore zones, not taking into account any restriction on direction. Pit optimization software generally extends the pit radially, searching for the best ore/waste ratio's and creating the well-known open-pit mine shape (Whittle 2006). However, this is not a practical design for dredging activities, while these mines require a more rectangular path that can be followed, because it is in this way that the dredger works most efficiently. The challenge when designing practical pushbacks is that the mine should expand to only one direction for a certain time span.

Pond shape

If the requirements for the mining direction are met, the pond shape will only require for the practical pushbacks to be a practical shape, meaning that, for example, impossible corners are removed and the pit is made sufficiently large to fit the processing plant. It is the expectation that these requirements can be easily met, while it is already applied for surface mines.

Implications

Given the descriptions above, three implications are defined as critical design criteria that have to be met for a practical dredge mine optimization:

1. Depth control
2. Multiple ponds
3. Mining direction

Of these three criteria, the first two are the most important. The third criterion is in place to ensure certain efficiency; however, a dredge mine could be designed without this criterion. The design criteria will be used to evaluate the two approaches discussed in the next chapters.

4 APPROACH ONE: WHITTLE

Approach one is to investigate the conventional optimization approach, by using readily available optimization software for optimizing dredge mines. The software that is chosen is Whittle, a commercial mine optimization software, which is specifically developed for surface mines. The reason for choosing this particular software package is twofold: 1) It integrates perfectly with the available ore modelling software Surpac, while both software packages have the same developer and 2) it is a well-known package and has an established reputation.

The chapter begins with an introduction of the software used and continues with a discussion on the usability for optimizing dredge mines. The chapter concludes with a discussion about the usability and its challenges, which will be verified in the case study.

Software

The software packages used for this approach are Surpac, for the modelling of the ore body and Whittle, for the optimizations. Both software packages are developed by Dassault Systemes, a large software developer for civil, aeronautical, medical and mining applications, with revenues over 2 billion euros (Dassault 2013). The two packages are briefly discussed below. The section continues with an assessment of the challenges for Whittle and what the proposed working method would be for optimizing dredge mines.

Surpac

Surpac is a comprehensive geology and mine planning software, supporting open pit and underground operations. For this approach, it is only used to create a block model. The workflow of creating a block model in Surpac is as follows:

1. Importing and visualizing the drillhole data
2. Develop an understanding of the geology
3. Determine the variability of the data and develop a geostatistical approach
4. Create a block model that encloses the entire ore body
5. Estimate the values for the block model

For the block model to be correctly imported in Whittle, certain design requirement needs to be met (Whittle 1999). These requirements are as follows:

- blocks above the surface should be assigned a specific rock code, generally, the code AIR is used for this purpose
- When using a predefined cut-off constrains, the blocks with a grade lower than the cut-off grade should be assigned a rock code specifying it as waste
- Percentages should be converted to fractions. Whittle has known stability issues with percentages

Whittle

The key object of Whittle is to evaluate the financial viability and optimal mine strategy for a deposit. It realizes these objects by modelling nested pits and creating scheduling strategies. The core of the program runs on the 3-D Lerchs-Grossman Algorithm (3LGA) and is used to model the nested- and final pit. The nested pits are created by setting the 3LGA to different revenue factors. These revenue factors are scenarios for a higher or lower product price, usually a factor between 0.3 and 2 of the base price (Whittle 1999). The revenue factors are selected by the user. The nested pits are then used to define practical pushbacks with a certain number of benches in between them. These steps can be done manually or automatically, depending on the available license. Given these pushbacks and a production capacity, Whittle calculates an optimal mining sequence. Together with this mining sequence, economic data and mining data is given to the user, such as the NPV and tonnage mined.

The workflow of Whittle is as follows (Whittle 2013):

1. Import block model with assigned grade and rock code
2. Set pit slope zones for the optimization
3. Define the economic parameters, such as mining and processing costs and selling price
4. Model nested pits
5. Define operational parameters, such as capital costs, discount rate and production limit
6. Determine the final pit and pushbacks and choose the number of benches

It should be noted that Whittle is made of modules, each with its own price. The core modules enable the user to model nested pits and conduct a sequence optimization. All other features are optional, for example the selection of practical pushbacks or number of benches. For this project, only the core module was available.

Preliminary usability evaluation

A preliminary usability evaluation is done based on the tutorial manual (Whittle 2013) and the extended reference manual (Whittle 1999) of Whittle.

Design criterion one is the depth control. Whittle's core module does not support a function to define a minimum depth. Therefore, to meet the pond restriction, it is only possible to select nested pits that are at their maximum depth. However, it is expected that this will result in impractical situations, where the nested pit will be at its maximum depth the moment it has reached its maximum outline, which would make sequencing impossible.

The creation of multiple ponds, criterion two, is a direct result of the use of revenue factors to define the nested pits. The algorithm will only look at the economic viability and is not aware of the fact it creates multiple pits. Whittle does support a function to define practical push backs, which enables the user to define a minimal pond area. However, this function is not able to stop the creation of multiple ponds

For design criterion three, it must be possible to set a mining direction. The core module of Whittle does not support a function to set a mining direction or to force the expansion into one direction.

5 *APPROACH TWO: NEAREST NEIGHBOUR ALGORITHM*

For the development of approach two, a complete new process is developed, based on the unique design criteria of a dredge pond. Furthermore, it uses an adapted version of the Nearest Neighbour algorithm, which will be explained later in this chapter. The chapter begins by restating the objective of the optimization and defining a way to simplify the optimization task. It continues with an overview of the techniques used by this approach and a description of the work flow. The chapter ends with a discussion about the possible limitations.

Simplification

The objective of the optimization in this report is to maximize the NPV. This can be accomplished by defining a mining strategy for the ore body, which should include a sequence to mine the ore body given a certain pond size and an optimal depth. One way of looking at these ponds is as square blocks that migrate through the landscape. When looking at the system in this way, it is possible to simplify the challenge of optimization by reducing it to a two dimensional challenge. The two dimensions represent the coordinates of the blocks that migrate through the landscape.

To incorporate the optimal outline of the mine, an optimal depth is selected for each grid point based on an evaluation of the depth-value relation for that grid point, explained later in this chapter. These grid points are called columns, because they are in essence stacked blocks. The columns are then set to represent a value and a tonnage to be mined based on the stacked blocks till the ultimate depth, creating a two dimensional grid with values and tonnages per location. From this grid, it is possible to determine an optimal route through the ore body, based on a production, which determines how long it takes to mine a column. From each column, a direct neighbour can be selected to continue mining. This is another simplification, while normal dredging operations have the possibility to choose all directions. Direct neighbours are defined as blocks that have their sides next to each other.

To summarize, the following simplifications are made:

1. A dredge pond can be seen as a square block with set dimension for their length and width
2. Mining only commences to one of the direct neighbouring blocks
3. Each column has a distinct optimal depth

Techniques

The techniques used to develop this approach are split into three parts, namely:

- data importing
- Determining the ultimate depth
- Evaluating the optimal sequence.

All parts are developed using Matlab®, a high-level language for numerical computation, visualization and programming. The full Matlab® code can be found in appendix B. In this section, only the ideas behind the code are described.

Data importation
The block model is built in Surpac. Information about Surpac and the creation of block models can be found in the previous chapter. The attributes of the bock can also be selected for the export. The blocks are then read in Matlab, sorted for x, y and z values and grouped per grid point.

Ultimate depth

The ultimate depth is determined per column by searching for the highest cumulative value, which is defined as the value of the minerals in the above lying blocks and current block, minus the costs of mining and processing the above lying blocks and current block. This is depth-value relation. In this way, a cumulative value is calculated for each block in a column. The ultimate depth is then selected by choosing the block with the highest cumulative value for each column. A matrix is made to store the coordinate, ultimate depth, cumulative value and cumulative tonnage for each column. The cumulative tonnage is defined as the volume of the block multiplied by the density and summed for the above lying blocks and current block.

Optimal sequence

The search of an optimal route through the mineable area can be described as a Travelling Salesman Problem (TSP), a well-known combinatorial optimization problem. The goal is to find the shortest tour that visits each city in a given list exactly once and then returns to the starting city. Some applications are mentioned by Gilbert (1992) and Matai (2010), but in this case, the cities represent the ponds that have to be mined and the distance between the cities are represented by the value of the blocks. The adaption of the problem for this case lists as follows:

- A constrain is set for the migration of the ponds, while it can only move to one of the direct neighbours
- A closed route is not possible, given the constrain on the migration and the fact that the first pond is already mined and cannot be visited again
- While blocks cannot be visited again, it is possible to reach a dead end, which happens when all direct neighbours are already mined. A lot of ore can be missed if this happens when most blocks are not mined yet. Therefore, the algorithm has to prevent, to some extent, the reach of a premature dead end.

To solve a TSP, exact and heuristic algorithms exist. Some of these algorithms are described by Hahsler (unknown) and Gilbert (1992). In this report, only the heuristic options are discussed. The reason for choosing a heuristic approach is because there is no known polynomial-time algorithm that is able to solve all instances of the problem (Rego 2010). The heuristic approaches can be generally split into tour construction procedures and tour improvement procedures (Hahsler, unkown. Gilbert, 1992).

For this approach, only a tour construction procedure is developed, based on the Nearest Neighbour Algorithm (IA). The Nearest Neighbour Algorithm's basic structure is as follows (Gilbert 1992):

- Select a starting point and construct a first link
- Consider in turn all options not yet in the tour and select the option that is closest to the current point.

In this case, all starting points are considered. For each starting point, the above sequence is followed. Furthermore, the necessary adaptions, mentioned in the beginning of this section, are implemented in the algorithm.

To further improve the results of the algorithm, it bases the decision of the next block on a potential route from each direct neighbour through the entire ore body. The potential route is based on the neighbouring block with the highest discounted value, to take into account the fact that it is better to follow a route with high initial values. The discount rate is applied for the year the column would have been mined for that potential route. The algorithm considers a maximum of six years of discount rate. If one were to use more than six years, the values of the columns would be very low. This would result in an algorithm that is not able to see the columns properly, because they are several orders smaller than the initial columns.

If the neighbouring blocks of the potential route are already mined, the algorithm looks at the second-highest value and so on. When a block is chosen in this way, the algorithm will look if that block is surrounded by mineable blocks. If all neighbouring blocks of this chosen block are already mined, it will reconsider its choice: if there is another block available, surrounded by mineable blocks, it will choose this block; if this is not the case, it has reached a dead end and will stop searching. Figure 10 depicts the work flow of the algorithm.

Appendix C shows an example of the optimization engine as described above.

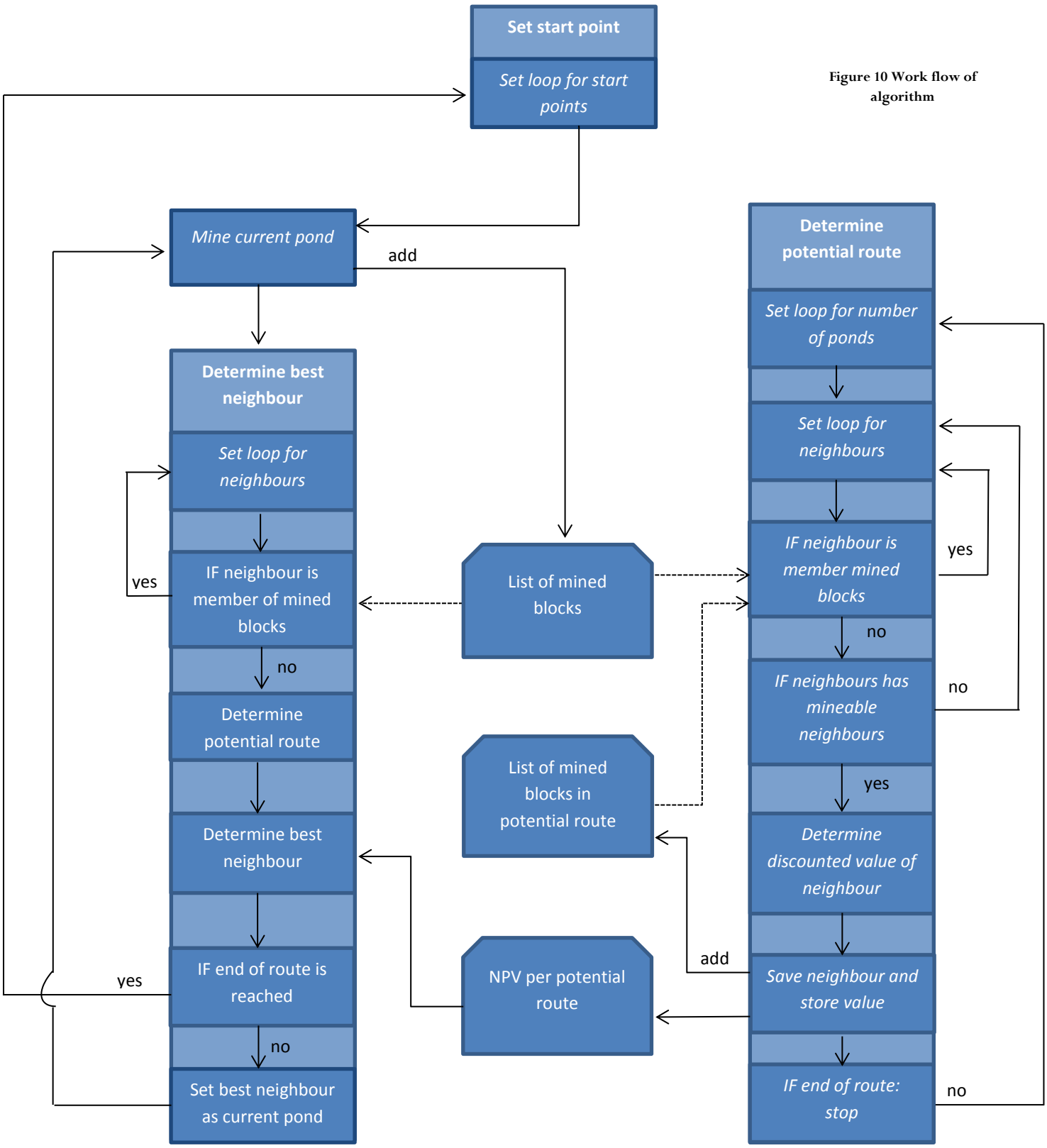


Figure 10 Work flow of algorithm

Functionality

In this section, a description is given on how to use the algorithm. It is split in two parts: an input part and an output part, which are discussed separately.

Input
To use the algorithm, a block model must be imported. The block model must obey some restrictions, listed below:

- The blocks must be of square size, with equal dimensions for all blocks. Sub-blocking is not allowed.
- The height of the blocks may equal the selectivity of the chosen dredging equipment, but can be chosen smaller
- All blocks that are above the surface must have a zero density
- The mineral content must be assigned as a fraction
- If a predefined cut-off is applied, the mineral content for these blocks may be set to zero if it skips the processing plant

The block model should then be exported to an excel file. If it is only possible to export it to a csv file, a conversion to an excel file must be applied. In the current version of the software (2.71), the density should be in the fifth column of the excel file and the mineral content should be in the sixth column.

Besides the block model, the user driven parameters and market driven parameters, as shown in Figure 4, can be configured. A complete list of the input parameters is given below.

- *Power of the search engine*
  - This parameter controls the amount of blocks that are considered when determining the potential route. If it is set to zero, it will consider all blocks.
- *Opex and Capex*
  - The operating costs are split into a processing part and a mining part. These costs will influence the ultimate depth of the ponds and will also affect the discounted cash flow and NPV. The Capex will only affect the cash flows and NPV's.
- *Price*
  - The price determines the amount of money received for a certain tonnage of minerals. The price will greatly influence the ultimate depth per pond. Furthermore, it will have an effect on the cash flows and NPV's.
- *Slope*
  - The slope angle determines the amount of material that is kept in the pond due to the embankments. The software compensates for two sides of embankments.
- *Production*
  - The production determines how quick a column is mined and therefore, greatly influences the cash flow and lifetime of the mine.
- *Group number*
  - This parameter determines the amount of groups that are made to properly display the sequence in mine modelling software

Output

The software delivers two sorts of outputs, a csv file containing an updated block model and graphs and tables, showing the information about the economics and sequences. The updated block model is a csv file, containing the centroids of the blocks, together with a sequence number and a group number. The columns are given group numbers, while most mine modelling software's have a limit on the number of grades for assigning colours. The number of groups can be configured at the input parameters.

The following graphs and tables are given by the software:

- A surface plot of the ultimate depth
- A movie, showing the sequence in which the ponds can be mined for the highest NPV
- A graph and table, containing the discounted cash flow per week
- A table, containing the NPV per starting point

Limitations

This section discusses limitations of the approach.

Limitation 1

The ultimate depth is determined per column, which may result in depth differences between neighbouring columns that cannot be maintained in practice. Differences of a few meters should not give major problems, given the large length and width of the column. It is likely that in these cases, a natural slope will arise. However, when the difference grows larger, it becomes more unstable and impractical.

Limitation 2

The implementation of determining a potential route for each neighbour should decrease the possibility of running into a dead end prematurely. However, it is unlikely that it will pass all the available blocks, because it is still possible for the algorithm to run into a dead end when an ultra-high graded zone exists or when it accidently encloses a group of blocks. In the case of an ultra-high graded zone, the values of these blocks will be high, which could compensate for the premature stop. Furthermore, the algorithm can enclose a group of blocks accidently, while it does not know that it will not be able to reach it again.

Limitation 3

It is not possible to mine the area in layers with the software. The reason is that this would be too complex to implement for a first introduction of the new approach.

Limitation 4

A block model should meet the requirements of the pond, which means a limitation on the size of the blocks. Blocks may be chosen smaller, however, the result will be that shape of the path could become very impractical for a dredge mine. If the blocks are smaller than the size of the processing plant, it will not be possible to design a dredge mine.

6 CASE STUDY

For both approaches, a case study is done, using the same heavy mineral case. The case study is split into three parts: part one gives an overview of the project and the input parameters for the optimizations, part two evaluates approach one for the case study and part three evaluates approach two for the case study.

PART 1 – PROJECT ALPHA

This part describes the ore deposit. It begins with a description of the depositional environment and a selection of a test area. It continues with determining a block model for the ore body with a brief geostatistical analysis and ends with an evaluation of the mining parameters and some mining context.

The project that is used for the case study in this report is a real heavy mineral placer deposit, called project 'Alpha'. It consists of sheet-like deposits that stretch over 25 kilometers, deposited by ocean waves on a beach. The heavy minerals originate from weathered rocks, whose grains of sediment are sorted by weight when currents of water flow over them. Therefore, the heavy minerals accumulated on sandbars, where the current is strong enough to keep the lighter particles in suspension, but weak enough to deposit the heavy minerals (Grotzinger 2007). The vertical variability in such deposits is relatively high compared to the horizontal variability, because the sediments are deposited as horizontal layers. The mineral composition of the deposit is classified and not discussed in this report.

Given the large size of the deposit, a smaller test area is selected for computational reasons. The deposit is modelled using Surpac and the test area is selected by modelling the entire ore body as different sheet. Figure 11 shows these sheets for a cut-off of 1% heavy mineral content. It also shows the test case area in red, which is approximately 6 by 5 kilometers.

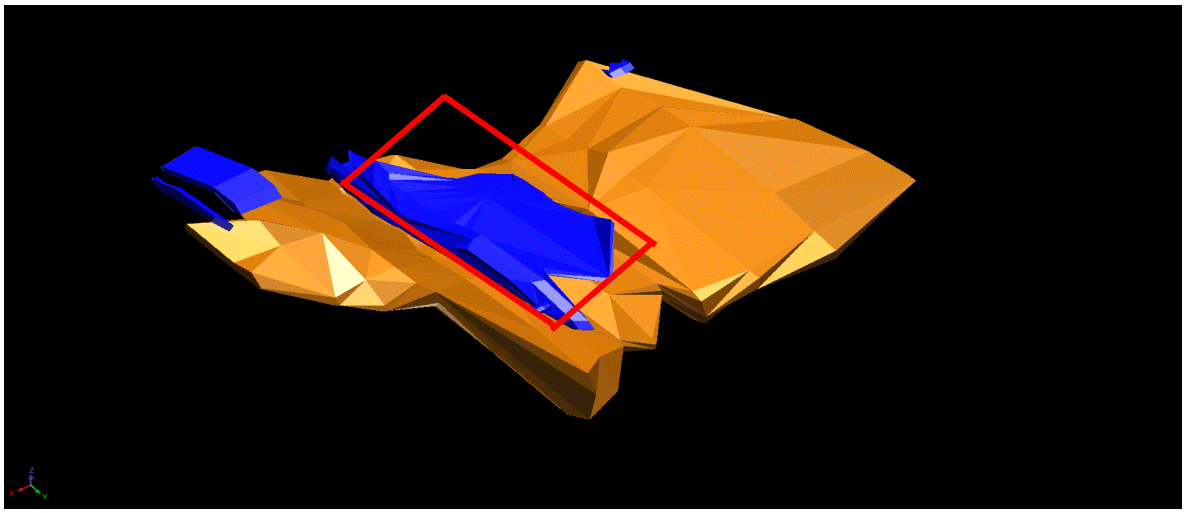


Figure 11 Different ore sheets for project Alpha. The blue part is the top sheet and the orange part is the lower sheet. The sheets are made with a 1% heavy mineral content cut-off. The red area is the test area used for the optimization.

Block model

Two block models are made: one for each approach, while the requirements for the block model differ for both approaches. Both block models have their block height in common, while the block height is mainly dependent on the selectivity of the mining equipment. In general, smaller blocks increase the accuracy. However, it is of no use to decrease the size of the block when the chosen mining equipment cannot mine that particular block without mining other blocks too. In this case, a selectivity of 2 meters is used for a cutter suction dredger, as described in chapter one. Given the high variability in the vertical axis, the blocks are also given this height of 2 meters.

For the first approach, blocks of 75 by 75 meters are used and for the second approach, blocks of 350 by 350 meters are used, based on the average pond size for a dredge mine as described in chapter one. It should be noted that the rule of thumb for block sizes, as described by Hustrulid (2006), states that blocks should not be smaller than one-fourth of the borehole spacing. In this case, the borehole spacing is large, more than 400 meters, while it is a project in development. It is decided to neglect this rule of thumb for the block model of approach one, while block models for Whittle are normally made when the borehole spacing is a lot smaller.

The block models are made in Surpac using ordinary kriging. The data shows a trend in the ore body, with a high graded zone at the surface and a high graded zone in the deeper regions of the ore body. The trend is not strong and changes are generally not abrupt. Therefore, ordinary kriging would suffice as an estimation method (Journel 1989). A variogram is made using a lag of 425 meters, which corresponds to the smallest borehole spacing of around 400 metres. The obtained variogram fit is shown in Figure 12. The following parameters are obtained from the variogram fit:

- Range: 1280 metres
- Nugget: 0.59
- Sill: 0.52

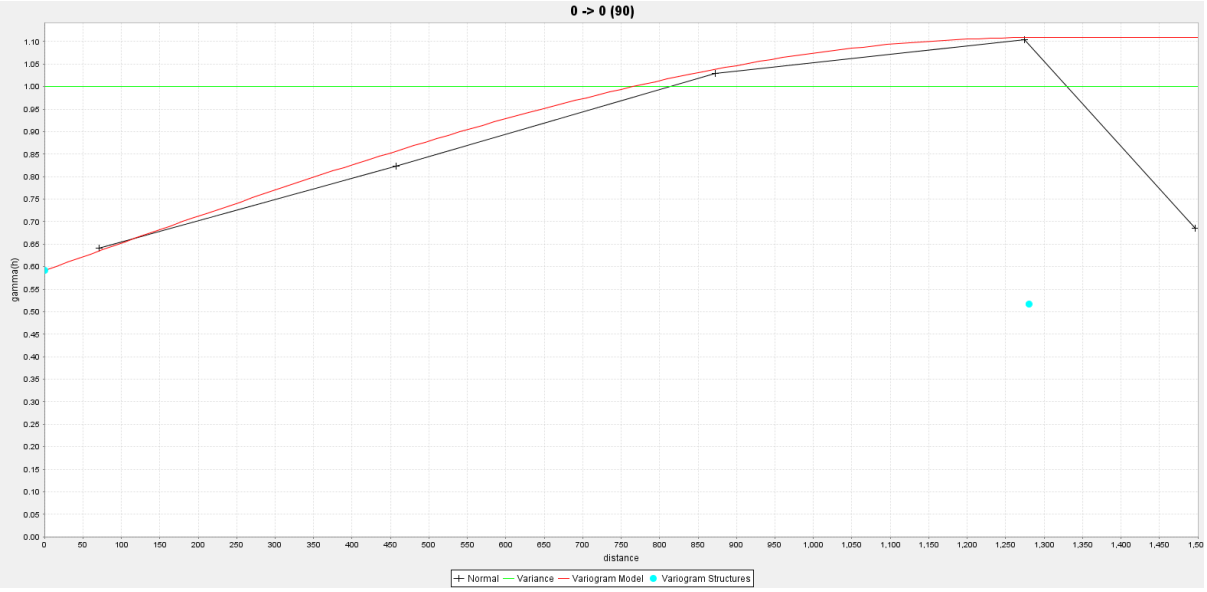


Figure 12 Variogram fit for a lag of 425 metres.

Mining parameters

The mine optimization requires a certain number of parameters to be known, of which some will be discussed here:

- Productivity
- Opex & Capex
- Price
- Discount rate

Productivity

A relatively high theoretical productivity of 4500 m^3/h is chosen for this operation, while the ore body is relatively large. A total availability of 80% is assumed for the dredging equipment and an overcapacity of 30% is required for a flexible operation, based on experience within IHC. Given these numbers, the cutter suction dredger is required to have a capacity of around 8000 m^3/h. Given this capacity and availability, the annual production will be about 39 Mm^3. This information will be used in the Opex and Capex calculations.

Opex & Capex

The operating and capital expenditures for the mining activities are calculated in appendix A. The total operating costs for mining are estimated at 113.800.000 USD per year. These include costs for dredging equipment, transportation and handling, auxiliary equipment, personnel, processing control and facilities. The capital expenditures for the mining activities are estimated at 242.300.000 USD. The processing costs are based on in-house data from IHC and are set at 48 USD/tons of mined heavy minerals and a recovery of 90%. The capital expenditures for the processing plant are also based on in-house experience and are estimated at 450.000.000 USD.

Price

The price is set at 482 USD/tons of heavy minerals sold and are based on a more detailed assessment of the minerals found in the deposit, made by IHC. This will not be discussed in this report.

Discount rate

The discount rate is set at 10%, based on Hustrulid (2006), Stollery (1990) and Stocks (1984) for general mining and heavy industry applications.

Mining data

Given this block model, a grade versus tonnage graph is made, shown in Figure 13. This figure indicates that a higher cut-off would results in a lower total tonnage but an increased average grade, while relatively high graded blocks, of which there are only a few, start having an increasingly strong effect on the average grade. Figure 14 shows the result of a certain cut-off value on the total mine value. The graph shows that the initial drop in value for cut-offs up till 0.75 is relatively low. After 0.75, the drop in value is strong, which indicates that an increase in the cut-off value would result in the loss of money.

Both Figure 13 and Figure 14 show the average slime content, while this is described as an important factor for a dredge mine in the chapter 'Background'. One could say that, based on Figure 13, there might be a negative correlation between grade and slime content. To investigate this possibility, the slime content is plotted against the grade for each sample and shown in Figure 15. The graph shows that there is a higher chance for high slime content at a lower grade than at a higher grade and therefore, supports the previous statement. Further research is required to investigate this relationship; however, this is outside the scope of this report. The graph does show that there is a large variability in the slime content for the 0-2% zone, which is an important zone for mining and lies well within the above mentioned potential cut-off of 0.75 percent. If this project is realized, the slime content will be an important factor to take into account. However, in this report, the slime content will not be part of the optimization process.

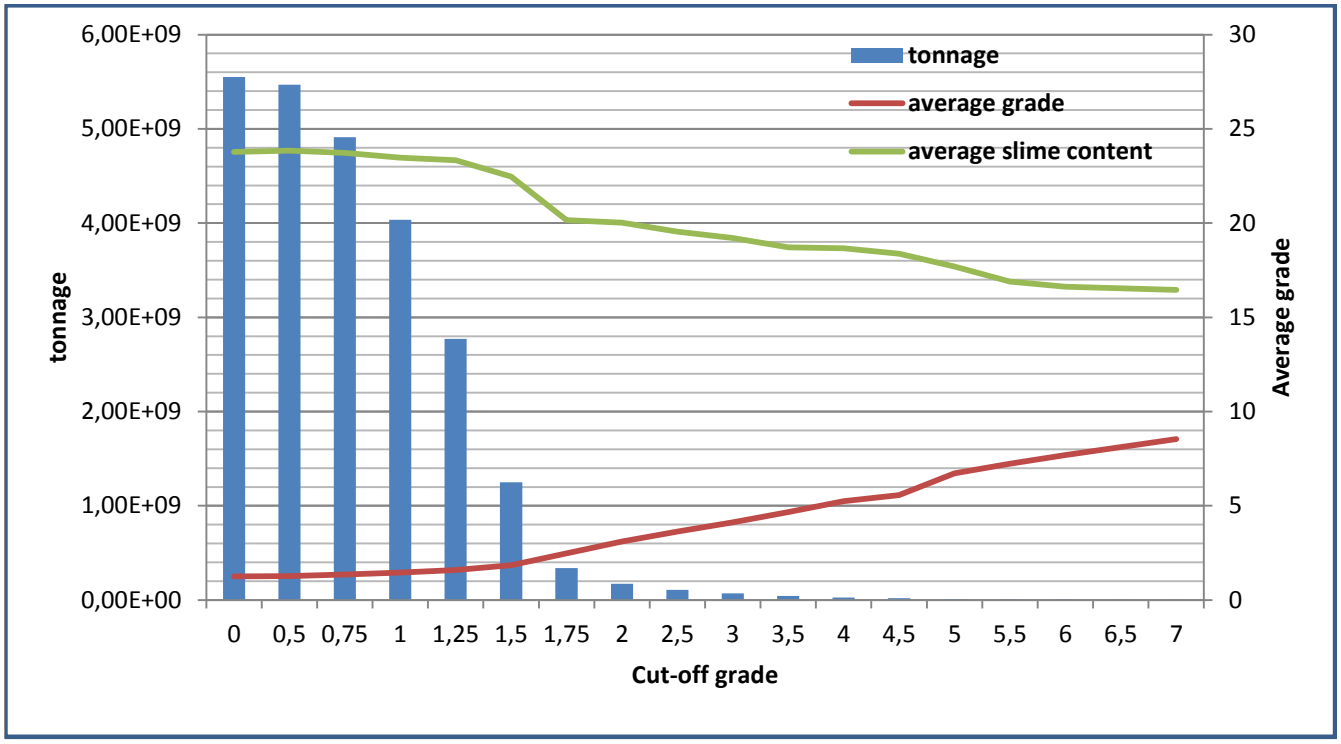


Figure 13 Tonnage, average slime content and average heavy mineral grade at different cut-off scenarios.

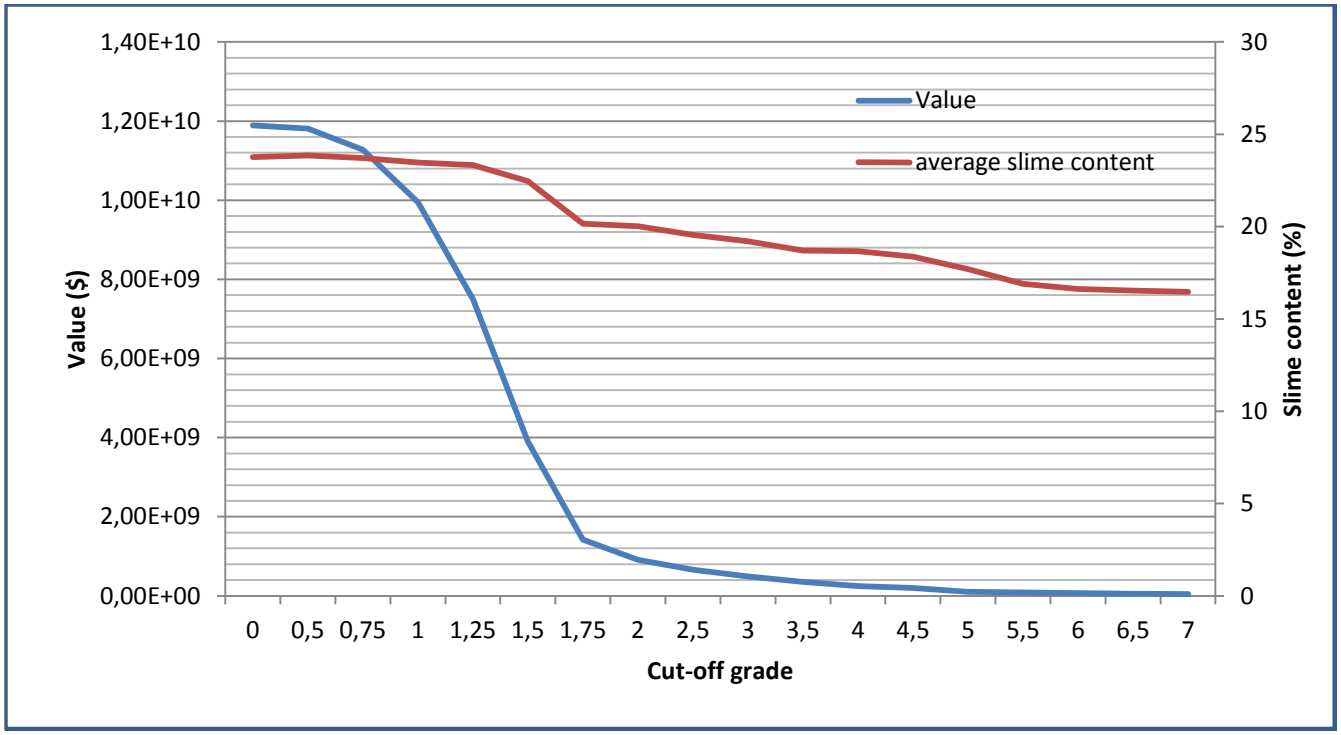


Figure 14 Total value of the mine and slime content versus different cut-off scenarios.

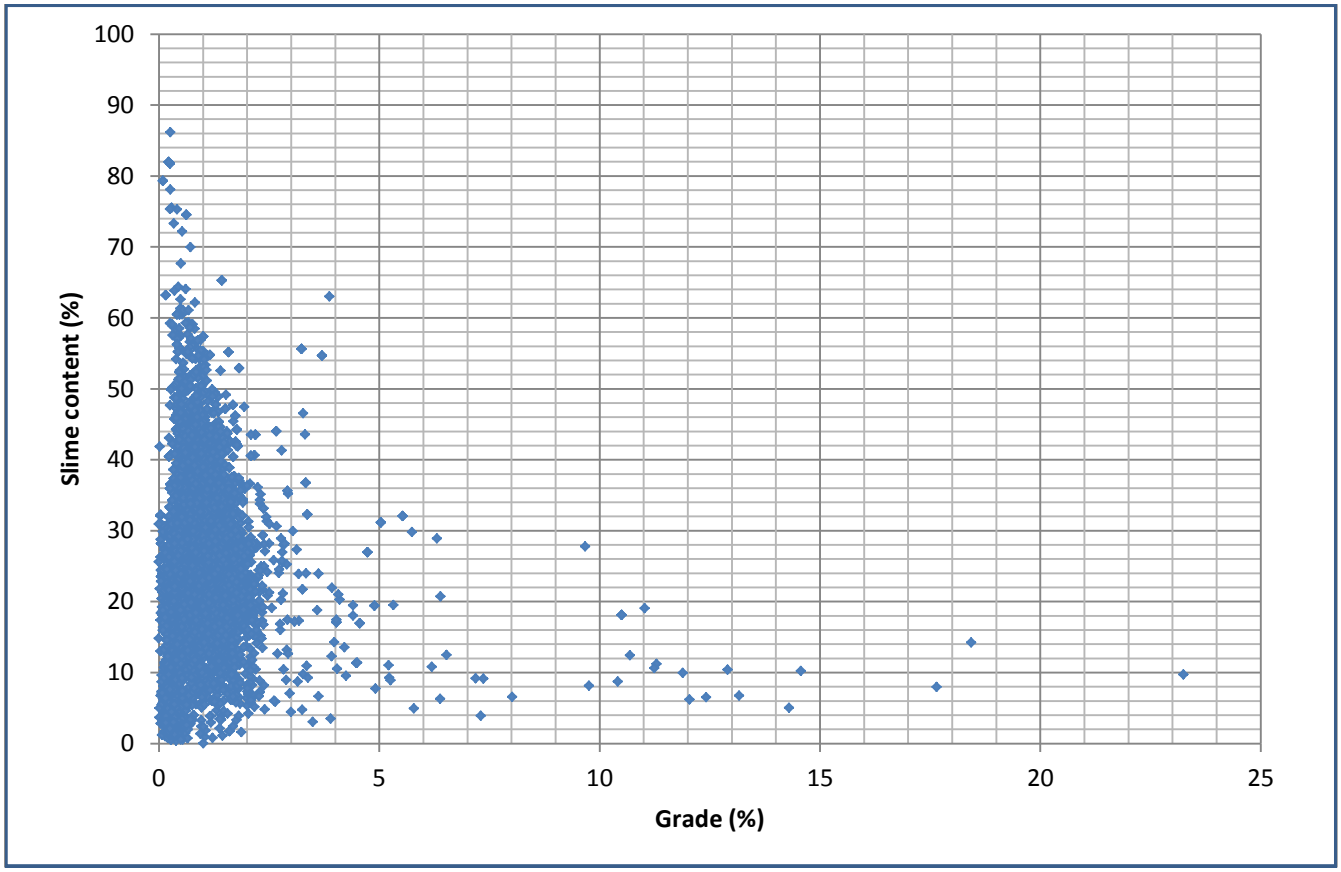


Figure 15 Slime content plotted against the grade for each available sample location.

6 CASE STUDY

PART 2 – APPROACH ONE

The first optimization is done using the core modules of Whittle. These include the automated creation of nested pits. In this report, most of the conventional settings of Whittle are used, while only the core module was available. The conventional settings can be found in the Gold Tutorial for Whittle, standard included in the Whittle core module.

Nested pits

At first, the nested pits are modelled. The revenue factors for creating these nested pits are chosen as follows:

- *0.1 – 0.6* *in 21 steps*
- *0.65 – 0.95* *in 31 steps*
- *1 – 1.5* *in 21 steps*
- *1.55 – 2* *in 10 steps*

Given these revenue factors, a total of 82 pits are generated. Pit 1 and pit 15 are shown in Figure 16 and Figure 17, respectively and show that the pits follow the high graded zones, even when this means that a very small pit is created, for example in the most northeaster corner in Figure 17. This pit is too small to be practical, especially for dredging applications.

When displaying a side view of pit 15 it becomes clear that the top layer is mined first. Unfortunately, Whittle does not support scaled axis, which makes the pits hard to visualize from a side view, given the thin layer compared to its horizontal extend. Therefore, the pit is imported into Surpac, which does support scaled axis. The result is shown in Figure 18. The thinnest pit is not more than 3 meters thick.

The reason why Whittle will mine some top layers first, is that it will consistently mine the high graded zones first and exclusively, even if these high graded zones are really thin. For dredging purposes, it is more interesting to consider the value for the complete mined depth. If the high graded zone lies on top of a very low graded zone, it is not wise to start mining at this point. Unfortunately, this is not considered in Whittle.

Whittle produces two standard economic scenarios: a worst case and a best case. The economic scenarios are based on different final pits, for which a choice has to be made by the user. The maximum NPV for the best case is $3.290.000.000 and $2.450.000.000 for the worst case. The worst case is a better representation of the actual mine design, while a dredge mine can be seen as a strip mine. The outline of the final pit with the highest NPV is shown in Figure 19.

It is not possible to create a practical mining sequence, while these are made using the available nested pits, which can only be adapted using the practical pushback module, which is not included in the license. However, the outline and NPV does give an indication of the size of the project.

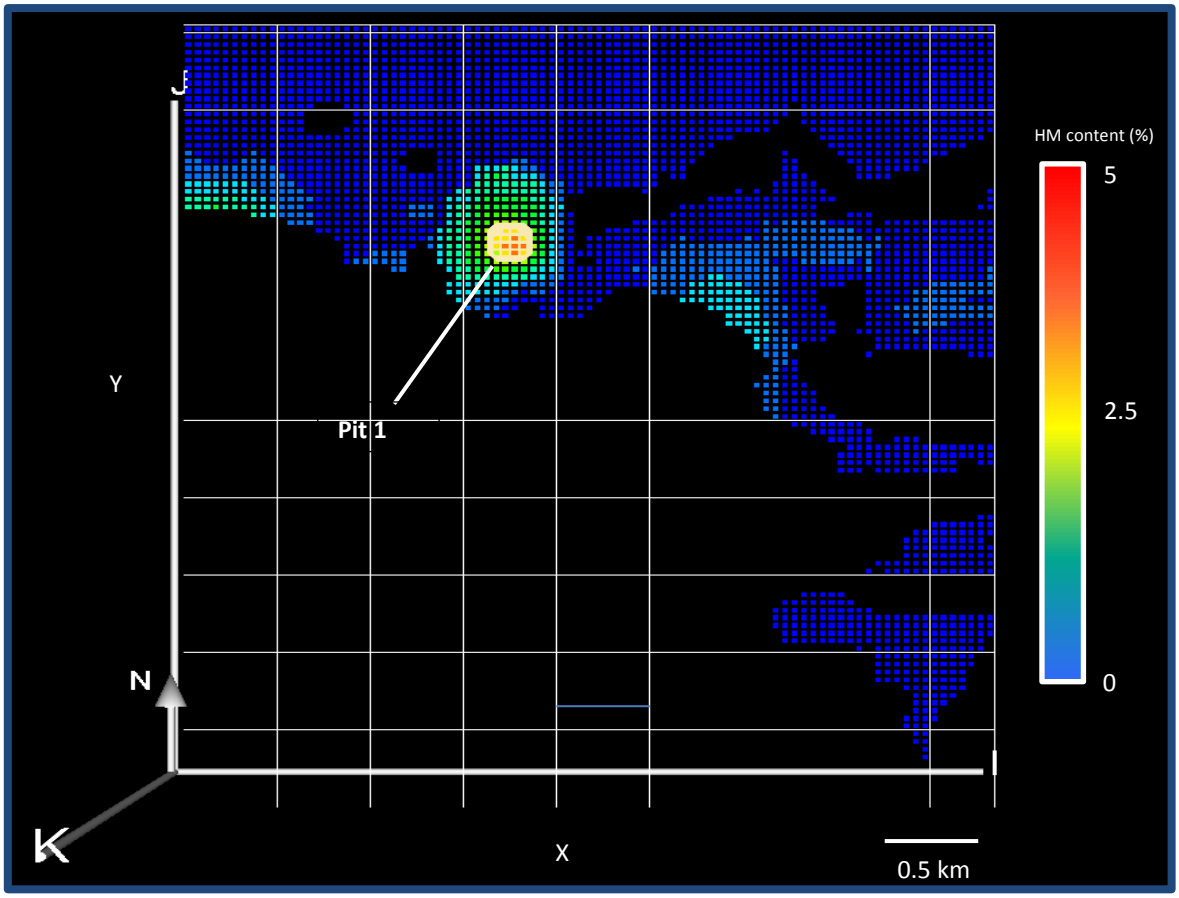


Figure 16 The first pit, enclosing the high graded zones (red dots) and surrounded by lower grades (blue and green dots).

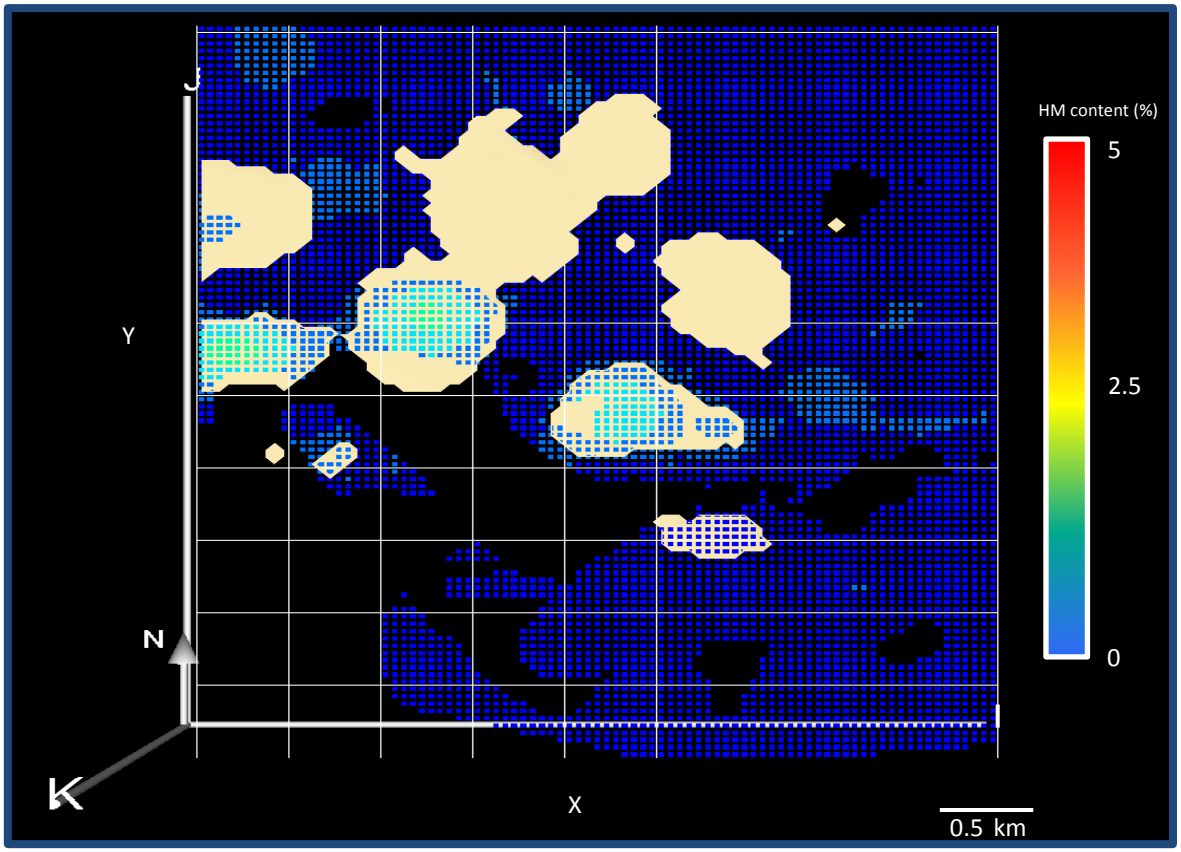


Figure 17 Pit 15, covering an extensive area with multiple pits.

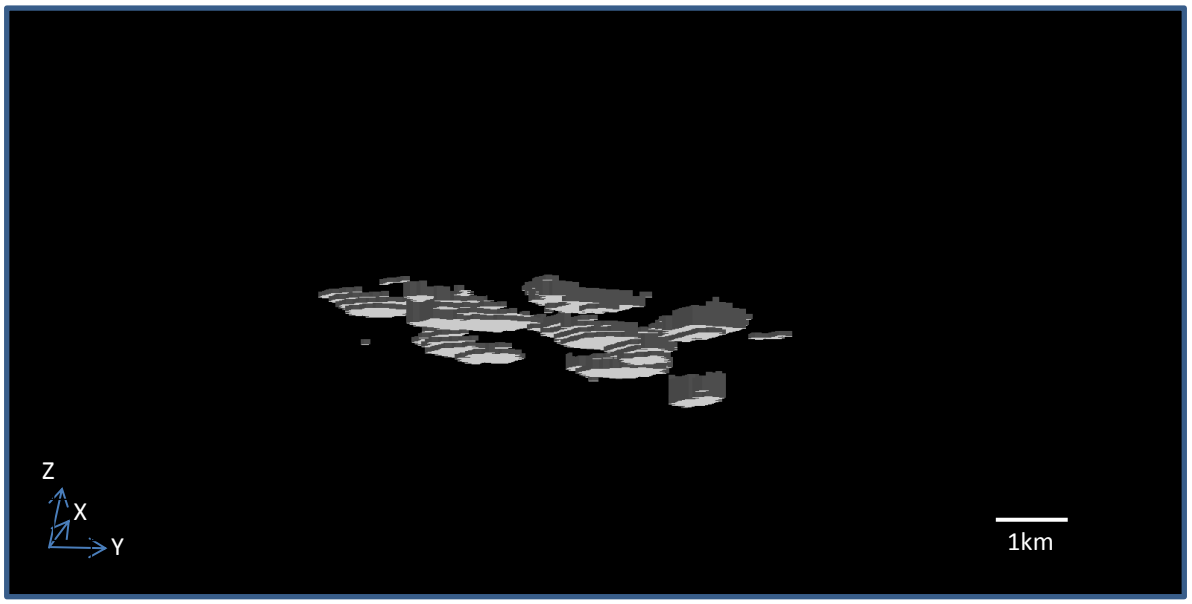


Figure 18 Pit 15 shown in Surpac with the z axis scaled by a factor 25.

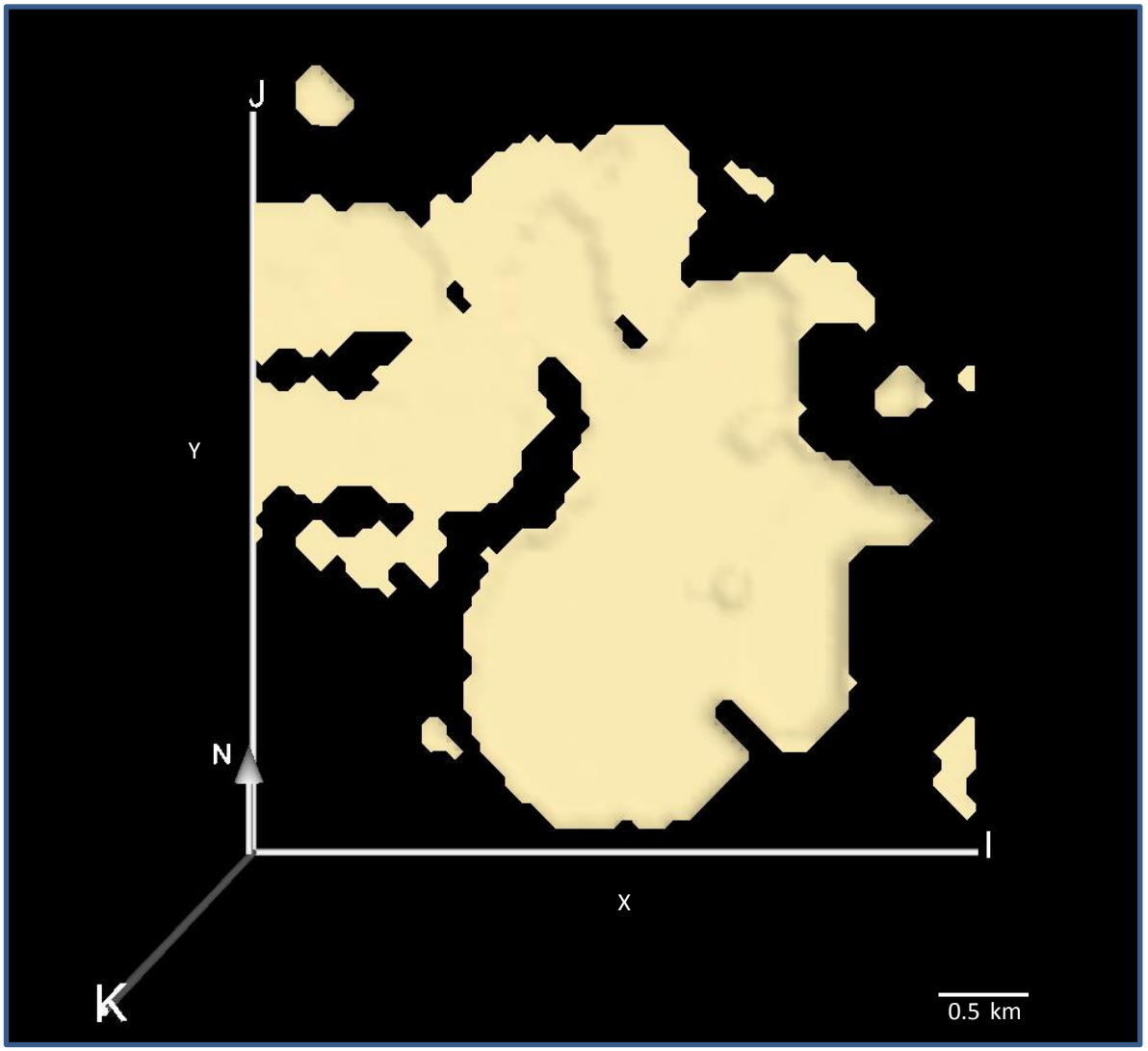


Figure 19 Pit extend of the final pit with the highest NPV

6 CASE STUDY

PART 3 – APPROACH TWO

Approach two consists of two parts: part one evaluates the ultimate depth and part two determines the optimal sequence. Both parts are discussed separately below.

Ultimate depth

The ultimate depth is determined by evaluating the cumulative value per depth for each column. It then chooses the depth with the highest value. Figure 20 shows a surface plot of the ultimate depth. Each face represents a column. The reason why some faces do not look like squares is because the surface plot attaches the faces to each other to make a fitting surface.

What is immediately clear from Figure 20 is that there are depths that far exceed the maximum reach of a cutter. This will have major implications on the design of the dredge mine, while a choice has to be made between mining it all and therefore, facing the challenge of mining it in multiple layers, or mining only the top layer. For this optimization exercise however, it was already chosen in chapter five to exclude this feature from approach two. Therefore, it is decided to neglect this problem.

The average depth difference between the columns is about 10 meters, which is feasible for a dredge mine, while the pond size is 35 times as large. This makes it possible to smooth these differences. However, some columns differ more than 25 meters, which has to be reached step-wise.

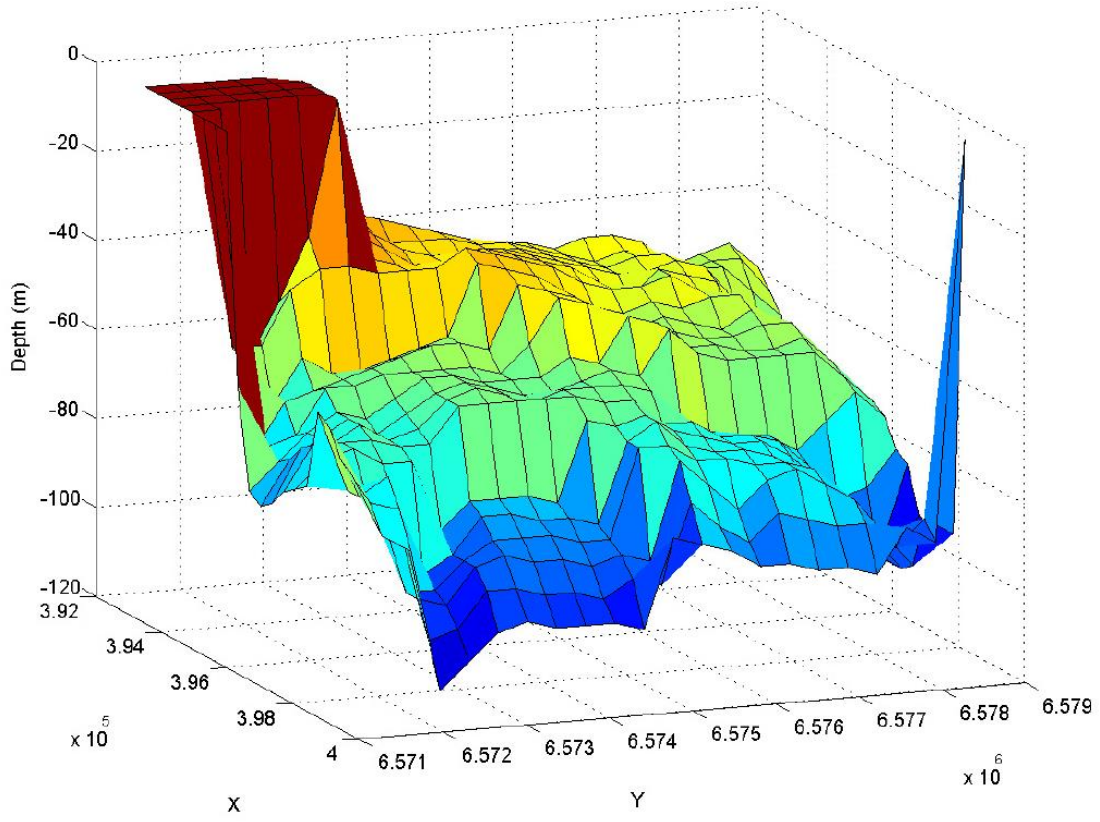


Figure 20 Surface plot of the ultimate depth

Optimal sequence

The optimal sequence is determined by running the adapted version of the Nearest Neighbour Algorithm as described in chapter five. The sequence is shown in Surpac using different colours for the different sequence groups. A total of 20 groups are made to visualize the sequence. Figure 21 shows the mining sequence.

The discounted cash flow per week for the route with the highest NPV is shown in Figure 22. It shows a very high mine life, exceeding the 60 years. A break-even point is determined at 105 weeks. To show the result of choosing an optimal route versus a random route, the NPV's for all routes are shown in a boxplot (Figure 23). The maximum NPV is $2.602.000.000, which is 34% larger than the average NPV of $1.943.000.000. These figures show that for this case, it is economically interesting to use the optimization[1].

The sequence mines a total of 263 columns, which is only 63% of the total available columns. There are numerous routes that mine more columns; however, these routes have a lower NPV. For example, Figure 24 shows a route that mines 338 columns, 81% of the available columns. The result is a NPV that is 10% lower than the maximum NPV. The reason for this lower NPV is that it is a project with a very long lifetime and is therefore, heavily influenced by the discount rate. Blocks that are mined in year 40 have a significant lower value than the blocks that are mined in year one. It is therefore of great importance to create a route through the high graded zones for the starting period of the mine.

Sensitivity analysis

Given the large size of the deposit, a scenario is evaluated with a 50% and 100% increase in production. For the case of a 50% increase in the production, the maximum NPV is $4.754.000.000, which is a 89% increase compared to the base case. Furthermore, the lifetime is decreased by 20 years. For the case of a 100% increase in the production, the maximum NPV is $6.779.000.000, an increase of 170% compared to the base case. An interesting side effect of increasing the production is that an increasing number of columns are mined. For the case of a 100% increase in production, 352 columns are mined, compared to the 268 columns for the base case. This is an increase of 31%.

To investigate the effect of a different price, a scenario is evaluated for a 5% increase and decrease of the base price. A 5% increase in the heavy mineral price will result in an NPV of $2.880.000.000, an increase of 14.7% compared to the base case. The ultimate depth does not change for this case, which indicates that the ultimate depth is not very sensitive to a change in price. The large increase in NPV can be explained by the fact that the high graded zones that are reached in the beginning are less affected by the discount rate. For the case of a 5% decrease in price, the NPV will decrease with 4.9% to $2.388.000.000. A possible explanation for this relative small decrease is that the discount rate will restrain some of the decrease in price.

[1] *Please note, that the economic figures in this report are based on estimations and do not represent the actual value of the deposit. Furthermore, a mine life that exceeds the 60 years requires substantial intermediate capex, which are also not included in the assessment.*

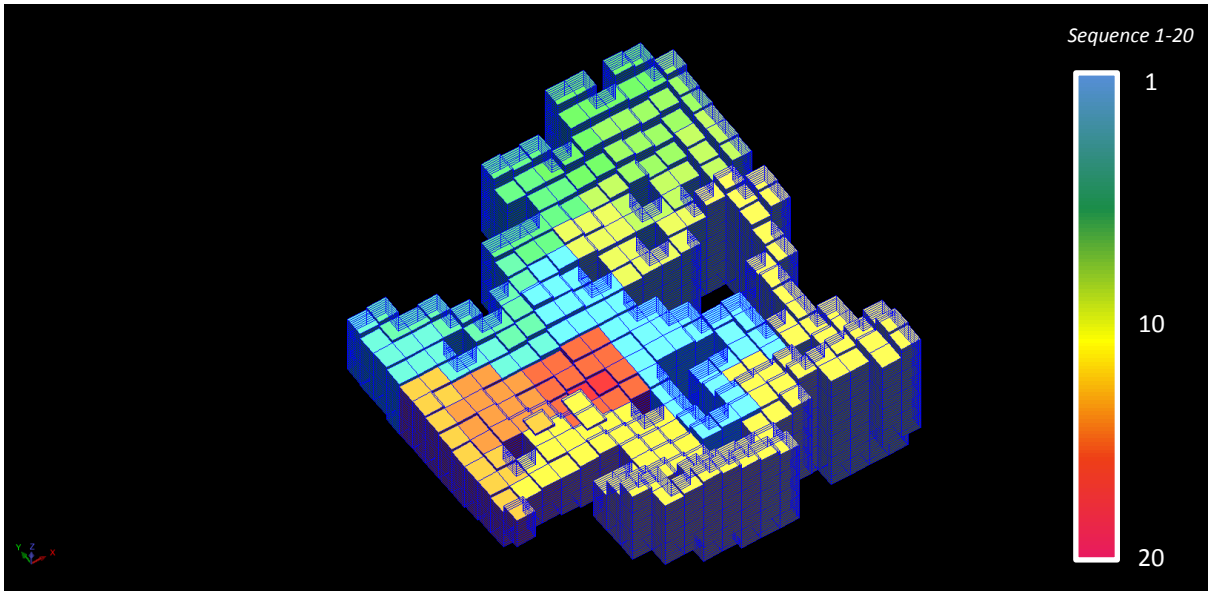


Figure 21 Mining sequence for the highest NPV, displayed in Surpac. Blue is mined first, then green, yellow, orange and red. The axis orientation is shown in the lower-left corner.

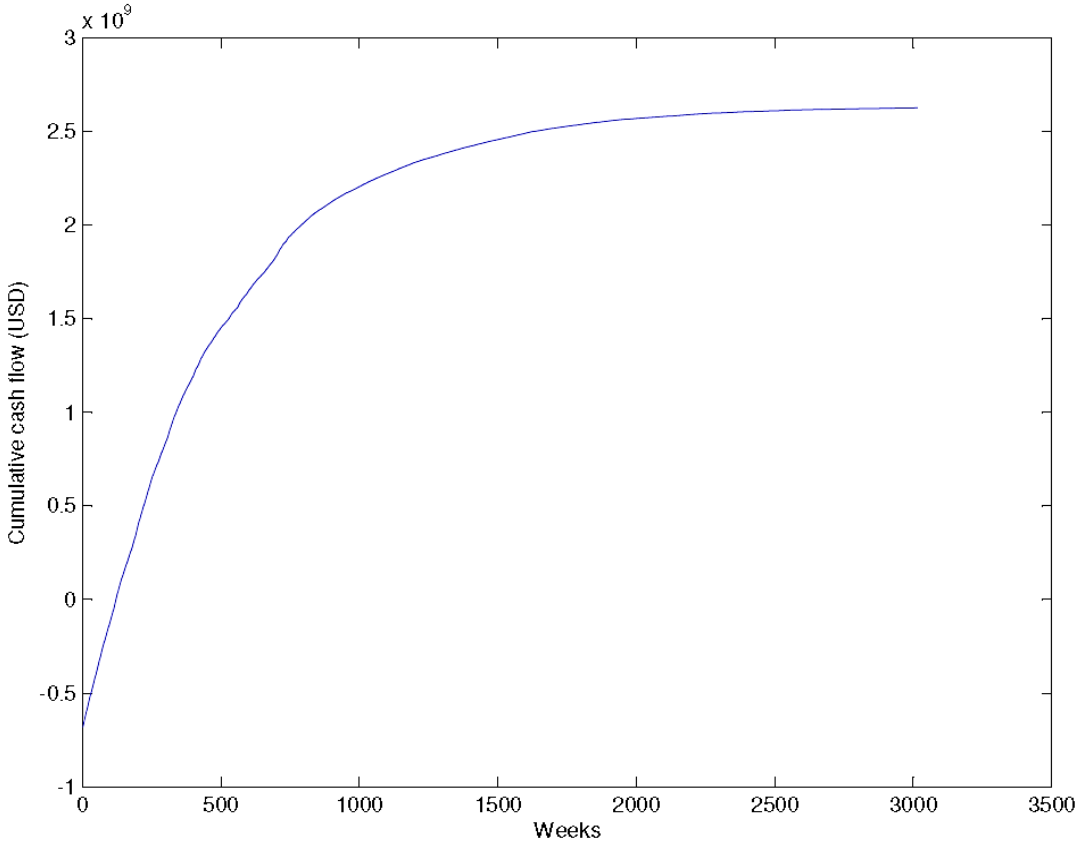


Figure 22 Cumulative discounted cash flow per week for the route with the highest NPV.

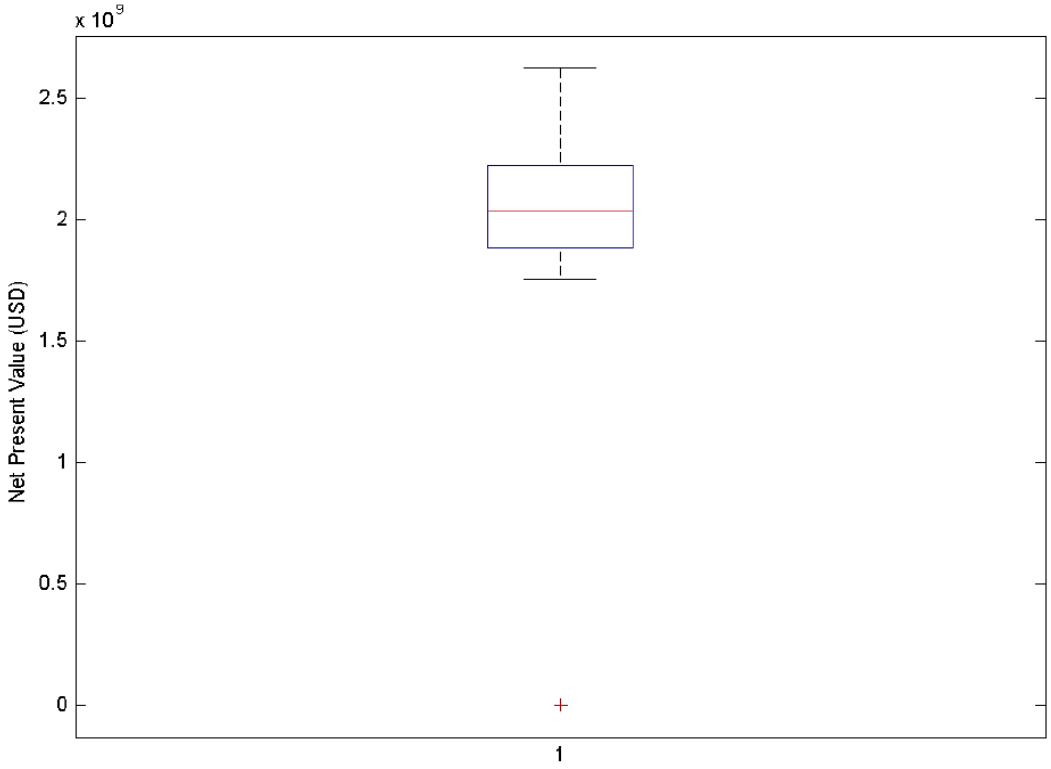


Figure 23 Boxplot of the Net Present Value's for different starting points

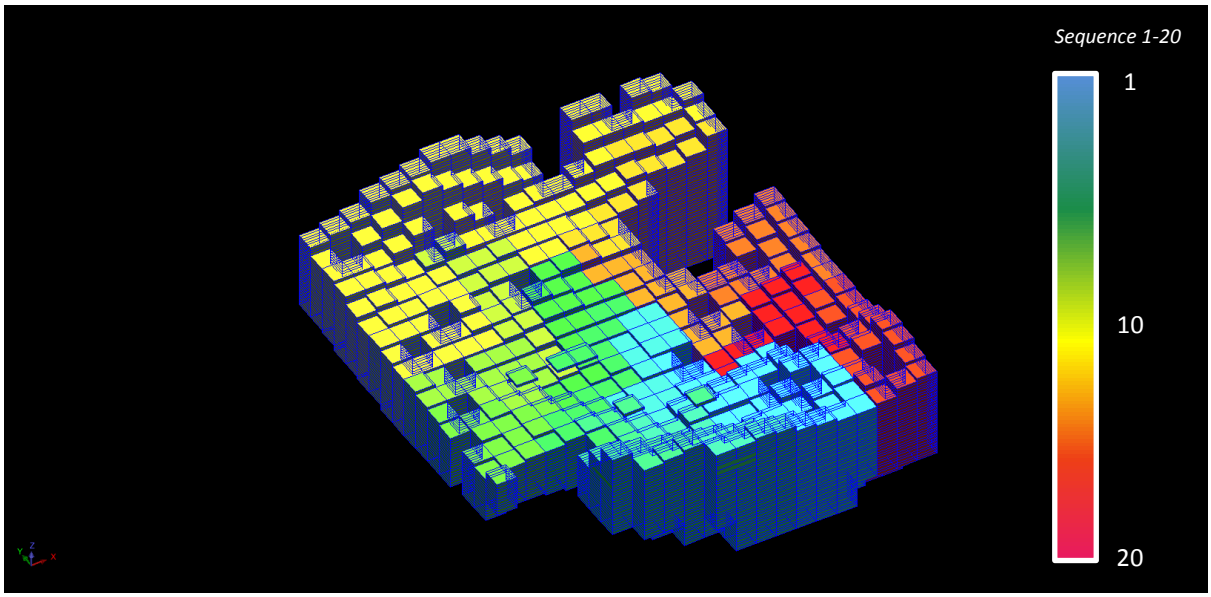


Figure 24 Mining sequence for the longest route, displayed in Surpac. Blue is mined first, then green, yellow, orange and red. The axis orientation is shown in the lower-left corner.

7 DISCUSSION

In this chapter, the usability of the conventional and new approach for optimizing dredge mines is discussed. Chapter 3 states the critical design criteria for an optimization. These criteria will be used as a guideline to discuss the usability of the approaches. Furthermore, the power of the new algorithm is discussed.

Approach one

The first approach is the conventional approach, using Whittle to determine an optimal mining sequence. The software is specially developed for open pit mines, which reduces the flexibility of the program. The software is discussed for each critical design criteria below.

The core module of the software does not meet the design criteria for depth control, while it is only possible to select nested pits as pushbacks. The case study showed that the nested pits only focus on values, creating some thin nested pits for thin high graded zones. These thin pits are impractical to mine for a dredger, especially if the pit is vertically extended in a later stadium of the mine life, while this would mean that the dredger would have to go back to that spot. For that matter, a dredging operation is less flexible than a conventional dry mining operation. It should be noted that the optimization is only tested for the core module. Expansions of the program, for example the 'Practical Pushback' expansion, might enable the user to specify some restriction for the nested pits, improving the suitability of the program. However, this is not included in this report and would be the scope of further research.

The creation of multiple ponds is generally not preferred for dredge mine designs, while this would have significant impact for the design of the processing plant. One possibility to circumvent the creation of multiple pits by the program is to select a nested pit that is not made of multiple pits. However, this would severely limit the options and would also decrease the possibilities for pit sequencing. It is possible to use the software for creating an optimal outline of the mineable area. However, this would mean that large potential of the software would remain unrealized.

The mining direction affects the usability of the pit sequence, while the dredger works most efficiently in one direction, as described in chapter one. Whittle is developed for conventional mines, that work with benches, for which a preferred mining direction does not exists. Therefore, Whittle will only base its mining direction on the value of the surrounding blocks. The case study showed this for project Alpha.

The combination of the above mentioned factors makes the software impractical for a complete optimization of a dredge mine. It is possible to obtain an indication of the outline of the optimal mine using Whittle, however, it should be kept in mind that this outline could be based on thin top layers of high graded sheets, which are also impractical to mine using a dredger.

Considering the limitations of Whittle, it was chosen to develop a new method for optimizing dredge mines. The reason for developing a new method was twofold:

1. The design of a dredge mine significantly differs from the design of a conventional surface mine.
2. The design of a dredge mine is easy to simplify, which significantly decreases the complexity of the software and therefore, also the time it takes to develop the software.

By simplifying the problem, it was possible to develop an optimization code in three weeks. However, these simplifications have an effect on the usability and flexibility of the software. This will be discussed in the next section.

Approach two

Design criteria

The second approach should be seen as a first step in developing an optimization tool, exclusively for dredge mines. The usability of the software is discussed below in terms of the critical design criteria.

The ultimate depth is determined per column. The benefit of this simplification is that it prevents the sequence to go back to an earlier mined spot to extend its depth. However, the implication of using this simplification is that large depth differences may occur, which are geo-technically speaking impossible. The mining depths shown in the case study (Figure 21) are relatively consistent, but do show areas of high differences that would be impossible to maintain in practice. Further research should be conducted to improve the code for a more practical ultimate depth.

Multiple ponds are not created by the software, while the algorithm determines a route for a pond to be followed. The case study showed that for large deposits, the algorithm will leave some columns unmined, while their present value has become relatively small. This might not be a realistic case, while the price of the mineral will also be probably higher in the future, compensating for the large discount factor. However, this is more a limitation of the economic model than of the approach itself and will not be further discussed here.

The mining direction criterion is met, however, results in a limitation, namely that the ponds can only migrate to a maximum of four possible directions, while in reality, more directions may be chosen. Further development of the code could include an option to expand the possible mining directions. The result of this limitation is that the algorithm might reach a dead end earlier than would be strictly necessary.

Besides the design criteria, the block model limitations greatly influence the usability. The limitations on the block model size will not be practical in most situations, while projects in the end of the exploration phase tend to have small borehole spacing. The accuracy obtained in this way needs to be captured in the block model, which is not possible for this approach.

Algorithm

The algorithm optimizes for the NPV in a passive and active way. The active way is the part of determining the best next column based on the NPV of a potential route. The passive way is the part where it chooses the route with the highest NPV. However, the algorithm does not truly optimize for the NPV, while the discount rate can only be applied for a maximum number of years for determining the potential route. After this maximum number of years, the algorithm will be more focussed on the recovery factor.

The power of the NPV method significantly decreases for projects with a very long lifetime, while the discount factor becomes very high. Therefore, further development should be focused on the optimization of the recovery factor.

8 CONCLUSION

The purpose of this report is to investigate the implications of the design of a dredge mine on mine optimizations and evaluate possible approaches for optimizing such dredge mines. To discuss the implications, a background study was done first to state the important characteristics of a dredge mine and briefly discuss the basics of mine economics. Six parameters were selected as important design criteria for a dredge mine:

- Slope
- Pond size
- Mine direction
- Backfill
- Processing operation
- Material properties

The optimization that was chosen in this report is by Net Present Value, which results in defining an ultimate pit and optimal sequence for a mine. The implications of the design of a dredge mine on the optimizations were stated as three criteria that would have to be met by the optimization. These criteria are:

- **Depth control**: the pond has to be at its final depth the moment it is mined. If that would exceed the maximum reach of the dredger, the depth can be reached by mining the entire area in layers.
- **Multiple ponds**: Only one pond can be mined in turn, while the processing plant is also located in the pond.
- **Mining direction:** The dredger works most efficient when going into one direction for a certain period.

For these criteria, two approaches were evaluated. The first approach is the conventional approach used for surface mine applications, for which the software package Whittle was used. Research indicated that the core module of Whittle may not be the right choice for optimizing dredge mines, while it lacked the following important functions:

- Function to force the nested pits to be at their final depth
- Function to force the nested pits to expand one pit only

Approach two was developed using Matlab® specifically for the optimization of dredge mines, while its design differs from an open pit mine. By defining the ultimate pit as an ultimate depth for each column, the first criterion is met. An optimal route is determined for one path; therefore, no multiple ponds are created. The second approach has four important limitations:

- The depth difference between columns could be too high to be stable
- Premature reach of a dead end or enclosing a group of non-mined columns
- Limitation on the size of the blocks
- Lack of function ability to mine the area in layers

The case study showed that the core module of Whittle was able to produce an indication of the value of the project and a possible outline; however, it was not able to produce a possible mining sequence because of the lack of earlier mentioned functions. Approach two showed it was able to produce an optimal route that is feasible for a dredge mine. Furthermore, the NPV for the route was higher than that of the worst case scenario of Whittle. The case study also indicated that for long-term projects, the NPV might not be the right choice to optimize for, while the discounting factor will get very high, which results in a low recovery.

9 RECOMMENDATIONS

This report can be seen as an introductory paper for the optimization of a dredge mine. It includes a first assessment of the use of the core module of Whittle for optimizing a dredge mine and introduces a new approach for this optimization. To further discuss the suitability of Whittle for dredge mines, a complete assessment should be made with the full license of Whittle. Furthermore, the new method should be tested and evaluated thoroughly before one can continue with the development of the software.

The development of the code should include the possibility of importing smaller sized blocks, hereby incorporating more accuracy of the block model into the software. One possible way of doing this is to define an overlaying pond that migrates through the blocks. To reduce the computing time, one may group the blocks intro groups with an average value or absolute value, as to define the best group to migrate to, for which then a complete optimization might be done.

Furthermore, the development might focus on including a function to make it possible to mine the area in layers. This might be done by constraining the first ultimate depth to the reach of the dredger and then re-run the program for the next layer. Another possible approach would be to evaluate the value of the second layer and determine the combined area to see if it is feasible to mine the entire layer or only parts of it.

It may also be interesting to add a feature that defines the ultimate depth that also considers the depth differences between the columns. This might be accomplished by evaluating the optimal depth for a larger area and then changing the ultimate depth per column, but not more than the maximum allowed difference in depth.

For large project, the software should be able to optimize for the recovery factor. It could be possible to include an optimization for NPV for the first years and then continue with an optimization for the recovery factor for the remaining years. In this way, the high graded zones are mined first and a large recovery factor is reached.

Regarding project Alpha, it may be interesting to incorporate a slime factor in the optimization process, to prevent the software for mining the high slime zones. This may be done by adding a penalty that depends on the slime content for each column or by including processing costs that differ for the slime content.

10 LITERATURE LIST

Books

Grotzinger, J., Jordan, T., Press, F. and Siever, R. (2007). *Understanding Earth*. 2nd ed. W.H.Freeman and Company.

Hustrulid, W. and Kuchta, M. 2006. *Open pit mine planning and design*. 2nd ed. Taylor and Francis.

Hartman, L.H. and Mutmansky.(2002). *Introductory Mining Engineering*. 2nd ed. JOHN | WILEY & SONS, Inc.

Hull, J. (2012). *Options, futures and other derivatives*. 8th ed. Pearson Education, Boston

Papers

Alessandri, T.M., Ford, D.N., Lander, D.M. and Leggio, K.B. (2004). *Managing risk and uncertainty in complex capital projects*. The Quarterly Review of Economics and Finance, vol. 44, no.5,pp. 751-767.

Bray, R.N. (2009). *A guide to cost standards for dredging equipment*. CIRIA.

Connolly, E. and Orsmond, D. (2011). *The mining industry, from bust to boom.* Reserve Bank of Australia

Cook,B.(2011). *Geological Excerpts and Explanations from Bent Cook*. Exploration Insights.

Dassault Systemes. (2013). *Annual Report*.

Dykstra, C.J. and Lockhorst, G. (1982). *Dredging, an attractive economical alternative for oil-sand mining*. Beaver Dredging Company Ltd.

Gilbert, L. (1992). *The Traveling Salesman Problem: An overview of exact and approximate algorithms*. European Journal of Operational Research 59: 231-247

Godoy, M. and Dimitrakopoulos, R. (2004). M*anaging risk and waste mining in long-term production scheduling of open-pit mines.* Society for mining, Metallurgy and Exploration, Volume 316, p: 327

Hahsler, M. and Hornik, K. (unknown). *TSP – Infrastructure for the Traveling Salesperson Problem.* Wirtschaftsuniversität Wien

Hajdasinski, M.M. (1988). *Optimization of mine size and mine life in a historical perspective.* Mining Science and Technology, Volume 7, p: 305-310

Hall, J. and Nicholls, S. (2007). *Valuation of mining projects using option pricing techniques*. JASSA, vol. 1, no.4, pp.22-29.

Hanson, N., Hodson, D. and Mullins, M. (2001). *Skin analysis in the selection of final pit limits.* Strategic Mine Planning Conference, Perth.

Herchenhorn, W., Muijen, H., Ouwekerk, R., Boor, M., Verkaik, C.J. and Wit, J.W. (2005). *From heavy minerals mining to a wet dredge mining operation: a succesful story in Brazil.* Western Dredging Association.

Jones, G. (unknown). *Mineral Sands: An overview of the Industry*. ILUKA.

Journel, A.G. and Rossi, M.E. (1989). *When do we need a trend model in Kriging.* Methematical Geology, Volume 21.

Lerchs, H. and Grossmann, I.F. (1965). *Optimum design of open-pit mines*. CIM Bulletin 58:47-54

Matai, R., Prakash Singh, S and Lal Mittal, M. (2010). *Traveling Salesman Problem: an overview of applications, formulations and solution approaches.* InTech

Rego, C., Gamboa, D., Glover, F. and Osterman, C. (2010). *Traveling salesman problem heuristics: leading methods, implementations and latest advances.* European Journal of Operational Research, Volume 211, Issue 3, p: 427-441

Sainsbury, G.M. (1970). *Computer-based design of open cut mines.* Proc.Aust. Inst. Min. Met., volume 6, p: 49-57

Smith, M.L. (2001). *Using Milawa/4X as a starting solution for mixed integer programming optimization of large open cut production schedules.* Strategic Mine Planning Conference, Perth.

Stocks, K. (1984). *Discount rate for technology assessment*. Butterworth & Co (Publishers) Ltd.

Stollery, K. (1990). *The discount rate and resource extraction.* University of Waterloo

Ware, C.I. (2007). *Back to basics at Richards Bay Minerals'mining operation: examples using six sigma methodology*. The 6th International Heavy Minerals Conference.

Wharton, C. (2004). *The use of extractive blending optimization for improved profitability.* Gemcom

Whittle (1999). *Four-X™ Stategic planning software for open pit mines: Reference Manual.* Whittle Programming Pty Ltd.

Whittle, J., Dimitrakopoulos, R. and Ramazan, S. (2004). *Optimisation and Risk Assessment in Mine Design and Production Scheduling.* Whittle Programming Pty Ltd.

Whittle (2013). *Introductory Gold Tutorial.* Dassault Systemes GEOVIA.

Internet

CEPA Environmental Registry. 1999. *Canadian Environmental Protection Act*. Localised on 4-06-2014 on the World Wide Web: http://cise-scie.ca/lcpe-cepa/default.asp?lang=En&n=CBE3CD59-1&offset=8&toc=show

EuDA 2014. *About dredging*. Localised on 10-06-2014 on the World Wide Web: http://www.european-dredging.eu/Definitions

APPENDIX A – COST ESTIMATION

The costs[2] of the operation are split into six sections, namely:

- Mining equipment
- Transport and handling systems
- Auxiliary equipment
- Personnel
- Processing Control and Information Systems
- Infrastructure

These sections will be discussed separately below. For each section, the costs are split into into the capital costs, the depreciation and interest costs per week if available and the maintenance and repair costs per week, including the following items:

- Maintenance and repairs
- Standard spare parts
- Technical consumables and cleaning items
- Steel wires
- Freight costs in respect of spare parts for repairs
- Directly attributable staff costs during repairs

The costs also show the other operational costs, which include the following if applicable:

- Labour
- Fuel/power consumption
- Lubricants
- Tires (if applicable)
- Wear parts

The labour costs for the operating of the mine are based on experience from IHC. A more detailed study should include a more in-depth discussion on this topic.

[2] *The information in this appendix is either obtained from CIRIA (2009) dredging equipment costs or InfoMine (2010) mining equipment costs.*

Mining equipment

Given an effective production of 8130 m^3/h and a depth of 25 meters (PARAGRAPH), the cutter model chosen is the B9029 (Figure 25). Two B9029 are required to achieve the 8130 m^3/h. The B9029 has three on board pumps, which enables it to pump up to 10 kilometers**.** Therefore, no additional booster station is required for pumping the feed to the recovery plant.

The labor costs include one operator, one deckhand and one engineer for the engines per shift.

Table 1 Mine equipment selection for project Alpha

Type of cutter	Amount	Capex (€)	Depreciation & interest (€ per year)	Maintenance and repair (€ per year)	Operational costs (€ per year)	Service life
B9029	4	206.000.000	39.740.000	15.739.000	39.708.900	20

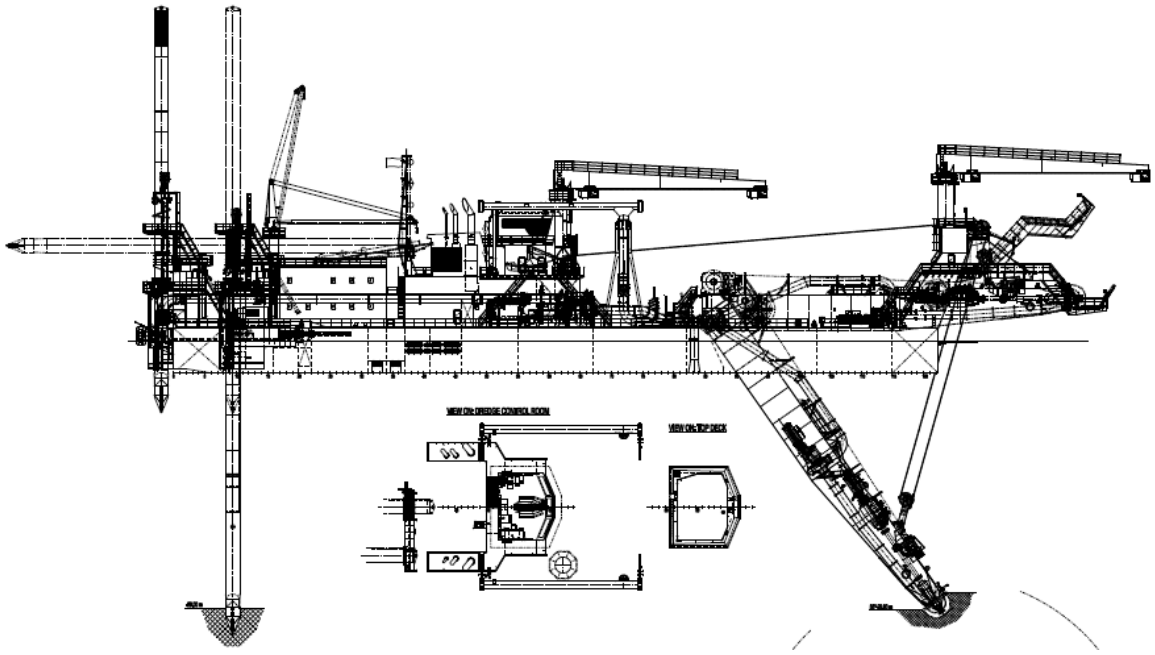


Figure 25 The B9029 Cutter Suction dredger.

Transportation handling system

A floating pipeline is required to connect the dredger with the processing plant. The distance is estimated at 1000 meters, adding some flexibility to the design. Furthermore, the pipe diameter is set at 900 mm to support the high throughput.

The initial tailings pond has to be maintained and monitored. The equipment used for this operation will be two bulldozers (Cat D6T) and a small hydraulic excavator (Cat 308D CR). No equipment will be required to install the initial tailings pond, while contractors are hired to do this job.

Table 2 Equipment required for the transportation and infrastructure for a wet operation

Type of equipment	Amount	Capex (€)	Depreciation & interest (€ per year)	Maintenance and repair (€ per year)	Operational costs (€ per year)	Service life
900 mm floating pipeline	1 km	2.900.000		197.333	360.000	2
Cat D6T	2	1.800.000		695.000	1.138.000	15
Cat 308D CR	1	562.100		205.500	333.000	15

Auxiliary equipment

One of the most importing things in a dredging operation is maintaining the production of the dredger. For that matter, the dredgers have to be maintained in a proper way. While there are two cutters, at least one service pontoons (38x13x3.5m each) and at least one multi-purpose pontoon (115t pulling force, 2848 kNm) are required for this operation. In order to position the service- and multi-purpose pontoon, tugboats are necessary. These may be small in size. Here, a 2x 300 kW tugboat is proposed. For quick access at the cutters, high speed crew & survey vessels will be necessary. These vessels may also be used in the initial tailings pond for surveying operations. Three of these vessels are proposed to be sufficient for this operation.

Table 3 Auxiliary equipment and infrastructure required for a wet operation

Type of equipment	Amount	Capex (€)	Depreciation & interest (€ per year)	Maintenance and repair (€ per year)	Other Operational costs (€ per year)	Service life
Dredge support equipment						
Service Pontoon	1	500.000	98.000	39.000		15
Multi-purpose pontoon	1	4.920.000	964.000	768.000		15
Tugboat (2x 300 kW)	2	1.800.000	353.000	328.000		15
High speed crew & survey vessel (350 kW)	3	1.890.000	317.000	344.000		15

Personnel

The personnel costs for the mining equipment are incorporated into the mining operating costs. Besides the employees for the equipment, other employees are required for a mine operation. However, only personnel that is essential for the operation is considered in this scope. Additional overhead personnel are not included, for example a receptionist. If a function requires a night shift, two personnel members will be put on the task, one during the day and one during the night.

There will be two fulltime fire fighters that are also trained to perform first aid. Two maintenance engineers will also be trained to help in case of an emergency.

- Production manager (1x, night shift)
- Tailings management
  - Officer (1x, no night shift
  - Engineer (2x, night shift)
- Mining Engineer (2x, no night shift)
- Geologist (1x, no night shift)
- IT
  - Officer (1x, no night shift)
  - Maintenance (2x, night shift)
- Maintenance/Utility
  - Officer (1x, no night shift)
  - Engineers (10x, night shift)
- Emergency crew (2x, night shift)
- Terrain management
  - manager (1x, no night shift)
  - Terrain operator (1x, night shift)
- EIA monitoring
  - Officer (1x, no night shift)
  - Engineer (2x, no night shift)

The costs for these employees are listed in Table 4.

Table 4 Labour costs

Type of personnel	Amount	Operational costs (€ per year)
Production manager	2	306.600
Tailings Officer	1	131.400
Tailings Engineer	4	324.120
Mining Engineer	2	262.800
Geologist	1	131.400
IT officer	1	131.400
IT maintenance	4	350.400
Maintenance Officer	1	131.400
Maintenance engineer	20	1.795.800
Emergency crew	4	320.000
Terrain manager	1	131.400
Terrain operator	2	201.500
EIA officer	1	131.400
EIA engineer	2	201.500

Processing Control and Information Systems

The mine will make use of the latest developments in Process Control Systems (PCS), enabling real-time visualization of the mines operation and real-time updates of the geological model. The operator of the shovel or cutter sees his new goals and updated information directly on his screen. The connection is made by using direct beam streamers; therefore, no cables will have to be installed.

The process control system will be monitored by a geologist and a mining engineering, reducing the chance of mistakes. The entire operation is overseen in the main office building, which also houses the server and two fulltime IT specialists per shift.

The operational costs are neglected in this study, while these (electricity for example) are minor with respect to other operational costs.

Table 5 Processing and Information systems requirements

Type of equipment	Amount	Capex (€)	Depreciation & interest (€ per year)	Maintenance and repair (€ per year)	Operational costs (€ per year)	Service life
Heavy Duty Server	1	150.000		10.400		5
PCS	1	350.000		2.500		5

Infrastructure

The infrastructure costs are split into the following parts:

- Utilities
- Buildings
- Other infrastructure

The overall costs for the infrastructure are shown in

Utilities

Power, fuel and water are essential for a sustainable mine production. These subjects will be discussed separately below.

Power

It is assumed that a sufficient 260 kV power grid is available, for which 260 kV/25 kV power step-down transformers will have to be installed and a 25 kV cable infrastructure has to be laid down to connect the cutters with the power grid. Additionally, 25 kV/400V step-down transformer (substations) are required for the housing and smaller equipment.

There will be 8 substations for the following locations:

- Processing plant
- Extraction and tailings area
- Main utility area/Main offices
- Mining area
- Initial tailings pond
- Product storage

A total of 6 km of 260 kV power transmission lines are required, together with up to 20 km of 25 kV power transmission lines, one 260 kV/25 kV transformer, eight 25 kV substations and at least 20 km of 400 V lines. Furthermore, back-up generators will be installed to guarantee power supply to the critical points to maintain the production.

Fuel and water

For dredging, the fuel requirements are low and two fuel tanks of 56.000 L each will be sufficient for the mining operation. Water tanks are necessary for an emergency supply of fresh water. Two water tanks of each 64.000 L will be sufficient.

Buildings

There will be three main buildings on site, one main office, one staff office and one workshop. These will be discussed separately.

The main office will be 25x35m and will house the management staff, additional supporting staff and some engineers. The building should be solid but elegant at the same time.

The staff office includes dressing rooms for the mine operators and other staff that work on the mine site, showers and presentation room. Furthermore, the building houses a medical department and a housing department where temporary staff will be able to stay over the night. The building should be build robust.

The main workshop is a place where most of the machines and equipment will be repaired. The workshop will be built at a high standard and the working equipment will be very high end, in order to reduce downtime.

A small emergency response building will also be located at the mine site.

Other infrastructure

Fire risks are brought down to a minimum by installing a 24/7 staffed fire station is located on the mine site. The list of equipment that is stationed at the fire station is as follows:

- 2x E-one airport crash truck 8x8
- 1x Ford light and air rescue truck 4x4
- 1x Ford utility car
- 1x tanker, International 7400 SBA

Furthermore, the fire station includes one ambulance for quick response. A small medical team will also be stationed at the mine site to perform first aid.

Some security will also be required, however, a small security set will be sufficient. The security measurements include the following:

- Camera registration at all locations
- Two fulltime security officers with night shifts
- Infrared surveillance at the offices

A canteen will be provided by the mine company, but will be run by a contractor. The contractor will be charged with the operating costs of the canteen. In return, the contractor may keep a part of the profit. The capital investment of the canteen will be on the mine company.

Mobile toilets will be placed on the mine site, close to the operation.

12 telescoping light towers will be placed on the mine site for additional lighting. Each lighting tower is equipped with four 1 kW lights and a small generator. These lighting towers will only be turned on during the night.

While the mine is located in a remote area, it is important to provide some comfort and service to the employees. Therefore, good coffee machines will be provided by the company.

Table 6 Capex and operational costs for the infrastructure of a wet based mine

Type of cost	Capex (€)	Operational costs (€ per year)	Service life
Electricity			
260 kV power transmission line (6 km)	3.016.400	2.890.000	20
25 kV power transmission line (20 km)	3.100.621	1.144.000	20
260/25 kV transformer	61.390	22.620	10
25kV/400V transformer (8)	546.800	202.800	10
Main Service Unit	62.080	22.360	10
Volt- and Ampère meter (9)	38.700	14.852	10
Back-up generator 2.250 kW (10)[3]	4.709.000	3.342.000	15
Fuel and water			
Fuel tank (double wall) 56.000L (2)	68.400	25.300	15
Water tank 64.000L (2)	42.030	7.600	15
Buildings			
Main office	600.000		30
Mining staff office	600.000		30
Service/Utility shop	2.500.000		30

[3] The operating costs for the back-up generator only includes maintenance and repair. The other operating costs are included in the operating costs of the mining equipment.

Emergency response building	400.000		30
Temporary facilities			
Contractors housing (6x3m) (5) (3 months a year)		4.665	-
Other infrastructure			
Airport crash truck E-One 8x8 (2)	1.200.000	10.000	10
Tanker International 7400 SBA	200.000	5.000	10
Light and air rescue	60.000	5.000	10
Ambulance E-450	70.000	5.500	10
Utility command car	45.000	3.500	10
Surveillance system	250.000	15.000	10
Mobile toilets (15)	46.500	7.500	5
Telescoping light towers 4 kW[4] *(12)*	264.500	157.500	10
Canteen[5]	3.500.000	-	30
Coffee machine[6] *(5)*	12.950	12.000	5
Total	**21.394.000**	**7.897.000**	

[4] Operating costs are based on a 12 hours per day operation.
[5] The operating costs for the canteen are paid by the contractor who sells the food
[6] The operating costs include the costs for the coffee beans and the maintenance costs.

APPENDIX B – MATLAB CODE (V2.9)

IHC OPTIMIZATION TOOL FOR DREDGE MINE DESIGNS

Author: M.J.Bijmolt

```
% This script is made to find an optimal mine design and mining sequence
% for a dredge mine. It makes use of an adapted version of the greedy algorithm.

%The script consists of four parts:
%part1: User input to define the important parameters
%part2: Block model import from Surpac and adjustment of data
%Part3: Depth optimization
%Part4: Sequence optimization
%Part5: Data analysis and visualization

%Version 2 holds significant improvement in the sequence optimization
%engine. It determines the best next pond based on a potential series of
%ponds from that point.
    %Note by version 2.9: Incorporated discount rate for potential route
    %till user defined amount of years

    %Limitations: 1) square ponds
    %             2) one overall density

clear all
close all
```

PART 1: USER INPUT PARAMETERS

```
% Here, the user can set a number of parameters. The units are given for
% each parameter. It is important to carefully set these parameters, while
% the algorithm is sensitive. An extra note is added for the most
% important/sensitive parameters.

%Set power of the search engine. Use n=1 for a quick run, but please note
%that this will not give the optimzal sequence. However, it does give the optimal
%depth. For n=0, a true sequence optimization is done.
n=0;

%Define the effective production. The production will effect the time it
%takes to mine a column, therefore, influencing the cashlfow too.
production_eff=4500;    %m3/h

%Define the Opex and Capex. The Opex influences the ultimate depth, while
%the capex only influences the cashflow.
opex=113788950;         %per year
opex_processing= 48.03;  %USD/tonnes HMC(100%)
capex=242266471+450000000;        %in year 1

%Define the price. The price will influence the ultimate depth and also the
%sequence. The price greatly influences the ultimate depth.
price=482;      %USD/tonnes hm 482 is basecase
```

```
%Define discount rate as a fraction. This will influence the discounted
%cashflow.
discount_rate=0.1;

%Define number of blocks that are imported from Surpac.
block_number=31350;

%Define block size. It is very important that it is set right, while it
%influences all processes heavily.
block_length=350;   %x
block_width=350;    %y
block_height=2;     %z

%Define slope. With this slope, a pond adjustment factor is calculated to
%compensate for the slope walls.
slope=30; %degrees

%Define overall density
sg=1.9;

%Define block model name (Include id number!) and filename. (Surpac always
%adds an id number to the block_name)
block_name='ihc_dredge_optimizer1';
filename='ihc_optimizer.xlsx';

%Define maximum number of production years. When a maximum is not required,
%set define_maximum to zero. If it is required, set it to 1.
define_maximum=0; %0 for no maximum, 1 for a maximum.
max_prod_years=150; %in years

%Define number of groups for Surpac periods. Surpac only supports so many
%ranges for coloring attributes. By defining a group, it is easier to
%set the right colors.
group_number_export=20;

%Define maximum number of years to take into account the discount rate for
%the potential route
max_discount_route=6;
```

PART 2: IMPORT BLOCK MODEL AND DATA ADJUSTMENT

```
% Prerequisites for the block model:
    %1)all blocks above the surface must have zero value sg
    %2)all blocks must have the same size, no subblocking. (Version 2.7)
    %3)the block model must be exported to a csv file, with sg in the
    %  fourth column and hm content in the fifth column as a fraction
    %4)the csv file must then be converted to a xlsx file using excel

% Define sheet range for each column
block_X_sheet=strcat('A2:A',num2str(block_number+1));
block_Y_sheet=strcat('B2:B',num2str(block_number+1));
block_Z_sheet=strcat('C2:C',num2str(block_number+1));
block_sg_sheet=strcat('E2:E',num2str(block_number+1));
block_hm_sheet=strcat('F2:F',num2str(block_number+1));
```

```
% Import block model coordinates (csv) file from Surpac
block_X=xlsread(filename,block_name,block_X_sheet);
block_Y=xlsread(filename,block_name,block_Y_sheet);
block_Z=xlsread(filename,block_name,block_Z_sheet);

% Import Sg and values
block_sg=xlsread(filename,block_name,block_sg_sheet);
block_hm=xlsread(filename,block_name,block_hm_sheet);

%Create table
block_table=table(block_X,block_Y,block_Z,block_sg,block_hm);
```

ADJUST DATA

```
% Sort per block column. This way, the blocks are alligned per column.
block_table_srt=sortrows(block_table,{'block_X','block_Y','block_Z'},{'ascend','ascend','desc
end'});

% Convert table back to matrix for easy handling
block_matrix_srt=table2array(block_table_srt);

%Calculate volume per block
block_volume=block_length*block_width*block_height;

%Calculate mining cost per tonne. The limitation is that mining costs
%do not differ per depth or per area. (Version 2.7)
opex_mine_tonnes=opex/(production_eff*24*365.25*1.9);

%Calculate discount rate per week
discount_rate_week=(1-discount_rate)^(1/52);

%Calculate percentage of block that is not mined for slope stability
%purposes. It only takes into account the slopes on the side of the route,
%while the mining face is mined anyway to get to the next block.
pond_correction_frac=1-
(((block_height^2)/tand(slope))*block_width)/(block_width*block_height*block_length);

%Calculate production per week in tonnes
production_week=production_eff*24*7*sg;
```

PART 3: OPTIMIZE FOR DEPTH

```
% An ultimate depth is selected for each column. Ultimate is defined as the
% depth with the highest value. The value is calculated as the cumulative
% sum of revenues of the blocks above (that also have to be mined) minus
% the costs for mining all these blocks. When more than one ultimate depth
% exists, the shallowest one is chosen.

% Define first column
NewCol=1;
count_col=1;

% set loop for all blocks. Goal is to 1) define ends of the column and 2)
% sum the values to create a cumulative sum for each block.
```

```
for I_blocks=1:block_number

    if block_matrix_srt(I_blocks,2) == block_matrix_srt(NewCol,2)   %Determine whether block
is still in same column

        %Set exception for the case I_blocks=1, while for this case, the
        %block_value_sum does not yet exists.
        if I_blocks==1
            %Calculate cumulative value and tonnage

block_value_sum(I_blocks,1)=block_matrix_srt(I_blocks,4)*block_volume*pond_correction_frac*pr
ice*block_matrix_srt(I_blocks,5)-
block_matrix_srt(I_blocks,4)*(block_volume*pond_correction_frac)*opex_mine_tonnes-
block_matrix_srt(I_blocks,4)*(block_volume*pond_correction_frac)*block_matrix_srt(I_blocks,5)
*opex_processing;   %Calculate cumulative sum of values

block_tonnage_sum(I_blocks,1)=block_matrix_srt(I_blocks,4)*block_volume*pond_correction_frac;
%Calculate cumulative tonnage per column

        else
            %Calculate cumulative value and tonnage
            block_value_sum(I_blocks,1)=block_value_sum(I_blocks-
1,1)+block_matrix_srt(I_blocks,4)*block_volume*pond_correction_frac*price*block_matrix_srt(I_
blocks,5)-block_matrix_srt(I_blocks,4)*(block_volume*pond_correction_frac)*opex_mine_tonnes-
block_matrix_srt(I_blocks,4)*(block_volume*pond_correction_frac)*block_matrix_srt(I_blocks,5)
*opex_processing; %Calculate cumulative sum of values
            block_tonnage_sum(I_blocks,1)=block_tonnage_sum(I_blocks-
1,1)+block_matrix_srt(I_blocks,4)*block_volume*pond_correction_frac; %Calculate cumulative
tonnage per column
        end

    else

        %Define optimal depth as the maximum cumulative value for the
        %current column
        block_value_max=max(block_value_sum(NewCol:I_blocks-1,1));  %Determine the maximum
value

        %Now look how many ultimate depths exists by running a loop for the
        %column range

        %Define block number of optimal depth
        count_max=0;

        for I_ton=NewCol:I_blocks-1
            if block_value_sum(I_ton,1)==block_value_max
                count_max=count_max+1;
                block_number_max(count_max,1)=I_ton;

                %Count number of non-empty blocks above maximum depth
                blocks_above(count_max,1)=size(find(block_matrix_srt(NewCol:I_ton,4)>0),1);
            end
        end
```

```
        % Add new column to the pond table (x,y,z,tonnage,value, blocks above ultimate depth,
sg)
        column_matrix(count_col,:)= [block_matrix_srt(I_blocks-1,1)
block_matrix_srt(I_blocks-1,2) block_matrix_srt(block_number_max(1,1),3)
block_tonnage_sum(block_number_max(1,1),1) block_value_max blocks_above(1,1)
block_matrix_srt(block_number_max(1,1),4)];

        NewCol=I_blocks;            %Set first rownumber for new colum
        count_col=count_col+1;      %Define new column number

        % Begin new value and tonnage lines for the new column

block_value_sum(I_blocks,1)=block_matrix_srt(I_blocks,4)*block_volume*pond_correction_frac*pr
ice*block_matrix_srt(I_blocks,5)-
block_matrix_srt(I_blocks,4)*(block_volume*pond_correction_frac)*opex_mine_tonnes-
block_matrix_srt(I_blocks,4)*(block_volume*pond_correction_frac)*block_matrix_srt(I_blocks,5)
*opex_processing;                   %Define first value number for next cumulative value

block_tonnage_sum(I_blocks,1)=block_matrix_srt(I_blocks,4)*block_volume*pond_correction_frac;
%Define first tonnage number for next cumultivate tonnage

    end

end
```

PART 4: SEQUENCE OPTIMIZATION

Define the best route for the dredger in terms of NPV. This part starts with a section that evaluates the neighbours for each block, in descending order of value.

```
% For easy reading, the blocks are called
% potential friends with a certain amount of money. The goal is to get the
% friends with the most money. From a starting point, a number of
% neighbouring potential friends can be reached. Each neighbour has a
% number of neighbours too and so on. To get the most amount of money, one has to evaluate
all neighbours of neighbours before making one of HIS neighbours his friend. Once a friend,
it can never be reached (mined) again.
```

CREATING A NEIGHBOUR LIST

```
% In this section, a list of all neighbours in descending value order for
% all the column points is made

% First, define an empty matrix for these neighbours
best_neighbours=zeros(size(column_matrix,1),4);

for I_friendlist=1:size(column_matrix,1)

    %Create an empty neighbour list for the current column point
    neighbourlist=zeros(4,1);

    %Find neighbours in +X direction
    if isempty(find(column_matrix(:,1)==column_matrix(I_friendlist,1)+block_length &
column_matrix(:,2)==column_matrix(I_friendlist,2)))
```

```
        neighbourlist(1,1)=0;
    else

neighbourlist(1,1)=find(column_matrix(:,1)==column_matrix(I_friendlist,1)+block_length &
column_matrix(:,2)==column_matrix(I_friendlist,2));
    end

    %Find neighbours in -X direction
    if isempty(find(column_matrix(:,1)==column_matrix(I_friendlist,1)-block_length &
column_matrix(:,2)==column_matrix(I_friendlist,2)))
        neighbourlist(2,1)=0;
    else
        neighbourlist(2,1)=find(column_matrix(:,1)==column_matrix(I_friendlist,1)-
block_length & column_matrix(:,2)==column_matrix(I_friendlist,2));
    end

    %Find neighbours in +Y direction
    if isempty(find(column_matrix(:,2)==column_matrix(I_friendlist,2)+block_width &
column_matrix(:,1)==column_matrix(I_friendlist,1)))
        neighbourlist(3,1)=0;
    else
        neighbourlist(3,1)=find(column_matrix(:,2)==column_matrix(I_friendlist,2)+block_width
& column_matrix(:,1)==column_matrix(I_friendlist,1));
    end

    %Find neighbours in -Y direction
    if isempty(find(column_matrix(:,2)==column_matrix(I_friendlist,2)-block_width &
column_matrix(:,1)==column_matrix(I_friendlist,1)))
        neighbourlist(4,1)=0;
    else
        neighbourlist(4,1)=find(column_matrix(:,2)==column_matrix(I_friendlist,2)-block_width
& column_matrix(:,1)==column_matrix(I_friendlist,1));
    end

    %Remove zero lines, which are non-existing neighbours
    empty_rows=find(neighbourlist==0);
    neighbourlist=neighbourlist(~ismember(1:size(neighbourlist,1),empty_rows), :);

    %Make a table with descending maximum and indices in the second column
        %First, make the table
        for I_max=1:size(neighbourlist,1)
            matrix_for_table_max(I_max,1)=-column_matrix(neighbourlist(I_max,1),5);
        end
        %Make matrix
        matrix_max_friends=[matrix_for_table_max neighbourlist];
        %Sort matrix in descending order of value
        matrix_max_friends_srt=sortrows(matrix_max_friends,1);

    %Make a neighbourlist with descending good neighbours
        %Pull last neighbour indices from sorted table
        friends_descending_order=matrix_max_friends_srt(:,2);
        %Add to friendlist

best_neighbours(I_friendlist,1:size(friends_descending_order,1))=transpose(friends_descending
_order);

    %Clear values
```

```
        friends_descending_order=0;
        matrix_max_friends=0;
        matrix_for_table_max=0;
end
```

START MASTER LOOP FOR STARTING POINTS

```
% In this section, the actual sequence optimization is done. First, a loop
% for all the column points is initiated. When a starting point has zero tonnage, it is
skipped.
% Then, a loop is initiated for all
% the possible points (or friends). The current block is then evaluated in
% terms of cashflow and mine time. From that current block, the
% neighbouring blocks are evaluated. For each neighbouring block, a
% potential route is created for the entire (remaining) grid. If a block is
% already mined, the second best block is selected. For each block, it is
% investigated whether its neighbours are already mined. If this is the case,
% the particular block is not considered and the next best choice is evaluated.
% The values for the potential route are then summed and compared. The
% neighbour with the highest 'potential route value' is chosen as the next
% block to be mined.

%Create empty matrix for discounted cashflow
discounted_cashflow_sequence=zeros(max_prod_years*52,size(column_matrix,1));
current_friends=zeros(size(column_matrix,1),size(column_matrix,1));

% Set the power of the engine if n=0
if n==0
    n=size(column_matrix,1); %Now, it considers all the blocks in the grid.
end

% Initiate loop for all starting points
for I_seq=1:size(column_matrix,1)

    %Set new start point
    start_point=I_seq;

    %Reset discounted cashflow and number of weeks
    number_weeks_disco=0;
    discounted_cashflow_week=0;

    %Define current friend list
    current_friends(1,I_seq)=I_seq; %add starting point as a member of the current friend
list

    %Check if the first column has zero tonnage, if this is the case,
    %consider a different starting point
    if column_matrix(start_point,4)>0

    %Start loop for number of blocks in one sequence
    for I_route=1:size(column_matrix,1)

        %Calculate discounted cashflow per week for the given column
            %Define number of working weeks for given column and value per
            %week
```

```
column_production_weeks=ceil(column_matrix(start_point,4)/(production_eff*24*7*sg));
                if column_production_weeks==0
                    value_perweek=column_matrix(start_point,5);
                else
                    value_perweek=column_matrix(start_point,5)/column_production_weeks;
                end

            %Define number of weeks production so far
                number_weeks_disco=number_weeks_disco+column_production_weeks;

            %Define discounted cashflow per week
                %first, empty discounted cash flow week and counting
                %mechanism
                discounted_cashflow_week=0;
                count_discoweek=0;

                if I_route==1
                    for I_discount=1:column_production_weeks

discounted_cashflow_week(I_discount,1)=value_perweek*(discount_rate_week^I_discount);
                    end
                else
                    for I_discount=(number_weeks_disco-
column_production_weeks):number_weeks_disco
                            count_discoweek=count_discoweek+1;

discounted_cashflow_week(count_discoweek,1)=value_perweek*(discount_rate_week^(I_discount));
                    end
                end

                %Substract capex from week 1 once
                if I_route==1;
                    discounted_cashflow_week(1,1)=discounted_cashflow_week(1,1)-capex;
                end

            %Add weekly discounted cashflow to discounted cashflow for
            %current sequence
                if I_route==1
                    if size(discounted_cashflow_week,1)==1
                        discounted_cashflow_sequence(1,I_seq)=discounted_cashflow_week;
                    else

discounted_cashflow_sequence(1:size(discounted_cashflow_week,1),I_seq)=discounted_cashflow_we
ek;
                    end
                        number_cashweeks=size(discounted_cashflow_week,1);
                else

discounted_cashflow_sequence(number_cashweeks+1:number_cashweeks+column_production_weeks+1,I_
seq)=discounted_cashflow_week(:,1);

%discounted_cashflow_sequence(1:size(vertcat(discounted_cashflow_sequence(1:number_cashweeks,
I_seq),discounted_cashflow_week),1),I_seq)=vertcat(discounted_cashflow_sequence(1:number_cash
weeks,I_seq),discounted_cashflow_week);

                        number_cashweeks=number_cashweeks+size(discounted_cashflow_week,1);
```

```
                end

            %Stop searching for ponds if maximum years of production is reached
            if number_cashweeks>=max_prod_years*52
                %disp('production is achieved prematurely')
                break
            end

            %Search the best new neighbour based on his best neighbours (all steps ahead
version)

            %Start for loop for number of neighbouring columns
            for I_adjacent=1:nnz(best_neighbours(start_point,:))

                %Set potential_weeks_production to current number of
                %production weeks
                potential_weeks_production=number_cashweeks;
                %Get neighbour
                get_neighbour(I_adjacent,1)=best_neighbours(start_point,I_adjacent);

                %check whether neighbour is already a member of the
                %friendlist, which are stored in the 'current friends' matrix
                if isempty(find(get_neighbour(I_adjacent,1)==current_friends(:,I_seq)))==1

                %If not the case, then proceed finding route of best
                %friends and summing the values for this route
                    %Define weeks production for potential new column
                    production_potential=column_matrix(get_neighbour(I_adjacent,1),4);
                    %Add potential week to summation if total current week
                    %number is less than 6*52 weeks
                    if potential_weeks_production<max_discount_route*52

potential_weeks_production=potential_weeks_production+production_potential;
                    else
                        potential_weeks_production=max_discount_route*52;
                    end
                    %Begin summing values

values_route_sum(I_adjacent,1)=column_matrix(get_neighbour(I_adjacent,1),5)*(discount_rate_we
ek^potential_weeks_production);

                    %Begin an imaginary friend list, in which the
                    %friends in the imaginary search loop are saved to
                    %exclude member that are sort of already in the
                    %team.
                    imaginary_friends_check=zeros(n,1);

                    %Begin loop for the search radius.
                    for I_engine=1:n
                        if I_engine==1

                            %Reset loop parameters
                            check_number_friends=0; %reset loop count for actual holdup
                            break_premat=0; %reset loop count for imaginary holdup

                            %Find all neighbours of neighbours
                            for
```

```
I_non_mined=1:nnz(best_neighbours(get_neighbour(I_adjacent,1),:))

                                %When the end of the imaginary
                                %route is reached, end the search
                                if
check_number_friends==nnz(best_neighbours(get_neighbour(I_adjacent,1),:)) ||
break_premat==nnz(best_neighbours(get_neighbour(I_adjacent,1),:))
                                        break
                                else

                                    %Check if neighbour of neighbour is
                                    %already a member of the friendlist
                                    if
isempty(find(best_neighbours(get_neighbour(I_adjacent,1),I_non_mined)==current_friends(:,I_se
q) ))==1

                                        %Check if neighbour of neighbour
                                        %has neighbour that are not mined
                                        %yet
                                        break_out=0; %Reset break loop

                                        for
I_out=1:nnz(best_neighbours(best_neighbours(get_neighbour(I_adjacent,1),I_non_mined),:))
                                            if
isempty(find(best_neighbours(best_neighbours(get_neighbour(I_adjacent,1),I_non_mined),I_out)=
=current_friends(:,I_seq)))==1 &&
isempty(find(best_neighbours(best_neighbours(get_neighbour(I_adjacent,1),I_non_mined),I_out)=
=imaginary_friends_check(:,1)))==1
                                            else
                                                break_out=break_out+1;
                                            end
                                        end

                                        %If neighbour of neighbour has
                                        %neighbours that are not mined yet,
                                        %add this neighbour to the
                                        %imaginary route.
                                        if
break_out<nnz(best_neighbours(best_neighbours(get_neighbour(I_adjacent,1),I_non_mined),:))

friends_of_friends(I_engine,1)=best_neighbours(get_neighbour(I_adjacent,1),I_non_mined);

                                            %Add value of this imaginary
                                            %point to cumulative sum
                                                %First, add number of weeks
                                                %production to potential list
                                                if
potential_weeks_production<max_discount_route*52

potential_weeks_production=potential_weeks_production+column_matrix(friends_of_friends(I_engi
ne,1),4)/production_week;
                                                else

potential_weeks_production=max_discount_route*52;
                                                end

values_route_sum(I_adjacent,1)=values_route_sum(I_adjacent,1)+(column_matrix(friends_of_frien
```

```
ds(I_engine,1),5)*(discount_rate_week^(potential_weeks_production)));

                                        %Add imaginary friend to
                                        %imaginary list

imaginary_friends_check(I_engine,1)=friends_of_friends(I_engine,1);
                                        break

                                    %If the neighbour of neighbour has
                                    %only heighbours that are already
                                    %mined, count this neighbour and
                                    %continue looking for a second
                                    %choice
                                    else
                                        break_premat=break_premat+1;
                                    end

                                %If the neighbour of neighbour is already mined, continue
looking for a second choice
                                else
                                    check_number_friends=check_number_friends+1;
                                end

                            end
                        end

                    %Break loop if it has hit a real dead end, with no more ways out
                        if
check_number_friends==nnz(best_neighbours(get_neighbour(I_adjacent,1),:)) ||
break_premat==nnz(best_neighbours(get_neighbour(I_adjacent,1),:))
                            break
                        end

                     elseif I_engine>1

                            %Reset loop parameters
                            check_number_friends=0; %reset loop count for actual holdup
                            break_premat=0; %reset loop count for imaginary holdup
                            check_break_premat=0; %Reset loop count for check imaginary
holdup

                            %check whether neighbour is already a member of the
                            %friendlist, which are stored in the 'current friends' matrix
                            for
I_non_mined=1:nnz(best_neighbours(friends_of_friends(I_engine-1,1),:))

                                %When the end of the imaginary
                                %route is reached, end the search
                                if
check_number_friends==nnz(best_neighbours(friends_of_friends(I_engine-1,1),:)) ||
break_premat==nnz(best_neighbours(get_neighbour(I_adjacent,1),:))
                                        break

                                else
                                    if
isempty(find(best_neighbours(friends_of_friends(I_engine-
1,1),I_non_mined)==current_friends(:,I_seq)))==1  &&
```

```
isempty(find(best_neighbours(friends_of_friends(I_engine-1,1),I_non_mined)
==imaginary_friends_check(:,1)))==1

                                        %Check if neighbour of neighbour
                                        %has neighbours that are not mined
                                        %yet
                                        break_out=0; %Reset break loop
                                        check_break_premat=check_break_premat+1;

                                        for
I_out=1:nnz(best_neighbours(best_neighbours(friends_of_friends(I_engine-1,1),I_non_mined),:))
                                            if
isempty(find(best_neighbours(best_neighbours(friends_of_friends(I_engine-
1,1),I_non_mined),I_out)==current_friends(:,I_seq)))==1 &&
isempty(find(best_neighbours(best_neighbours(friends_of_friends(I_engine-
1,1),I_non_mined),I_out)==imaginary_friends_check(:,1)))==1
                                            else
                                                break_out=break_out+1;
                                            end
                                        end

                                        %If neighbour of neighbour has
                                        %neighbours that are not mined yet,
                                        %add this neighbour to the
                                        %imaginary route.
                                        if
break_out<nnz(best_neighbours(best_neighbours(friends_of_friends(I_engine-
1,1),I_non_mined),:))

friends_of_friends(I_engine,1)=best_neighbours(friends_of_friends(I_engine-1,1),I_non_mined);

                                            %Add value of this imaginary
                                            %point to cumulative sum
                                                %First, add potential
                                                %weeks of production to
                                                %list
                                                if
potential_weeks_production<max_discount_route*52

potential_weeks_production=potential_weeks_production+column_matrix(friends_of_friends(I_engi
ne,1),4)/production_week;
                                                else

potential_weeks_production=max_discount_route*52;
                                                end

values_route_sum(I_adjacent,1)=values_route_sum(I_adjacent,1)+(column_matrix(friends_of_frien
ds(I_engine,1),5)*(discount_rate_week^(potential_weeks_production)));

                                            %Add imaginary friend to
                                            %imaginary list

imaginary_friends_check(I_engine,1)=friends_of_friends(I_engine,1);
                                            break

                                        %If the neighbour of neighbour has
                                        %only heighbours that are already
```

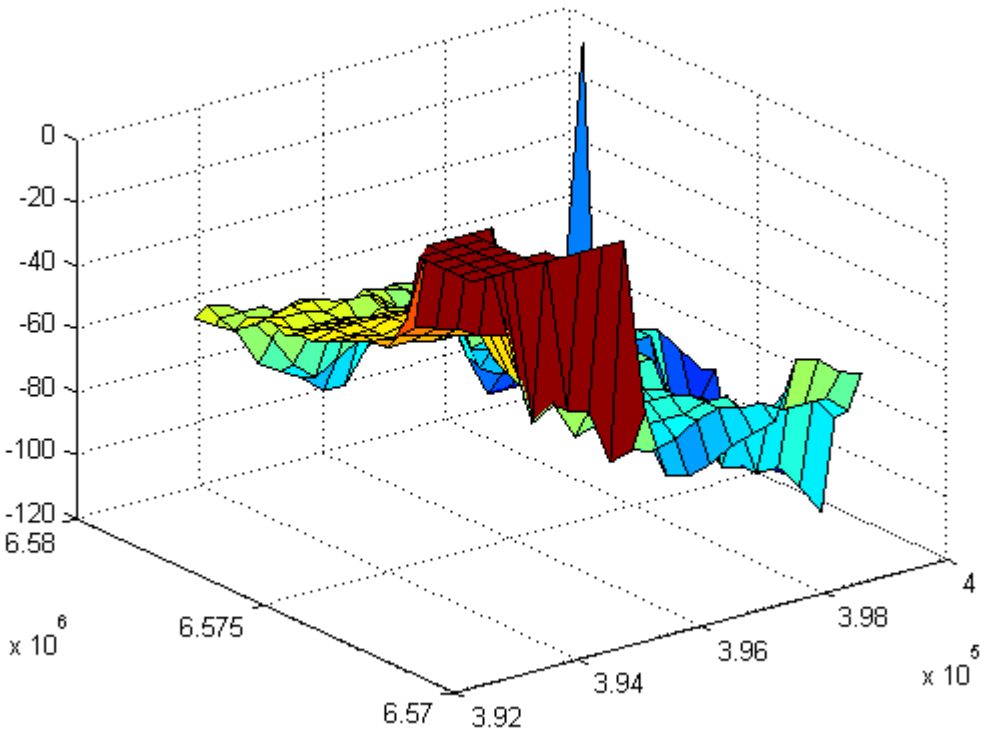
```
                                        %mined, count this neighbour and
                                        %continue looking for a second
                                        %choice
                                        else
                                            break_premat=break_premat+1;
                                        end

                                    %If the neighbour of neighbour is already mined,
continue looking for a second choice
                                    else
                                        check_number_friends=check_number_friends+1;
                                    end

                                end
                            end

                            %Break loop if it has hit a real dead end
                            if
check_number_friends==nnz(best_neighbours(friends_of_friends(I_engine-1,1),:)) ||
break_premat==check_break_premat
                                break
                            end

                        end
                    end

                else

                    %neighbour is already a member of the friend
                    %list. Therefore, set its value to -99. All the -99
                    %entries will be deleted
                    values_route_sum(I_adjacent,1)=-99;
                end

            end

            %Check if it is the end of the route. The end of the route is
            %achieved when all values are -99, which mean that all neighbours
            %are already mined
            if all(values_route_sum==-99)==1

                %End of route. First, empty matrices:
                get_neighbour=0;
                compare_routes_matrix=0;
                compare_routes_matrix_srt=0;
                values_route_sum=0;

                %End route
                break
            else

            %Compare the values of the route and choose optimal route

                %Create number of indices for matrix. The values are
                %multiplied by -1 to sort in a descending order. The values
                %are not used further
                compare_routes_matrix=[values_route_sum.*-1
```

```
transpose(linspace(1,size(values_route_sum,1),size(values_route_sum,1)))];
                %Remove neighbouring blocks that are already mined

compare_routes_matrix=compare_routes_matrix(~ismember(1:size(compare_routes_matrix,1),
find(compare_routes_matrix(:,1)==99)), :);
                %Sort the matrix
                compare_routes_matrix_srt=sortrows(compare_routes_matrix,1);

            %Get the best neighbouring block, set it as the new start point and
            %add it to the list of current friends
                start_point=get_neighbour(compare_routes_matrix_srt(1,2),1);

current_friends(I_route+1,I_seq)=get_neighbour(compare_routes_matrix_srt(1,2),1);

            %Empty matrices
                get_neighbour=0;
                compare_routes_matrix=0;
                compare_routes_matrix_srt=0;
                values_route_sum=0;

            end
    end

    %Define NPV for current route
    NPV(I_seq,1)=sum(discounted_cashflow_sequence(:,I_seq));

    else
        %If the first column has zero tonnage, then consider a different
        %starting point and set the NPV for this route to 0.
        NPV(I_seq,1)=0;
    end


end
```

PART 4: DATA ANALYSIS

```
%Visualize ultimate depth. This is done by creating a meshgrid for X, Y and
%Z.

    %Search the column matrix to find the unique x and y coordinates. At
    %all coordinates, the Z value is saved.

    %Set initial parameters
    count_surfY=0;
    count_surfX=1;
    NewX=1;
    Xmesh(1,1)=column_matrix(1,1);
    for I_surf=1:size(column_matrix,1)
        if column_matrix(I_surf,1)==column_matrix(NewX,1)
            count_surfY=count_surfY+1;
            Z(count_surfY,count_surfX)=(column_matrix(I_surf,6)+1)*-block_height;
            sizeX(count_surfX,1)=count_surfY;
        else
            count_surfY=1;
```

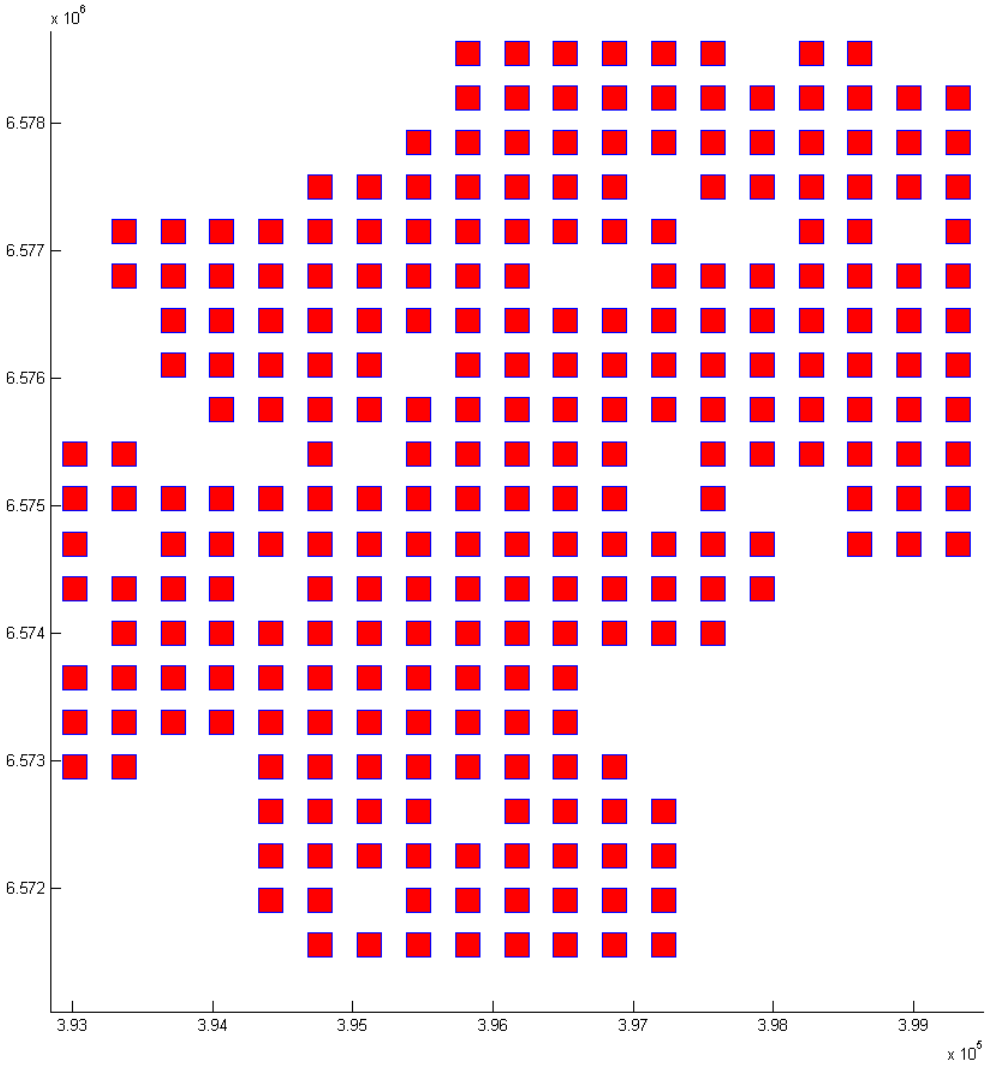
```
            count_surfX=count_surfX+1;
            NewX=I_surf;
            Xmesh(count_surfX,1)=column_matrix(NewX,1);
            Z(count_surfY,count_surfX)=(column_matrix(I_surf,6)+1)*-block_height;
        end
    end

    %Create meshgrid for X and Y (Z is already a meshgrid)
    [X, Y]=meshgrid(Xmesh,column_matrix(1:sizeX(1,1),2));


    %Draw a surface plot
    surf(X,Y,Z)
```



DISPLAY ROUTE ON XY SCATTERPLOT

```
% The route is displayed in a scatter plot with square markers, resembling
% the ponds. Furthermore, a movie is captured.

% Define starting point (highest NPV)
start_point_scatter=find(NPV==max(NPV));

% Define file name
movie_filename=strcat(block_name,'_',num2str(price),'.avi');

% Define axis properties
fig=figure;
FigHandle = figure('Position', [100, 100, 1000, 1000]);
```

```
axis([min(column_matrix(:,1))-0.5*block_length max(column_matrix(:,1))+0.5*block_length
min(column_matrix(:,2))-0.5*block_width max(column_matrix(:,2))+0.5*block_width])
set(gca,'nextplot','replacechildren');
set(gcf,'Renderer','zbuffer');

%Start movie
aviobj=VideoWriter(movie_filename);
aviobj.FrameRate=1000/250; %a total of 1000 frames in 250 seconds
open(aviobj);

%Start loop for all the blocks that are mined
for I_display=1:nnz(current_friends(:,start_point_scatter))


scatter(column_matrix(current_friends(I_display,start_point_scatter),1),column_matrix(current
_friends(I_display,start_point_scatter),2),block_length,'s','fill', 'MarkerFaceColor',
'r','MarkerEdgeColor','b' )
    writeVideo(aviobj,getframe);

    %Display all mined blocks
    hold on

end
close(aviobj);
```



EXPORTING DATA

Export data to Surpac in a csv file. First, define for each column a group number and add it to a new matrix. Then check how many blocks are above the lowest block for that column and add that many blocks to the matrix, with a adjusted depth. Finally, export the block model to a csv file.

```
% Note for exporting to Surpac: the csv file must imported in the old block model. The block
size has to be set to the original block size. A new attribute for the block model should be
made for the group number (with background value set to 0). The other attributes (like hm
content) are already present in the original block model and therefore, do not need to be
added again.

% Set how much blocks are in each group, rounded above
column_per_group=ceil(nnz(current_friends(:,find(max(NPV)==NPV)))/group_number_export);

% Set first group member
    group_member=1;
```

```
% Start a loop for all the blocks visited
for I_sequentie=1:nnz(current_friends(:,find(max(NPV)==NPV(:,1),1,'first')))

    %Stop adding lines if the route is ended
    if current_friends(I_sequentie,find(max(NPV)==NPV(:,1),1,'first')) ==0
        break
    end

    %Define member of the group
    if I_sequentie/column_per_group>group_member
        group_member=group_member+1;
    end

    %Add line to sequence matrix

sequence_matrix(I_sequentie,:)=horzcat(column_matrix(current_friends(I_sequentie,find(max(NPV
)==NPV(:,1),1,'last')),:),I_sequentie,group_member);
end

% Create blocks above that are mined too
    %Set number of extra lines to 0
    count_extra_lines=1;

for I_output=1:size(sequence_matrix,1)
    % Define a loop for each block above
        for I_extra_lines=1:sequence_matrix(I_output,6)
            %Save the blocks in 'extra_output_matrix
            extra_output_matrix(count_extra_lines,:)=sequence_matrix(I_output,:);
            %Adjust depth per block

extra_output_matrix(count_extra_lines,3)=extra_output_matrix(count_extra_lines,3)+2*I_extra_l
ines;
            %Set new 'count extra lines'
            count_extra_lines=count_extra_lines+1;
        end
end

    %Add extra lines to sequence matrix
    sequence_matrix=vertcat(sequence_matrix,extra_output_matrix);

%Write to csv file

dlmwrite('ihc_optimize_export.csv', sequence_matrix,'delimiter',';','precision',10);
```

Published with MATLAB® R2013b

APPENDIX C – OPTIMIZATION ENGINE

The algorithm works by defining a potential route from each neighbor and calculating a NPV based on that potential route. An example is given in Figure 26. The brown blocks are already mined, the blue block is the current pond and the other blocks are yet to be mined. The numbers in the blocks correspond to the values of the columns. From the blue block, the red block or the green block can be mined. The algorithm will select a potential route, by determining each block's highest neighbor, from a given start point. For the red neighbor, the potential route is displayed using red arrows. In this example, a discount rate of 10% is used per column to evaluate the route. The NPV of the red neighbor is then determined at 25.8, while the green neighbor has a NPV of 25.1. Therefore, the green neighbor is the best choice.

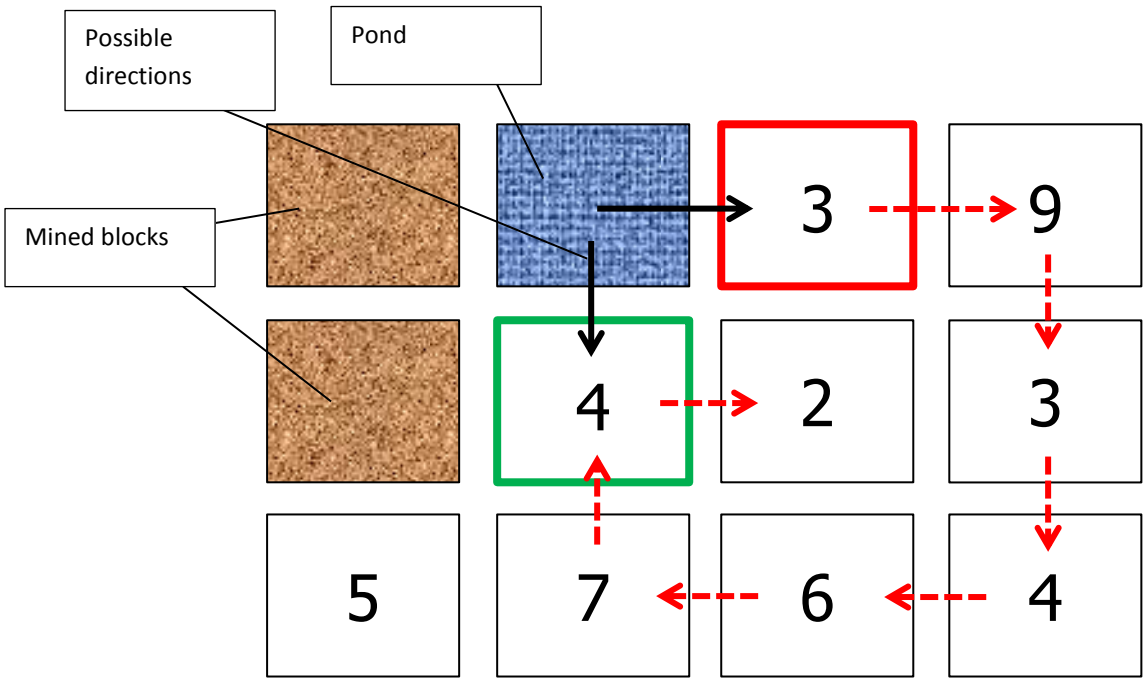


Figure 26 Example of the optimization engine