Utrecht University

Mathematics for Industry

Stability and Optimisation of the Homotopy method

Deltares

Authors
T. van den Brink,
R. Mak,
N. Strachan,
A. Tahiri,
M. Versluis,
M. Vrijhof,
K. Xiao

Supervisor
Dr. I. Kryven

Problem Poser
A. Mitchell, MSc,
Dr. B. Becker
(Deltares)

April 18, 2024

Utrecht University

Executive Summary

The problem posed by Deltares was to study the Homotopy method applied to the *Hoogheemschap van Rijnland* water system. The goal of the project was twofold.
The first aim was to decrease the runtime of the model. To this end, we introduced two methods. The SmartSeed method uses the computation of the previous hour as a seed for the computation of the next hour. This method allows us to skip two computation steps. When tested, the method decreased runtimes by 60 %. The second method we introduced is a semi-linearised model. It only solves the complete model for the first 24 hours and uses a linearisation for predictions further in the future. We expected quite significant runtime improvements but were not able to demonstrate this.

The second goal of the project was to investigate the theoretical foundations of current methods with regard to stability. The current methods are based on an article by J. Baayen that deals with slightly simpler systems than those needed at Deltares: in the article, water levels are optimized for only one objective. Since water management often comes with several, possibly conflicting, objectives, we tried to shape the ideas of said article into the context of multi-objective optimization. The major task here was to define the different notions of stability in the context of multi-objective optimization and see which results from the article could still be carried over to the new setup. Most results carry over, but finding a global optimum requires further research.

Contents

1 Introduction

The complexity of water resource management is increasing with the effects of climate change. Good water management ensures the sustainable development of human societies and natural ecosystems. Water management involves variable demands, limited resources, and complex environmental factors. Reservoir operations usually require consideration of multiple objectives, such as flood control, drinking water supply, irrigation, hydropower generation, and many more. These objectives may conflict with each other, making harmonization difficult at times. In this context, mathematical optimization methods offer highly valuable solutions. By applying optimization techniques, we can build models to simulate water supply and demand. This approach can help operators make the best decisions under complex and changing conditions, optimize the allocation and use of water resources, reduce waste, and improve energy efficiency.

If we are only interested in water flows and volumes, the optimization problem can be expressed as linear equations, and the solution of these equations is straightforward. Gradient optimization methods can solve these linear optimization problems with global optimum [1]. However, if we want to focus on the relationships between power, flow, and elevation for generating turbines, it can be a challenge to optimize over a non-linear system. Problems can be more computationally expensive compared to the linear part. A common solution to this problem is to transform the non-linear equations into a linear equation, by a method called Homotopy. The problem posed by Deltares was to investigate the Homotopy method applied to the water management of the Hoogheemraadschap van Rijnland, which covers a system of canals in which the water levels have to be managed.

1.1 General description of the current model

The dynamics of this system of canals can be described by the Saint-Venant equations, also known as the shallow water equations:

$$F = \frac{\partial Q}{\partial t} + \frac{\partial}{\partial x}\frac{Q^2}{A} + gA\frac{\partial H}{\partial x} + g\frac{Q|Q|}{ARC^2} = 0, \tag{1}$$

and the mass balance (or continuity) equation

$$\frac{\partial Q}{\partial x} + \frac{\partial A}{\partial t} = 0, \tag{2}$$

with time t and longitudinal coordinate x. These equations describe the fluid dynamics in situations where the horizontal length scale is greater than the scale of the water depth [2]. In the equations, the water depth is denoted by the variable H and the discharge by Q. The parameter A denotes the cross-section, $R = A/P$ represents the hydraulic radius, P is the wetter perimeter, C is the Chézy friction coefficient, and g is the gravitational constant.

The optimal solutions for water level H and discharge Q should be solutions to this equation, so the Saint-Venant equations act as a constraint in every optimization step. Since that constraint is inherently non-linear, two main problems have to be tackled. Firstly, there is no guarantee that the canonical optimization methods converge to a global optimum when constraints are nonlinear. Secondly, solving the nonlinear Saint-Venant equation and dealing with nonlinear constraints is computationally expensive, especially compared to the linearised system.

To circumvent those problems, Deltares introduced the so-called Homotopy method in the context of water management. In this method, the objectives are first optimized subject to a linearisation of the constraint posed by the Saint-Venant equation.
For the linearised model, we can find a local optimum. Next, the constraint is continuously deformed into the original nonlinear constraint, using a linear homotopy:

$$G(\theta) = (1-\theta)\tilde{F} + \theta F \qquad \text{for } \theta \in [0,1]. \tag{3}$$

Note that for $\theta = 0$, $G(0)$ is the linearised constraint, and for $\theta = 1$, $G(1)$ is the original, nonlinear constraint. The idea is to trace the optimum for the linearised constraint to that of the nonlinear constraint, by stepwise increasing the Homotopy parameter θ from 0 to 1. At every step, the result of the previous θ is used as seed for the next θ.

All those steps take time, and in the Homotopy method, the optimization problem is solved $n = \frac{1+\Delta\theta}{\Delta\theta}$ times for every run. The advantage of this approach is that in principle we should converge to the optimal solution. However, a large disadvantage is that the runtime of the model increases significantly. To mitigate this problem, the step size of θ in the Homotopy method is $\Delta\theta = 0.5$. Still, Deltares finds that the runtime for the linear step, $\theta = 0$, is approximately 2 minutes, whereas runtimes for the nonlinear steps $\theta = 0.5$ and $\theta = 1$ are approximately 7 and 9 minutes, respectively[1]. This adds up to a total runtime of approximately 18 minutes. Runtime introduces a gap between measurements and prediction, and since the model is run with new measurement data every hour, a gap of 18 minutes is significant and undesirable.

1.2 Structure of the report

In our report, we investigate the Homotopy method applied by Deltares and consider adaptations of the method to decrease the runtime. The Homotopy method is known to derive path-stable solutions for single-objective optimization, so we will delve into how this method can be used in multi-objective optimization. Furthermore, we will investigate how this method can derive robust solutions. The Homotopy path gradually introduces variability in the inlet flow and uncertainty in the demand. By tracing this path, it is possible to find water release strategies that robustly maintain safe levels under changing conditions, avoiding overflows and low water levels.

This report can be divided into roughly two parts. The first part is the theoretical part, which aims to investigate the theoretical underpinnings of multi-objective optimization when combined with the Homotopy method. In particular, we will discuss what stability means in the context of multi-objective optimization and which concepts from the original analysis can be carried over. The programming part explores the more practical side of multi-objective optimization. In particular, we investigate methods on how to improve the current optimization with respect to runtime. We developed the SmartSeed method, which uses the results of the previous run as a seed, or guess, for the next run. The SmartSeed method allows us to skip certain steps in the Homotopy method, reducing the runtime while not compromising on the obtained results. We also study the possibility of linearising the last half of the time interval of the problem to try and reduce the runtime.

[1]Note that the difference between the latter two, $\theta = 0.5$ and $\theta = 1$, is most likely caused by differences in post-processing.

2 Mathematical Description of the current methods

Before we can investigate the mathematical foundations of the current optimization methods used at Deltares, we will first need a mathematical description. The current optimization methods combine goal programming and the Homotopy method. In this section, we will describe and explain those concepts in a mathematical context.
The Homotopy method, as it is used at Deltares, is based on the paper in [2]. The authors show that a solution from a convex problem can be deformed to a solution of a non-convex problem so that a global optimum is obtained, provided that the constraints are given by some discretized version of the Saint-Venant equations. However, said paper is only concerned with optimization problems with a single objective function. In the last part of this section, we will see how we can adapt concepts from the original paper to the context of goal programming.

2.1 A Mathematical Description of Goal Programming

In the context of water management, we often have conflicting goals that we aim to optimize. Sometimes these goals clearly differ in priority. For example, economic benefits should not be prioritized over safe water level ranges. The idea behind goal programming is to solve multiple optimization problems in order of importance whilst using the result from the previous optimization problem for the next problem. These goals can be represented as functions $f_i : \mathbb{R}^n \to \mathbb{R}$, where the i corresponds to the objective and $\mathbb{R}^n$ is the n-dimensional search space in which we seek solutions. We will discuss the options of *Pareto weighting* and *goal programming*, but focus on the latter.

2.1.1 Pareto weighting

An immediate thought that would come to mind when considering multiple goals to optimize, is to give each of the objective functions a weight according to the priority of the goal. That is, when considering multiple objectives, we do not optimize over a single function, but instead describe each goal separately and give each goal its own objective function. The optimization problem will take the form

$$\begin{aligned} \min_{x\in\mathbb{R}^n} \quad & \sum_{i=1}^{m} w_i f_i(x) \\ \text{subject to} \quad & c_i(x) = 0, \qquad i = 1, \dots, m_1 \\ & g_i(x) \leq 0, \qquad i = 0, \dots, m_2, \end{aligned}$$

where $c_i(x)$ and $g_i(x)$ denote the equality and inequality constraints known prior to optimization, and the parameter $w_i \geq 0$ denotes the weight given to objective function f_i.

However, choosing the weights can be somewhat arbitrary and subjective. Ideally, one would consider all possible weightings that give all possible desirable solutions. Those desirable solutions can be described by so-called *efficient solutions*. Efficient solutions are defined in such a way that no other solution performs better on all goals; no solution strictly outperforms the efficient solution.

Definition 2.1 ([3, Efficient Solution])**.** A solution x that satisfies the constraint is called efficient if there is no other point $\bar{x}$ that satisfies

- the constraint,
- the strict inequality $f_i(\bar{x}) < f_i(x)$ for at least one i, **and**
- the inequality $f_i(\bar{x}) \leq f_i(x)$ for all constraints.

If the constraint set is convex and the optimization function is convex, then efficient solutions can be found by varying the parameters whilst going over $W^1 := \{w \in \mathbb{R}^m : \sum_{i=1}^{m} w_i = 1, w_i \geq 0\}$ [3]. According to Deltares, convex relaxations of the problem do not seem to reflect reality accurately enough, but perhaps one can use this method to find good enough approximations by a sufficiently

good convex relaxation.

The main problem with Pareto weighting is that it may still produce undesirable solutions. It may still lean more into an economic benefit instead of the more important safety. To fix this, we can optimize our goals in a specified order. This is the idea of goal programming.

2.1.2 Goal programming

Instead of trying to have one problem, we consider an ordered set of problems and state that the solution to the i$^{\text{th}}$ problem may not deviate from the best solution to a previous problem. This way, we preserve the properties from our previous solutions. Suppose that we find an approximate solution x_i^* for the problem $f_i^* := f(x_i^*)$ where $i < j$, then the next problem is defined as

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & f_j(x) \\ \text{s.t.} \quad & c_i(x) = 0 \ (i = 1, \dots, m_1) \\ & x_i \geq 0 \ \ (i = 1, \dots, m) \\ & f_i(x) \leq f_i^* \ \ (i = 1, \dots, j-1). \end{aligned}$$

The idea is that we already cover the most important goals and thus incentivize an order of importance in the optimization.

In practice, there are goals with equal priority. For these goals, a weighting is sufficient. In other words, we only perform goal programming when the priorities of the goals differ significantly. More precisely, each goal f_i is given a priority $p(i)$. Goals with the same priority p are then combined as

$$f(p) = \sum_{\substack{i=1 \\ p(i)=p}}^{m} w_i f_i,$$

where $w_i \geq 0$. We then perform goal programming with the goals $(f(p))_p$. This is precisely the description of goal programming as found in the documentation of RTC-tools.

2.2 Homotopy method

The Homotopy method is a special case of parametric programming. The idea is to continuously deform a solution of a convex relaxation of the original optimization problem to a good solution of the non-convex program. That is, for $\theta \in [0, 1]$, we look at

$$\theta \tilde{F} + (1 - \theta) F$$

where F is a relaxation of the problem and $\tilde{F}$ is the full non-convex problem. The notion of starting with a convex program is captured in the notion of *zero-convexity*.

Definition 2.2. A parametric optimization problem is called *zero-convex* if the problem is convex for $\theta = 0$.

We slightly deviate from the original formulation where we require linear equality constraints and a convex objective. The reason for this slight change is that the inequality constraint we add with goal programming perfectly fits into the context of convex optimization.
The main motivation behind the Homotopy method is the implicit function theorem. In general, suppose that we have a solution to the equation

$$F(x, \theta) = 0.$$

and assume that F is sufficiently smooth. If (x^*, θ^*) is a solution of the above system, then the implicit function theorem asserts that we are able to find a continuation in a small neighborhood of this point as long as $\frac{\partial F}{\partial x}(x^*, \theta^*)$ is non-degenerate.

2.2.1 Barrier formulation and path stability for single objective optimization

The idea is that we do not trace the solution of the original problem immediately, but instead look at the related barrier formulation. Let us see how this was done in the original problem and how we may adjust this idea to the case of multiobjective optimization. Recall that we started out with the optimization problem.

The original paper by [2] considers an optimization problem of the form:

$$\begin{aligned} \min_{x\in\mathbb{R}^n} \quad & f(x,\theta) \\ \text{s.t.} \quad & c_i(x,\theta) = 0, i = 1,\dots,m_1 \\ & x_i \geq 0, i = 1,\dots,m_2. \end{aligned} \tag{4}$$

Note that this optimization problem now relies on a parameter $\theta \in [0,1]$, which represents the homotopy change as we vary from the linear to the non-linear problem. This problem can be translated into a so-called *barrier formulation* with parameter $\mu > 0$:

$$\begin{aligned} \min_{x\in\mathbb{R}^n} \quad & f(x,\theta) - \mu \sum_{i=1}^{m_2} \log(x_i) \\ \text{s.t.} \quad & c_i(x,\theta) = 0, i = 1,\dots,m_1. \end{aligned}$$

The logarithm term in this formulation ensures that the solution x is nonnegative since it tends to infinity as x tends to zero - hence the term 'barrier'. The important observation is that the terms we add are convex, so if our original objective function is convex, then the new objective function remains convex. If $\mu \to 0$, then we should get a better approximation for the non-barrier problem. We note that the Lagrangian associated to this problem is given by:

$$\mathcal{L}_\mu(x,\lambda,\theta) := f(x,\theta) - \mu \sum_{i=1}^{m_2} \log(x_i) + \lambda^T c(x,\theta).$$

The idea is to track the solution of the system

$$\begin{aligned} & \nabla_x \mathcal{L}_\mu(x,\lambda,\theta) = 0, \\ \text{s.t.} \quad & c_i(x,\theta) = 0, i = 1,\dots,m_1. \end{aligned}$$

If (x^*,λ^*,θ^*) is a solution of the previous system, we can continue that solution in a small neighborhood of (x^*,λ^*,θ^*) as long as $\frac{\partial}{\partial(x,\lambda)}\nabla_x \mathcal{L}_\mu(x,\lambda,\theta)$ is non-degenerate. [2]. This motivates the notion of path stability.

Definition 2.3 ([2, Path Stability])**.** The problem 4 is said to be *path-stable* if the expression $\frac{\partial}{\partial(x,\lambda)}\nabla_x \mathcal{L}_\mu(x,\lambda,\theta)$ is non-degenerate for all points x such that $c(x,\theta) = 0$ and $x_i > 0$ for all $i = 1,\dots,m_1$.

That is, we want to be able to trace the solution. In [2, Theorem 3.7] it is shown that if the optimization problem is zero-convex, path-connected and path-stable, then the barrier problem has a unique solution for each $\theta \in [0,1]$ and $\mu > 0$.

2.2.2 Barrier formulation for multi-objective optimization

We can now consider multi-objective optimization. The idea is that constraints are added of the form $f_i(x) \leq f_i(x_i^*)$ where x_i^* is the previously found solution. Those solutions are added as inequality constraints just as we did with the non-negativity of the coordinates. We can then transform the j-th problem in the optimisation into the following barrier formulation $\mathcal{P}_\mu^{j,\theta}$:

$$\begin{aligned} \min_{x\in\mathbb{R}^n} \quad & f_j(x,\theta) - \mu\sum_{i=1}^{m_2}\log(x_i) - \mu\sum_{i=1}^{j-1}\log(f_i(x_i^*,\theta)+\varepsilon_i - f_i(x_i,\theta)) \\ \text{s.t.} \quad & c_i(x,\theta)=0, i=1,\dots,m_1, \end{aligned} \tag{5}$$

where we allow for a deviation $\varepsilon_i > 0$ from the previous solution. The reason we allow for this deviation is the following: if the previous problem has one unique solution, we might not be able to minimize the other objectives. However, if we are allowed to deviate slightly from that one unique solution, we can search more of the space to also minimize the other objectives. Secondly, this means that the point we have previously found is still valid, and thus we may use approximation methods starting from this point. This leads to the following definition:

Definition 2.4 (Path-Stability for Multi-Objective optimisation problems)**.** We say that a multi-objective optimization problem is path-stable if none of the sub-optimization problems have a degenerate Lagrangian for all feasible points (that is, points that satisfy the constraint and are allowed in the optimization objective).

We note that by adding barrier terms, the notion of feasible point changes. This definition might seem a bit strange because our optimization objective changes. Eventually, though, we will get to the final optimization problem, which is the one we continue, and that final function can be treated as the single objective function from before. Then the analysis of the original paper can be mostly carried over with minor adjustments.
Finally, note that the optimization objective remains convex:

Lemma 2.5 ([4, Example 3.13])**.** *Let g be a convex function. Then the function $-\log(-g(x))$ is convex on the set $[g<0]$.*

Then the term $f_i(x,\theta)-\varepsilon_i-f_i(x_i^*,\theta)$ - the argument of the last logarithm in Equation 5 - is evidently convex if we assume that all f_i are convex. This is the main motivation for going with a barrier formulation.
Alternatively to the barrier formulation, one could consider the generalized Lagrangian. When tracing the solution, however, the terms that come from the inequality constraint must satisfy very specific conditions, such as the non-negativity of the generalized parameter for the inequality constraint, which we cannot guarantee from the implicit function theorem. In the barrier formulation, we can keep the equality constraints from the original analysis, which makes this method a more convenient one.

2.2.3 Order of optimisation

The theory behind the Homotopy method for single-objective optimization is described in [2]. We have already explained how we aim to deal with the additional inequalities that are obtained from goal programming. However, this does not fully determine the order in which we perform goal programming. There are two approaches one could take. We can optimize all the goals for each θ, but this means that the optimization function changes wildly between optimization runs and therefore does not align with the continuation of the Homotopy.

We propose the following idea. We solve all optimization problems in order and try to continue the solution. After we perform the Homotopy, we get a solution x_i^* at $\theta=1$. When we want to optimize the $i+1$-th problem, we look at the convex problem at $\theta=0$ with the inequality constraints. The problems are easier to solve, so we can exploit the convexity at $\theta=0$ to efficiently find a new starting point for the next optimization whilst retaining a good solution. We note that to compare the function values between $\theta=0$ and $\theta=1$, the optimization function cannot change with θ, but this is of minor importance. This way, we can ensure we can study one objective function at a time which helps with the analysis of the problem as the Lagrangian needs to deform smoothly, which is not guaranteed if we jump from objective functions.

3 Methods and Analysis

The goal of this project is to decrease the runtime of the current methods. One way to achieve this is by skipping certain steps in the optimization process. The question arises whether the method still converges to the desired solution if those steps are skipped. Convergence might not even be guaranteed with all steps in place. Therefore, our job is twofold: we investigate the current methods in a mathematical framework and test a relatively simple approach to reducing the number of steps needed.

3.1 Analysis of Homotopy method in Multi-Objective Optimisation

We would like the model to adhere to the natural laws of the world. Therefore, in this section, we look specifically at two very important constraints when using the Homotopy method, namely the (discretized) Saint-Venant equations. We will first describe the discretization of the Saint-Venant equations and then discuss some properties as discussed in [2], which carry over almost verbatim. This will give us path stability as defined in our multi-objective optimization problem. Furthermore, we will discuss the notion of path-connectedness, which is the main problem when applying the Homotopy method to multi-objective optimization.

3.1.1 Saint-Venant Equations Discretised

Let us first describe the discretized version of the Saint-Venant PDEs, the equality constraints that govern the optimization problem we consider. The description will be the same as in [2].

Linear approximation of the Saint-Venant equations.
A linear approximation to the Saint-Venant equations is given by:

$$\begin{aligned} \frac{\partial Q}{\partial t} + g\overline{A}\frac{\partial H}{\partial x} + g\frac{Q|\overline{Q}|}{\overline{A}\cdot\overline{R}C^2} &= 0 \\ \frac{\partial Q}{\partial x} + \overline{\omega}\frac{\partial H}{\partial t} &= 0, \end{aligned} \tag{6}$$

where $\overline{A}$ is the nominal cross-section, i.e. the two-dimensional surface area in a cross-section, $\overline{R}$ is the nominal hydraulic radius, i.e. the average ratio of surface area to wetted perimeter, the portion of the area that is wet, and $\overline{Q}$ the average discharge, the amount of water being transported.

The linear approximation considers a rectangular cross-section with nominal width $\overline{\omega} := (dA/dH)(\overline{H})$. The needed nominal values are gathered as data and used as parameters in parts of the model considered in [2].

Discretization of the Saint-Venant equations.
The discretization makes a distinction between *interior variables*, i.e. variables with two neighboring hydraulic variables, and all other hydraulic variables, referred to as *boundary variables*. We want to emphasize that the variables in question are the water level H and discharge Q.

The model makes use of a staggered grid for the hydraulic equations and is semi-implicit in time. For the implementation of the model we refer back to [2], but for the purpose of this report, these are but details visible in the definitions of the following functions.

Introducing the Homotopy parameter θ in combination with the discretized linear and non-linear mass balance equation, we obtain:

$$c_{i,j} := \frac{Q_i(t_j) - Q_{i-1}(t_j)}{\Delta x} + \theta\frac{A_i(H(t_j)) - A_i(H(t_{j-1}))}{\Delta t} + (1-\theta)\overline{\omega}\frac{H_i(t_j) - H_i(t_{j-1})}{\Delta t} = 0, \tag{7}$$

for all $i \in I_H$, the index set containing all indices of the interior H_i variables, and all $j \in \{1, \ldots, T\}$.

Likewise, when introducing the Homotopy parameter θ in the equation containing the discretized linear and non-linear momentum equation, we obtain the following equation:

$$\begin{aligned} d_{i,j} := \frac{Q_i(t_j) - Q_{i-1}(t_j)}{\Delta t} + \theta e_{i,j} + g\big(\theta A_{i+\frac{1}{2}}(t_{j-1} + (1-\theta)\overline{A}\big)\frac{H(t_{j+1}) - H_i(t_j)}{\Delta x} \\ + g\left(\theta \frac{P_{i+\frac{1}{2}}(t_{j-1})) + \text{ sabs } (Q_i(t_{j-1}))}{A_{i+\frac{1}{2}}(t_{j-1})^2} + (1-\theta)\overline{\omega}\frac{\overline{P} \text{ sabs } (\overline{Q})}{\overline{A}^2}\right)\frac{Q_i(t_j)}{C_i^2} = 0, \end{aligned} \tag{8}$$

for all $i \in I_Q$, the set of indices of the interior Q_i variables, $j \in \{1, \ldots, T\}$ and where C_i denotes the local friction coefficient and the variables $A_{i+\frac{1}{2}}(t_j)$ and $P_{i+\frac{1}{2}}(t_j)$ are defined as:

$$\begin{aligned} A_{i+\frac{1}{2}}(t_j) &:= \frac{1}{2}(A_i(H_i(t_j)) + A_{i+1}(H_{i+1}(t_j))); \\ P_{i+\frac{1}{2}}(t_j) &:= \frac{1}{2}(P_i(H_i(t_j)) + P_{i+1}(H_{i+1}(t_j))). \end{aligned}$$

Moreover, to avoid singular derivatives, we need the following definition:

$$\text{sabs } x := \sqrt{x^2 + \varepsilon},$$

where ε is a small and smooth error-approximation for $|x|$. Lastly, the model calls for the convective acceleration $e_{i,j}$:

$$\begin{aligned} e_{i,j} = \text{sH}(Q_i(t_{j-1}))\frac{2Q_i(t_{j-1})}{A_{i+\frac{1}{2}}(t_{j-1})} \cdot \frac{Q_i(t_{j-1}) - Q_{i-1}(t_{j-1})}{\Delta x} \\ + (1 - \text{ sH}[Q_i(t_{j-1})])\frac{2Q_i(t_{j-1})}{A_{i+\frac{1}{2}}(t_{j-1})} \cdot \frac{Q_{i+1}(t_{j-1}) - Q_i(t_{j-1})}{\Delta x} \\ - \frac{Q_i(t_{j-1})^2}{A_{i+\frac{1}{2}}(t_{j-1})^2} \cdot \frac{A_{i+1}(H_{i+1}(t_{j-1})) - A_i(H_i(t_{j-1}))}{\Delta x}, \end{aligned} \tag{9}$$

with the logistic function: $\text{sH}(x) := (1 + \exp(-Kx))^{-1}$, and steepness factor $K > 0$.

We have shown this cursory explanation to be able to give some insight into the model. The two discretized equations, $c_{i.j}$ and $d_{i,j}$, are used as primary unchanging constraints throughout the goal-programming approach. For more in-depth information and derivation of the constraints, we refer to [2]. [2],

3.1.2 Assumptions needed for the model.

We recreated the discretization of the Saint-Venant equations, as discussed in [2]. To ensure the necessary properties of the model and valid results, the paper lists several assumptions, which we will also adhere to:

OBJ The objective function $f(x, \theta)$ is twice continuously differentiable and convex to x for all $0 \leq \theta \leq 1$. In case of multi-objective optimization, we assume all objective functions satisfy this property.

ICO The initial values for the discharge and water level at time t_0 and place i, $Q_i(t_0)$ and $H_i(t_0)$, are provided for all i. We replace these values in the model so that all variables of t_0 do not appear in the optimization problem.

BND All free boundary variables have both a lower and a strictly larger upper bound. We comment that as we add constraints, the interior hydraulic variables may be restricted. In the original paper, it was assumed that the interior hydraulic variables are unbounded. This was only needed for the path-connectedness argument, which we appropriately modify here.

HBC The water level boundary conditions $H_i(t_j)$ are fixed for all i and time t_j. The boundary conditions are replaced with the values so that the optimization problem does not contain the water level boundary conditions.

QBC There is at least one free flow boundary condition. Any two free-flow boundary conditions must have at least one interior flow variable situated between them.

The theorems proven in [2] often need these assumptions. We shall use those theorems from the paper, which can be applied directly even when certain notions change.

3.1.3 Path-Stability

We have already seen that path stability only holds if the Barrier formulation is free of singularities. The following lemma proven in [2] provides a useful characterization of path-stability. We first need the notion of the tangent space of the constraints. This coincides with the usual definition of the tangent space.

Definition 3.1 ([2, Definition 3.1])**.** We define the tangent space of the constraint set $c(x, \theta)$, denoted by $T(x, \theta)$, as the set

$$T(x, \theta) := \{y : \langle \nabla_x c(x, \theta), y \rangle = 0\}.$$

Proposition 3.2. *[2, Proposition 3.4] For $\theta \in [0, 1]$ and $\mu > 0$ fixed, the associated barrier problem $\mathcal{P}_\mu^{i,\theta}$ is path stable if and only if for each feasible point x it holds that*

- $\frac{\partial}{\partial(x,\lambda)} \nabla_x \mathcal{L}_\mu^{i,\theta}(x, \lambda, \theta) =: \nabla_{xx}^2 \mathcal{L}_\mu^{i,\theta}(x, \lambda, \theta)$ *is non-singular on the tangent space.*
- *The Jacobian of the constraint gradients, denoted by $\nabla_x c(x, \theta)$, has full rank.*

For the multi-objective optimization case, we can append the extra convex function to the original convex function of the optimization. However, the notion of a feasible point does change when adding the convex functions.
The next part of proving path stability is to prove that the Hessian of the Lagrangian is non-singular when we differentiate with respect to the boundary variables. This is captured in the following proposition, which roughly has the same reasoning as in [2, Lemma 4.4]:

Proposition 3.3. *Consider the Lagrangian $\mathcal{L}_\mu^{i,\theta}(x, \lambda, \theta)$ associated to the i-th optimisation problem. Then the Hessian matrix $\nabla_{x_{bdy} x_{bdy}}^2 \mathcal{L}_\mu^{i,\theta}(x, \lambda, \theta)$ with respect to the boundary variables x_{bdy} is positive definite.*

Proof. We note that the Hessian of this objective function can only change because of the extra barrier terms added in the case of multi-objective optimization. However, as already remarked, these barrier terms are convex. Hence, their Hessian with respect to the boundary variables is positive semi-definite. We already showed that the Hessian of the Lagrangian without the additional terms is positive definite under the conditions we described earlier. Therefore, their sum must remain positive definite, as well. □

From here, we can prove the non-singularity of the Hessian matrix $\nabla_{xx}^2 \mathcal{L}_\mu^{i,\theta}$. This was already proven in Baayen for one constraint using the previous proposition. But then this will carry over verbatim to our problem. Thus we obtain the following proposition:

Proposition 3.4. *Fix $\theta \in [0, 1]$ and $\mu > 0$. Then the Hessian matrix $\nabla_{xx}^2 \mathcal{L}_\mu^{i,\theta}$ is non-singular on the tangent space $T(x, \theta)$.*

If we combine this with the characterization of path stability that relates this to Hessian matrices and non-singularity of the constraint gradients [2, Proposition 4.1] on a larger search space, it follows that $\mathcal{P}_\mu^{i,\theta}$ stays clear of bifurcations by virtue of Proposition 3.2.

3.1.4 Path-Connectedness

Let us now look at the path connectedness of the problem in the context of goal programming. We have already seen that in this new context, more barrier problems are added so that the path stability of the multi-objective optimization problems can be analyzed. When we translate to a barrier problem, the domain of feasible points gets adjusted to the geometry induced by the objectives. For example, when we perform our first optimization problem, the domain gets restricted to both the Saint-Venant equations and the set $[f_i \leq f_i(x_i^*) + \varepsilon_i]$. The proof of path-connectedness provided in [2] relies on the fact that the image of a path-connected set under a continuous map is path-connected. Concretely, we have a continuous map $g : x_{bdy} \to x_{hyd}$. That is, the solution of the Saint-Venant equations is entirely determined by the boundary variables. The boundary variables are discretized linearly, even for $\theta > 0$. Thus, the set of allowed boundary variables is convex.

However, we can only say that $[f_i \leq f_i(x_i^*) + \varepsilon_i]$ is convex. It is, in general, not true that the intersection of a convex set and a path-connected set remains path-connected. One can, for example, think of a circle and a line. They intersect in two points only and hence we may have that the geometry of our sets gets 'shattered' by the convex geometry.
A bit more can be said if we take the shape of our restrictions into account. We only come across restrictions of the form $[f_i(x) < f_i(x_i^*) + \varepsilon_i]$ for feasible interior points. These sets are convex. The projection onto the boundary variables $\pi_{x_{bdy}}$ is linear and therefore $\pi_{x_{bdy}}([f_i(x) < f_i(x_i^*) + \varepsilon_i])$ is convex. As the discretization is linear in the boundary variables, the feasible boundary variables of the Saint-Venant equations are convex. As the intersection of convex sets is convex, we obtain a convex set of boundary variables.

The problem, however, is that one loses information when projecting onto the boundary variables for the set $\pi_{x_{bdy}}([f_i(x) < f_i(x_i^*) + \varepsilon_i])$ where this did not happen for the Saint-Venant equations. Nevertheless, we always have a feasible point by construction. In a small enough region of this feasible point the inequalities $f_i(x) < f_i(x_i^*) + \varepsilon_i$ are satisfied. Since the map g from the boundary to the interior variables is continuous in a sufficiently small neighborhood around the feasible point, all the values $f_i(x)$ are strictly below $f_i(x_i^*) + \varepsilon_i$. Since g is continuous, we can make the distance of the interior part of the feasible point sufficiently small if the boundary part is sufficiently close. We note that for a sufficiently small region, we can satisfy all the inequalities $f_i(x) \leq f_i(x_i^*) + \varepsilon_i$. Since the space of interior variables is not restricted, the vector constructed by combining the boundary part y_{bdy} and $g(y_{bdy})$ is a feasible point. Thus, we have local path connectedness.

The hope is that the path-connected region is sufficiently rich to discover new solutions. Currently, we are not able to conclude that we can uniquely trace the solution in any order of optimization. Nor can we find sufficient conditions on the objective functions.

3.2 Computational Methods

The above-described Homotopy method is implemented in the `rtc-tools` package developed by Deltares. From our analysis, we can conclude that this Homotopy method at least ensures that we find a locally optimal solution. For the use case of water management, this is considered sufficient, as long as the solution is robust. That means that the solution does not change significantly every hour, so that the advice of the pump operators is consistent every hour. Deltares indeed found this to be the case for their model.

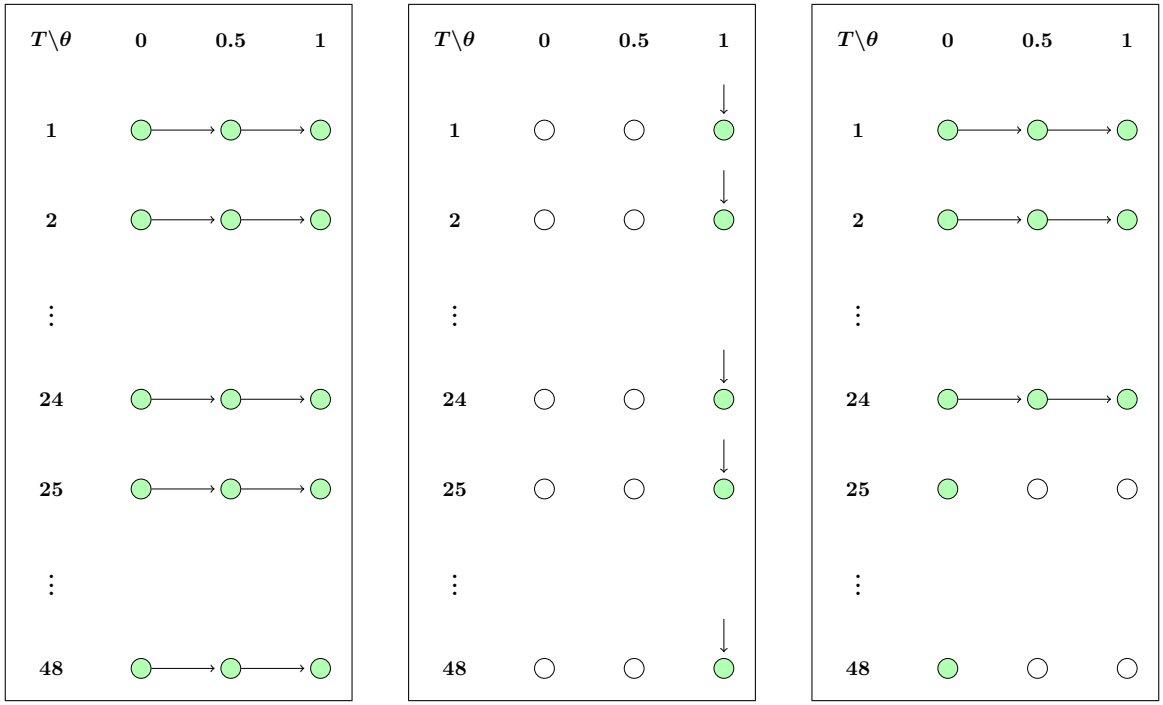
Now, we move to the second goal of our project. Even though the Homotopy method seems to work quite well - even from a theoretical perspective - it is computationally quite expensive; to trace the optimal solution, we have to solve the optimization for every theta step. In practice, the runtime is already significantly reduced by taking steps of $\Delta\theta = 0.5$. However, the runtime is still significant. To improve this, we came up with two methods that build on the Homotopy method. The gist of these methods is that they skip certain steps in the optimization process to reduce the runtime. In the following sections, these methods will be explained in more detail.

3.2.1 SmartSeed Method

The first method for decreasing the runtime is the so-called SmartSeed method. In this method, we make use of the fact that the optimization is run every hour. Since every run gives us data for the next 48 hours, we can use the output of the previous run, as a seed for $\theta = 1$ for the next run. This way the linear $\theta = 0$ and intermediate $\theta = 0.5$ can be skipped, which could result in significant runtime improvements. Since the solution found in the previous hour should be quite robust, we expect this to be a sufficient seed to find the optimal solution. Nonetheless, this previous computation does not always function as a sufficient seed. In this case, the SmartSeed method falls back on the Homotopy method. That way, we can ensure that the convergence of the solutions is still robust. However, by the previous argument, this should not happen too often under normal conditions. The SmartSeed method is schematically displayed in Figure 1b.

3.2.2 Linearising after $T = 24$

A second approach for decreasing the runtime is solving the non-linear problem for the first 24 timesteps, and solving the linear system for the last 24 timesteps. The justification for this approach comes from the use case of the model. The most important part of the output of the model is the prediction for the first 24 hours, while the last 24 hours are merely an indication of what is to come. The latter ensures that the operators can prepare for more extreme changes, in case of heavy rainfall, for example, but we do not need to know exact water levels with high accuracy. By skipping half of the nonlinear steps, we might exchange some accuracy for a significant improvement in runtimes. Schematically, the linearising method is displayed in Figure 1c.



(a) Homotopy method (b) SmartSeed method (c) Linearsing method

Figure 1: Schematic overview of the different methods. In the header, the θ steps are displayed, while the index represents the time. For the green nodes, the optimization problem is solved for this combination of (t, θ). In (a) the conventional Homotopy method is displayed. Here, we solve all timesteps for all θ. In (b), a schematic view of the SmartSeed method is displayed. Here, we use the input of the previous run as a seed for $\theta = 1$, indicated by the vertical arrows. This way the $\theta = 0, 0.5$ steps can be skipped. Lastly, (c) represents the linearization method. For the first 24 hours, the Homotopy method is used, but for the last 24 hours, only the linear model is solved.

4 Computational results

Originally, the use case of the Homotopy method is to apply it to the Rijnland model. Since this is quite a large model and thus has long runtimes we decided to test our methods on a smaller model. Our methods will be tested making use of the cascading channels test model from the `RTC-Tools` package, provided by Deltares. Figure 2 depicts a schematic representation of the test model.

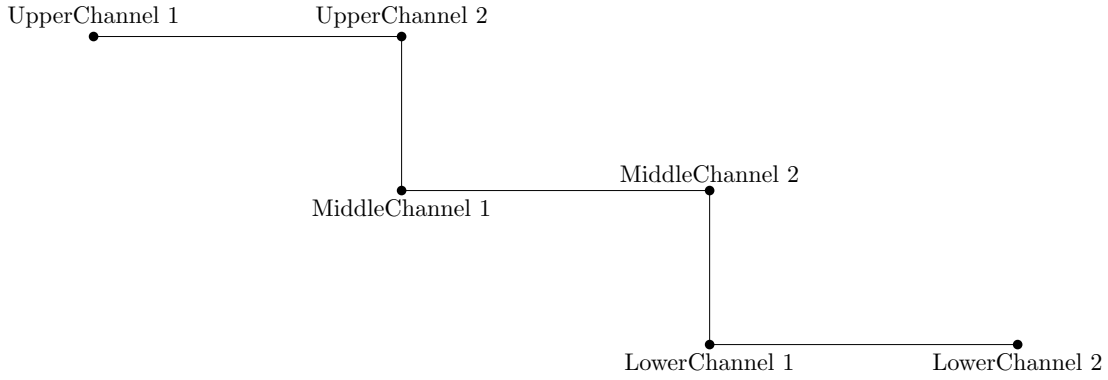


Figure 2: Schematic representation of the Cascading channels test model. Note that this figure is not to scale. The distance between UpperChannel 1 and UpperChannel 2 is much larger than the distance between UpperChannel 2 and MiddleChannel 1, for example.

This test model represents a system consisting of long channels, with height differences between these channels. The fluids in each of these channels are governed by the Saint-Venant equations as in Equation (1). When optimizing such a system, we have certain goals, such as we do not want the water level in the channels to surpass a certain threshold to prevent flooding. This could be done by transporting the water between the channels, by pumping it to a higher channel, or by discharging it to a lower channel. However, since pumping costs money, we want to minimize the time we spend pumping. This is where the optimization comes in: we want to satisfy all our safety regulations while also minimizing costs.

Now that we have established that the test model is indeed a suitable test case, we need to determine what constitutes a good method. When comparing the different methods, we need to consider three important criteria. First of all, we would like the solutions to converge to the same, or at least a similar, optimum. In that case, the robustness of the solutions is conserved. From the Homotopy method, we have learned that this is ensured by generating a seed that is close to the actual optimum. This means that the error of the seed compared to the optimum should be small. We define the error as the difference between the Smart Seed and the Homotopy method, the method that is currently used by Deltares. Lastly, we want our methods to be more efficient in terms of runtime, where we, again, compare to the runtime of the Homotopy method.

4.1 SmartSeed

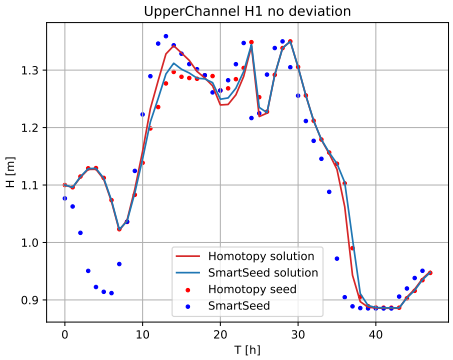
When the SmartSeed is close to the Homotopy method, we can conclude that it is justified to make use of the SmartSeed method. We therefore compare the seed of the Homotopy method with the seed of the SmartSeed method. When both seeds are similar, we can assume that they converge to approximately the same locally optimal solution. To test this, we first run a base scenario test with the initial conditions from Table 1 using the Homotopy method. This will act as a proxy for the run of the 'previous' hour.

Variable	Value [m]
UpperChannel.H[1]	1.1
UpperChannel.H[2]	1.0
MiddleChannel.H[1]	0.6
MiddleChannel.H[2]	0.5
LowerChannel.H[1]	0.1
LowerChannel.H[2]	0.0

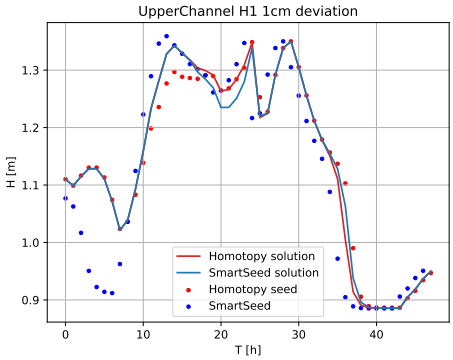
Table 1: Initial conditions for the base scenario. These initial conditions will be used for the SmartSeed method.

The results of this run will be used for the SmartSeed method. Now, we consider four different scenarios, with different initial conditions. Those runs represent the model run in the next hour when some of the conditions have been changed. Note that only the water height of UpperChannel 1 is changed for these runs, while the other variables are left unchanged.

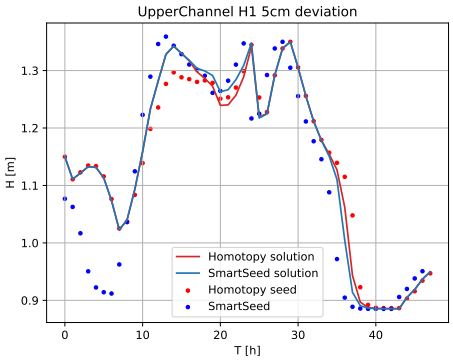
First, we consider the scenario where the water height is identical to the initial conditions. This would correspond to a system that was in a steady state, that is, no inflow of water via pumping, rain, or discharge. Secondly, we consider the scenario where the water level is increased by 1 cm, which means that we have some inflow of water at UpperChannel 1. This can be considered quite a realistic scenario. Thirdly, we consider a more extreme scenario, where the difference between the initial conditions is 5 cm. Lastly, a quite unrealistic case is considered where the difference between the initial solution is 25 cm. Note that this last scenario is mainly for testing the worst-case scenario of SmartSeeding. The seeds for the different scenarios are displayed in Figure 3.



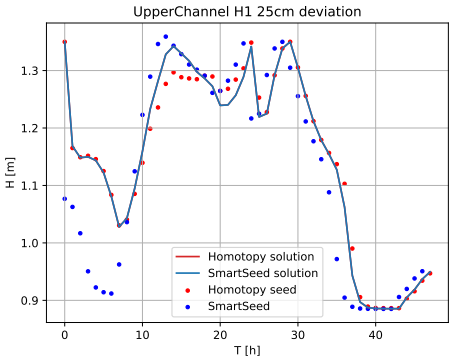
(a) Same initial condition



(b) 1cm initial condition



(c) 5cm initial condition



(d) 25cm initial condition

Figure 3: Seeding used by the Homotopy method and the SmartSeed method for different initial conditions. The red dots represent the final solution, whereas the blue and orange lines represent the seed that is put into a run with $\theta = 1$. For the Homotopy method, this is the result for $\theta = 0.5$ and for the SmartSeed, this is the result of the previous run.

Two observations can be made about this figure. First, note that the SmartSeed seed does not start at the same height, even if there are no deviations. This is because the SmartSeed uses the results from the last hour, so the initial conditions have changed slightly during this one hour. Furthermore, we see that SmartSeed can be quite different compared to the seed generated by the Homotopy method. Fortunately, we see that the solutions are quite similar in the end, which indicates that using the SmartSeed method can be a good strategy for estimating the seed.

4.2 Linearised after $T = 24$

In this subsection, we want to compare the solutions generated by the Homotopy method and the linearised method. If these methods result in approximately the same solution, it would be justified to linearise for the later timesteps. The solutions are plotted in Figure 4.

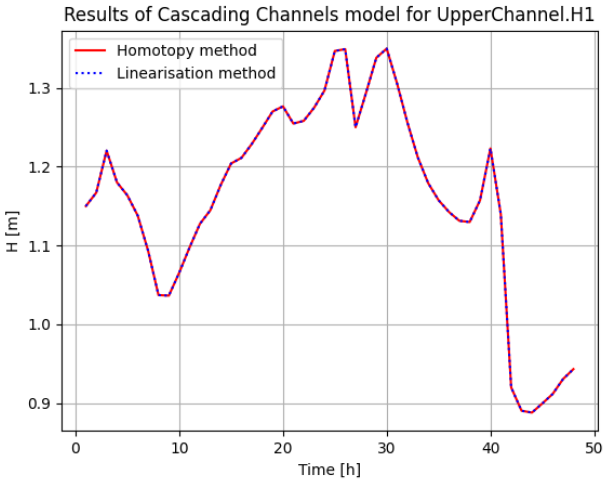


Figure 4: Plot of the solutions for UpperChannel.H[1] for 5 cm deviations from the initial condition for the linearisation and Homotopy method.

Note that the solution generated by the partially linearised method is the exact same as the solution generated by the Homotopy method.

4.3 Runtime

Lastly, we would like the proposed methods to reduce the runtime of the optimization. To measure this, we varied the water heights in all channel nodes by 0, 5, or -5 cm. This results in a total of $3^6 = 729$ different initial conditions. For all these runs, we measure the runtime of the optimization. The results are plotted in Figure 5.

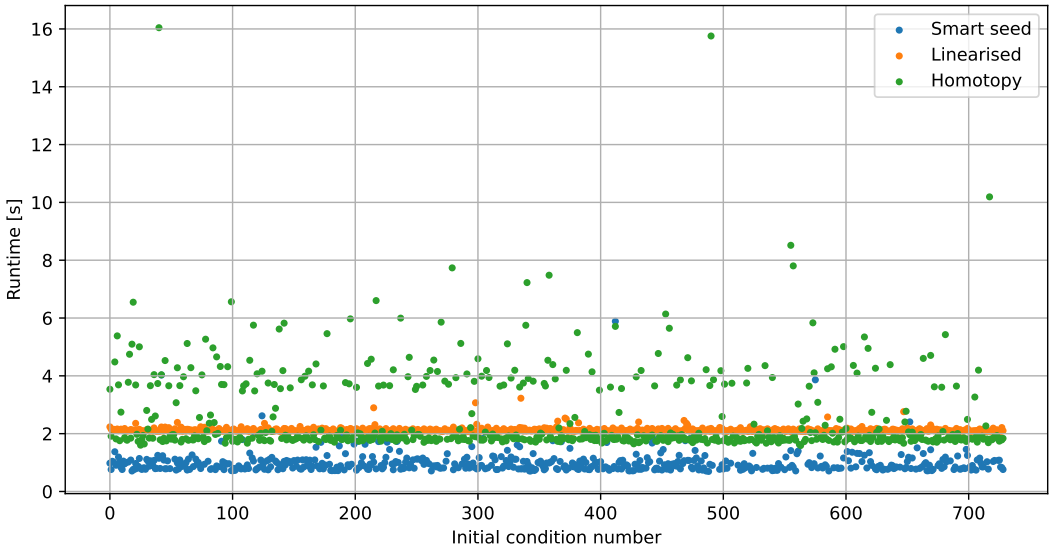


Figure 5: Runtime for the different methods for 5cm deviations from the initial condition.

Qualitatively, the SmartSeed method appears to have a shorter runtime than the linearised and

Homotopy method. However, we have to keep in mind that in some scenarios the optimal solution could not be found. The results can be summarised in Table 2.

Method\Results	**Average runtime (s)**	**Runtime improvement (%)**	**No. of divergence**
Homotopy	2.565	0	105
SmartSeed	1.006	60.8	192
Linearised	2.126	17.1	0

Table 2: Data of the average runtime, a runtime improvement compared to the Homotopy method, and several initial conditions where no optimal solution was found.

5 Discussion and Conclusion

Our analysis and implementations have given us plenty to discuss. We will discuss our findings in order of occurance in the rest of the report.

5.1 Theory

The main contribution of the theoretical part is reformulating goal programming so that the context fits that of the Homotopy method. We analyzed the notion of path-stability, as defined in [2], in this new setup. The main insight is that we think one needs to focus on one objective at a time to ensure a traceable Homotopy path. In addition, we can also exploit that the problem at $\theta = 0$ is convex, if we allow for the more general definition of a convex problem. We use this to find a point to start the second Homotopy. However, this is only a beginning in the understanding of the Homotopy method in multi-objective optimization.
Although we have tried to study path-connectedness rigorously, and it does look promising for multi-objective optimization, the topic still requires more research. Our endeavors do not guarantee global path-connectedness for the goal-programming approach. However, we can guarantee local optimality, which, as Deltares commented, is in practice good enough.

5.2 SmartSeed

The results of the experiments with the SmartSeed method are quite well summarised in Figure 3. First note that the SmartSeed, the blue dots, are identical in all of the figures. This is expected, since these are the results of the last run/hour, and thus will not change. For all the SmartSeeds we see that initially, they are quite far off compared to the seed computed by the Homotopy method. This is to be expected since the initial conditions are perturbed. After about 12 hours the seeds move closer to each other. This means that the SmartSeed is initially not a very good seed, but for later times it is. When comparing the solutions of the SmartSeed method with the solution of the Homotopy method, most of these differences seem to disappear. For the 0cm, 1cm, and 5cm deviations, we only see a difference in the solution in a small time window of a few hours. For the 25 cm deviation, the results are identical. What we can conclude from this is that the SmartSeed might not be close to the seed generated by the Homotopy method, but it does generate quite similar solutions. Qualitatively, the solutions even seem to be identical, which is an indication of the robustness of the method.

5.3 Linearising after $T = 24$

The results plotted in Figure 4 suggest that linearising the model after $T = 24$ would give the same result as the Homotopy method. This is not as expected, since the model is inherently non-linear. Because of this, we believe that the implementation of the linearised model contains a bug and we have to be skeptical of these results.

5.4 Runtime

Lastly, we need to take a look at the results for the runtime. We start with the Homotopy method. Here we see that most of the time, the runtime is under 2 seconds. However, there are also quite some initial conditions for which the solution took longer to find. We suspect that in those cases the theta step of 0.5 was too big and we needed to backtrack. For the linearising method, we see that the average runtime is slightly above 2 seconds. This does not make sense, since for half of the time steps we only solve the linear model, which supports our hypothesis of the previous section. What we do see is that this model seems to be a bit more consistent in its runtime, perhaps because it needs to do less backtracking. Lastly, the SmartSeed method is about 60 % faster when compared with the Homotopy method, which is as expected. The drawback of this method is that it is not as robust as the Homotopy method, in the sense that for quite a large portion of the initial conditions, no optimal solution is found.

5.5 Conclusion

From our research, we can conclude that the SmartSeed method seems to be a good method to improve the runtime while keeping the solutions quite similar to the solutions generated by the Homotopy method. We have tested this to be the case for small and large deviations of the initial conditions. However, we have not tested this for changing the inflow variables of the system. For the linearising method, we also see decreases in the runtime and identical solutions to the Homotopy method. However, as stated before we are quite sceptical towards these results.

5.6 Remarks

An important remark to make is that all the analysis and tests were done on a scaled-down version of the model used by Deltares, namely the so-called cascading channels model. However, this model still uses the same optimization method, thus we can expect it to generalize to the larger model.

5.7 Recommendations

Our suggestion would be to try to implement the SmartSeed method in the larger model and test if the robustness and runtime improvement of this method generalize to the larger model. Besides this, we believe that the linearising could also yield significant runtime improvements if implemented correctly. Furthermore, it might be worth studying the effect of changing the inflow and outflow variables on the SmartSeed method. In our research, we have not been able to implement this due to time constraints, but it is an important aspect of the model.

5.8 Acknowledgements

We want to thank the problem posers Ailbhe Mitchell and Bernhard Becker for their help in understanding the problem, explanations of the code, and ready availability to answer our questions along the way.

References

[1] B. B., O. D., and Piovesan, "A comparison of the homotopy method with linearisation approaches for a non-linear optimization problem of operations in a reservoir cascade," *Energy Systems*, 2023. DOI: https://doi.org/10.1007/s12667-023-00608-w.

[2] J. Baayen, T. Piovesan, and J. VanderWees, *Continuation method for pde-constrained global optimization: Analysis and application to the shallow water equations*, 2020. arXiv: 1801.06507 [math.OC].

[3] G. Eichfelder, "Twenty years of continuous multiobjective optimization in the twenty-first century," *EURO Journal on Computational Optimization*, vol. 9, p. 100 014, 2021, ISSN: 2192-4406. DOI: https://doi.org/10.1016/j.ejco.2021.100014. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2192440621001416.

[4] S. P. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.