Dawid Kalicki

# Model based control of water flow and level in river systems with hydropower plant

Master’s thesis in Cybernetics and Robotics
Supervisor: Sebastien Gros
Co-supervisor: Øyvind Holm
June 2022

**Master’s thesis**

**NTNU**
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Engineering Cybernetics

NTNU
Norwegian University of
Science and Technology

VOITH

Dawid Kalicki

# Model based control of water flow and level in river systems with hydropower plant

Master's thesis in Cybernetics and Robotics
Supervisor: Sebastien Gros
Co-supervisor: Øyvind Holm
June 2022

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Engineering Cybernetics

NTNU
Norwegian University of
Science and Technology

# Preface

This thesis is the final work of a 2 years long Master of Science in Engineering degree programme at Norwegian University of Science and Technology (NTNU) in Trondheim. I started at this programme after finishing Bachelor's degree in Electrical Engineering. The topic of water-level control at hydropower plants was new for me and turned out to be a complex and interesting problem, that I enjoyed working on. The thesis is written in cooperation with Voith Hydro AS. I would like to thank Øyvind Holm for suggesting and giving me the opportunity to work with this project and all other employees at Voith that introduced me to the topic of water-level control. I would also like to thank my supervisor Sebastien Gros for valuable guidance throughout this project. Big thanks goes also to Bjørnar Petersen for providing access to power plant data used in the thesis.

# Summary

Simple controllers like PI controllers are most common in the implementation of water level control systems at hydropower plants. Such simple controllers have limitations, especially when more advanced control behaviors are desirable. In this thesis, a way to implement a model predictive controller (MPC) on a run-of-river hydropower plant is presented. The MPC allows for easier implementation of some features such as constraining water level change rate, and keeping the control inputs constant. In addition, the MPC is able to utilize upstream measurements to improve the controller response by predicting future behaviors. To implement an MPC, a river model has been built using St. Venant equations. This model is also used by a moving horizon estimator (MHE) that can estimate states and disturbances based on available measurements, which is necessary to run the MPC. The MHE can also be degraded to an open-loop estimator that can be used in case of data loss. The presented solution has been tested on simulations with real data from powerplants Funnefoss, Rånåsfoss, and Bingsfoss, with good results. The MPC has not been tested on a real powerplant, which is necessary to confirm its performance.

# Sammendrag

Enkle regulatorer som en PI-regulator er ofte brukt i implementasjon av vannstandsregulering på elvekraftverk. Slike enkle regulatorer kan være vanskelig å bruke til å implementere mer avanserte regulator funksjoner. I denne oppgaven presenteres en måte å implementere en MPC regulator for vannstandsregulering på elvekraftverk. MPC gjør det enklere å implementere funksjoner som begrensning av vannstandsendring, og å opprettholde et konstant pådrag. I tillegg kan MPC bruke målinger som er tatt lenger opp elvestrekning, for å forutse hvordan de vil påvirke vannstanden og på denne måten respondere kjappere på framtidige forstyrrelser. Modellen av elven som brukes i MPC-en er basert på "St. Venant equations". Denne modellen er også brukt i en MHE, som estimerer tilstander og forstyrrelser i systemet som er nødvendige for å kunne kjøre MPC-en. MHE-en kan også bli degradert til en åpen sløyfe estimator som kan bli brukt blant annet ved tap av data. Løsningen som er presentert har blitt testet ved hjelp av simulasjoner som brukte ekte data fra kraftverkene i Funnefoss, Rånåsfoss og Bingsfoss, med gode resultater. MPC-en har ikke blitt testet på et ekte kraftverk, noe som er nødvendig for å bekrefte dens faktiske reguleringsevne.

# Contents

# Acronyms

| | |
|---|---|
| BING | Bingsfoss |
| DOF | Degree of freedom |
| FUNN | Funnefoss |
| MHE | Moving Horizon Estimator |
| MPC | Model Predictive Controller |
| NLP | Nonlinear program |
| RANAS | Rånåsfoss |
| SP | Setpoint |
| SVE | St. Venant equations |

# Symbols

| Symbol | Description | Unit |
|---|---|---|
| $A$ | cross section area | $m^2$ |
| $v$ | water velocity | $m/s$ |
| $h$ | water level above river bed | $m$ |
| $Q_l$ | lateral inflow per unit length | $m^3/s$ |
| $v_l$ | downstream component of velocity of the lateral inflow | $m/s$ |
| $z_b$ | river bed height above sea level | $m$ |
| $S_x$ | channel slope | |
| $S_f$ | friction slope | |
| $R$ | hydraulic radius | $m$ |
| $P$ | wetted perimeter | $m$ |
| $k_s$ | friction coefficient | $m^{1/3}/s$ |
| $g$ | gravity on earth | $m/s^2$ |
| $Q_{\text{in}}$ | inflow to a river segment | $m^3/s$ |
| $Q_{\text{out}}$ | outflow from a river segment | $m^3/s$ |
| $Q_{\text{aggs}}$ | flow through power plant's aggregates | $m^3/s$ |
| $Q_{\text{gates}}$ | flow through power plant's gates | $m^3/s$ |
| $n$ | index of a river segment | |
| $t$ | time | |
| $k$ | discrete time | |
| $y$ | measurement | |
| $y^{\text{ref}}$ | measurement referance/setpoint | |
| $u$ | control input | |
| $x$ | state | |
| $d$ | measured disturbance | |
| $w$ | unmeasured disturbance | |
| $v$ | measurement noise | |
| $\theta$ | model parameters | |
| $J$ | cost function | |
| $z$ | decision variables | |
| $N_c$ | MPC horizon length | |
| $N_e$ | MHE horizon length | |

# 1 Introduction

## 1.1 Background

Hydroelectric power is the main power source in Norway, with over 1600 hydropower plants and a total yearly production of 138 TWh [26]. When running a power plant, one must comply with requirements on the watercourse that is being used. Some of the common requirements are minimum water level or maximum change rate of water level. These constraints are often related to aquatic life for instance salmon and other fish species, as well as to not cause damage to surrounding infrastructure and land. On a run-of-river power plant, the water reservoir is often small or almost not existing. This makes the control of such systems more challenging as small variations in water inflow have a bigger effect on the level than for a big reservoir. While keeping given requirements, the main goal is to utilize the available water resources as efficiently as possible, as this will result in bigger profit from plant operation.

### 1.1.1 Water level control

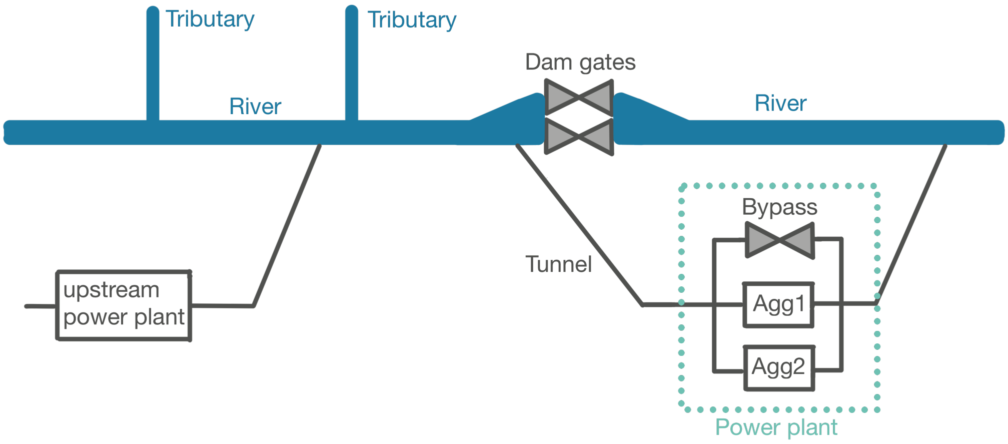


Figure 1: Scheme of a hydropower plant system

A common control problem for a run-of-river power plant is to keep water level at a given setpoint (SP). Volume $V$ of a liquid at a section of an open channel, can be described using mass balance, as shown in equation 1. The change of volume is equal to the flow of liquid into section $Q_{\text{in}}$, minus flow out of section $Q_{\text{out}}$. If the channel's geometry is known, the water level can be calculated from the water volume at that section.

$$\frac{\partial V}{\partial t} = Q_{\text{in}} - Q_{\text{out}} \tag{1}$$

In a case of a hydropower plant, it is often desirable to control the headwater, which is the water level on the upstream side of the dam. In that case, the in- and outflows can be decomposed to:

$$Q_{\text{out}} = Q_{\text{aggs}} + Q_{\text{gates}} \tag{2a}$$
$$Q_{\text{in}} = Q_{\text{measured}} + Q_{\text{lateral}} \tag{2b}$$

When running a hydropower plant, the flow through gates $Q_{\text{gates}}$ and the flow through the aggregates in the power plant $Q_{\text{aggs}}$ are the flows that can be controlled. These are affected by some additional internal dynamics for instance the time it takes water to travel between the intake and the plant, and the response time of the gates. In many cases the power plant is built directly on the river, then the travel time between the intake and the aggregates is approximately zero. The inflow to a river section $Q_{\text{in}}$ is harder to measure. It is common to measure the flow $Q_{\text{measured}}$ some kilometers upstream of the river. In some rivers, multiple power plants are laying in a cascade. Then the outflow from the upstream power plant can be used as a measure of the incoming flow. In both cases, the flow measurement is taken some distance from the power plant of interest. Therefore it will take some time before the measured flow reaches the dam. That delay is not constant and will vary with water's velocity, which is decided by the river's properties like slope, width, and length. This means that the $Q_{\text{measured}}$ can not be used directly in the equation 1. In addition, some river systems can have significant lateral inflow $Q_{\text{lateral}}$ in between the measurement point and the point the water level is to be controlled. Lateral inflow comes from tributaries, groundwater flow, or interflow and will vary with rainfall and snow melting, and is difficult to measure.

The most common solution for this control problem is a PI controller that uses the level measurement at the dam to calculate the $Q_{\text{out}}$ needed to keep the water level at a given setpoint. The output from the PI controller goes then to a flow divider, that distributes flow between each gate and aggregate in the power plant while keeping the requirements. One common requirement is the maximum rate at which the water level can change. This is currently implemented by using ramping of the setpoint signal. Another property that is wanted in water level control systems is to keep the control inputs constant for as long as possible. This reduces the wear of the components, which is especially important on the gates. These are big constructions and reduced opening and closing of these will reduce the maintenance costs.

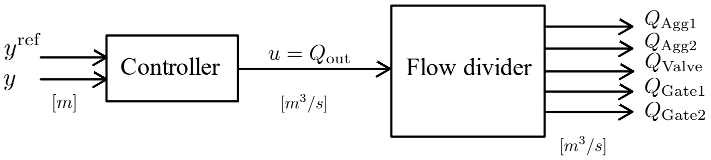


Figure 2: Current solution of level control

The control input calculated by the level controller is converted from flow to power based on the turbine's efficiency and drop height (pressure), and used as a setpoint in the turbine's controller/governor. A generic governor is shown in figure 3. The governor is made up of two control loops. The inner power control loop gets an error between measured and reference power and is fed to a PI controller via a droop gain $\alpha$. The PI then adjusts the turbine's guide vanes. The power reference is also fed forward to allow to set the power operating point. The power reference in this control loop is provided by the water-level controller.

The generators used in the hydropower plants are synchronous generators that need to run on the same frequency as the main power grid. The generator's frequency is directly proportional to the turbine's rotation speed. The main grid's frequency can vary slightly in day-to-day operations, because of changes in load on the grid. To keep the frequency as close as possible to a given setpoint the governor contains an outer frequency control loop. This control loop produces an error signal which is added to the error of the power control loop. In addition, the frequency control loop adds a derivative feed-forward loop as an aid to rapid response, when the network frequency changes quickly. [19]

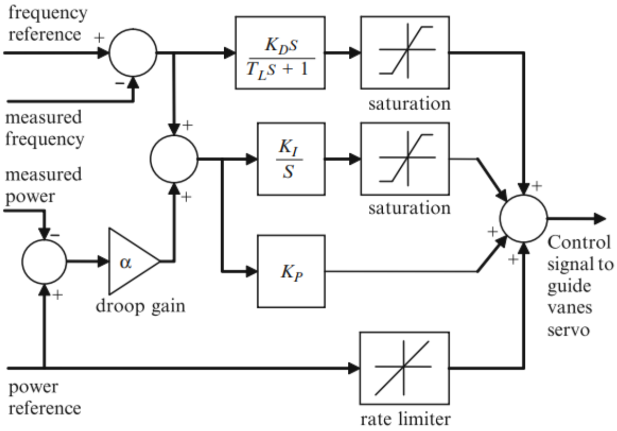


Figure 3: Generic PID governor for controlling power and frequency [19]

The effect of the two control loops can be easily seen by looking at the flow through the aggregates of a power plant, which is shown in figure 4. Figure 4 shows flow through aggregates at Rånåsfoss and Bingsfoss, which are two power plants that are used as an example system in this thesis. The flow is kept around a constant setpoint given by the level control. The noisy variations in the signal are caused by the frequency control loop that uses some of the flow to keep up the frequency on the main power grid. Because of how the governor adjusts the power operating point it is clear that ideally the power setpoint and thus flow through the power plant should be kept constant as much as possible. This will both limit the mechanical wear on the system and make the governor operate in steady-state mode. It can be seen that the water level controller at Rånåsfoss is doing a good job at keeping the flow constant in this time period, while the Bingsfoss controller struggles to find a stable control input. This is one of the problems that the controller presented in this thesis is trying to solve.

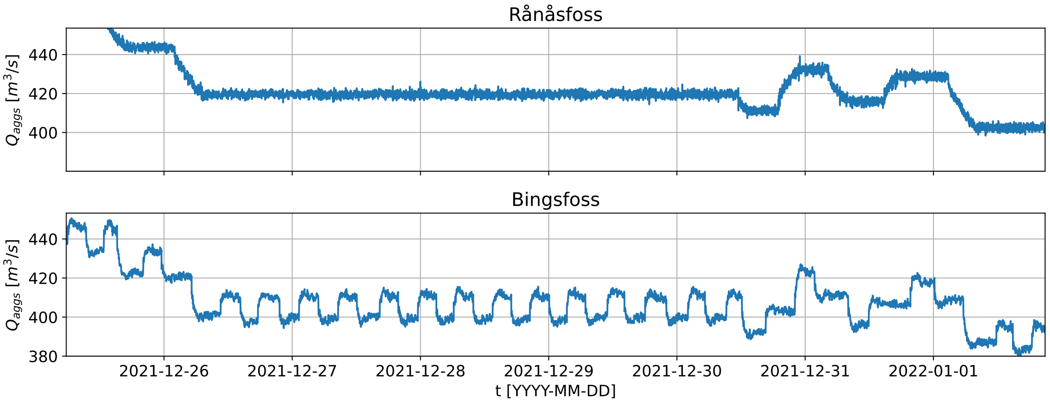


Figure 4: Flow through aggreagates at Rånåsfoss and Bingsfoss power plants

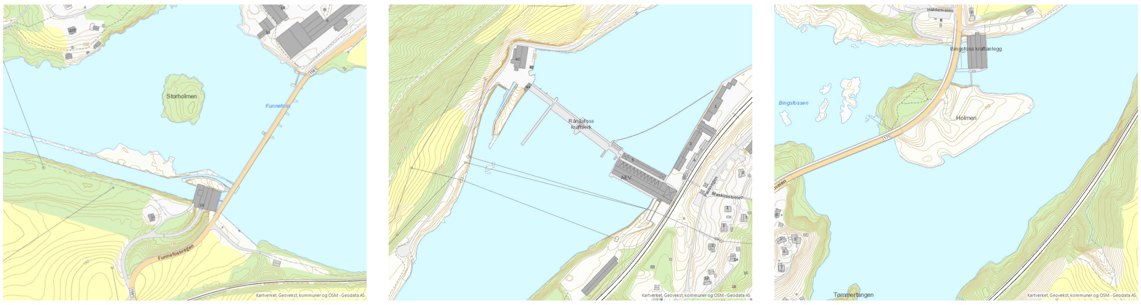
### 1.1.2 River system

To be able to more accurately test the developed methods, a river system has been chosen as an example case. The system has been proposed by Voith Hydro and is a river segment on river Glomma between Funnefoss and Bingsfoss. In this segment, there are three run-of-river power plants in a cascade. All power plants lay directly on the river, without any long intake tunnels. All the power plants are operated by Agder Energi. The most upstream plant is at Funnefoss, the next one is at Rånåsfoss, 22 km downstream from Funnefoss and the last one is at Bingsfoss, 5.5 km downstream from Rånåsfoss. The width variations between Rånåsfoss and Bingsfoss are also much bigger than between Funnefoss and Rånåsfoss, combined with the difference in length makes that the two segments have very different behavior. Map over the whole segment is shown in figure 6.

Inbetween Rånåsfoss and Bingsfoss is inflow from Mjøsa lake via the Vorma river, which adds great lateral inflow to the system. The size of the lateral inflow can be easily seen by looking at the aggregate dimensions in table 3. Note that the Rånåsfoss power plant consists of two powerhouses that lay on opposite riverbanks (see figure 5b). The power plant at Rånåsfoss can utilize more than twice the flow of Funnefoss. This additional water flow comes from the Mjøsa lake.

Table 3: Power plants and their properities

| Power house | Turbines | Total power output $[MW]$ | Max $Q_{\text{aggs}}$ $[m^3/s]$ |
|---|---|---|---|
| Funnefoss | 2 Kaplan bulb turbines | 40.0 | 439.86 |
| Rånåsfoss II | 1 Kaplant turbine | 44.8 | 412.54 |
| Rånåsfoss III | 6 Propeller turbines | 81.0 | 764.52 |
| Bingsfoss | 3 Kaplan bulb turbines | 32.4 | 733.95 |


Storholmen
Funnefoss
Funnefossvegen
Kartverket, Geovekst, kommuner og OSM - Geodata AS
Rånåsfoss kraftverk
AEV
Maskinistveien
Kartverket, Geovekst, kommuner og OSM - Geodata AS
Haldenveien
Bingsfoss kraftanlegg
Bingsfossen
Holmen
171
Tømmertangen
Kartverket, Geovekst, kommuner og OSM - Geodata AS

(a) Funnefoss (b) Rånåsfoss (c) Bingsfoss

Figure 5: Power plants along the river

## 1.2 Objectives

The main objective of this thesis is to develop a model predictive controller (MPC) and a moving horizon estimator (MHE), that can be implemented to control water levels at run-of-river hydropower plants. The new control solution must keep all the requirements imposed on that river. The new controller should have the ability to keep the control input constant for as long as possible while keeping the water level close to the setpoint. The solution should be general and possible to implement relatively easily on other river systems. A step in developing the controller is finding models that can accurately reflect real river behaviors. At the core of an MPC is a system model, therefore verifying model equations is a necessary step in creating the final solution. The end goal of this thesis was to test the controller on an actual power plant.

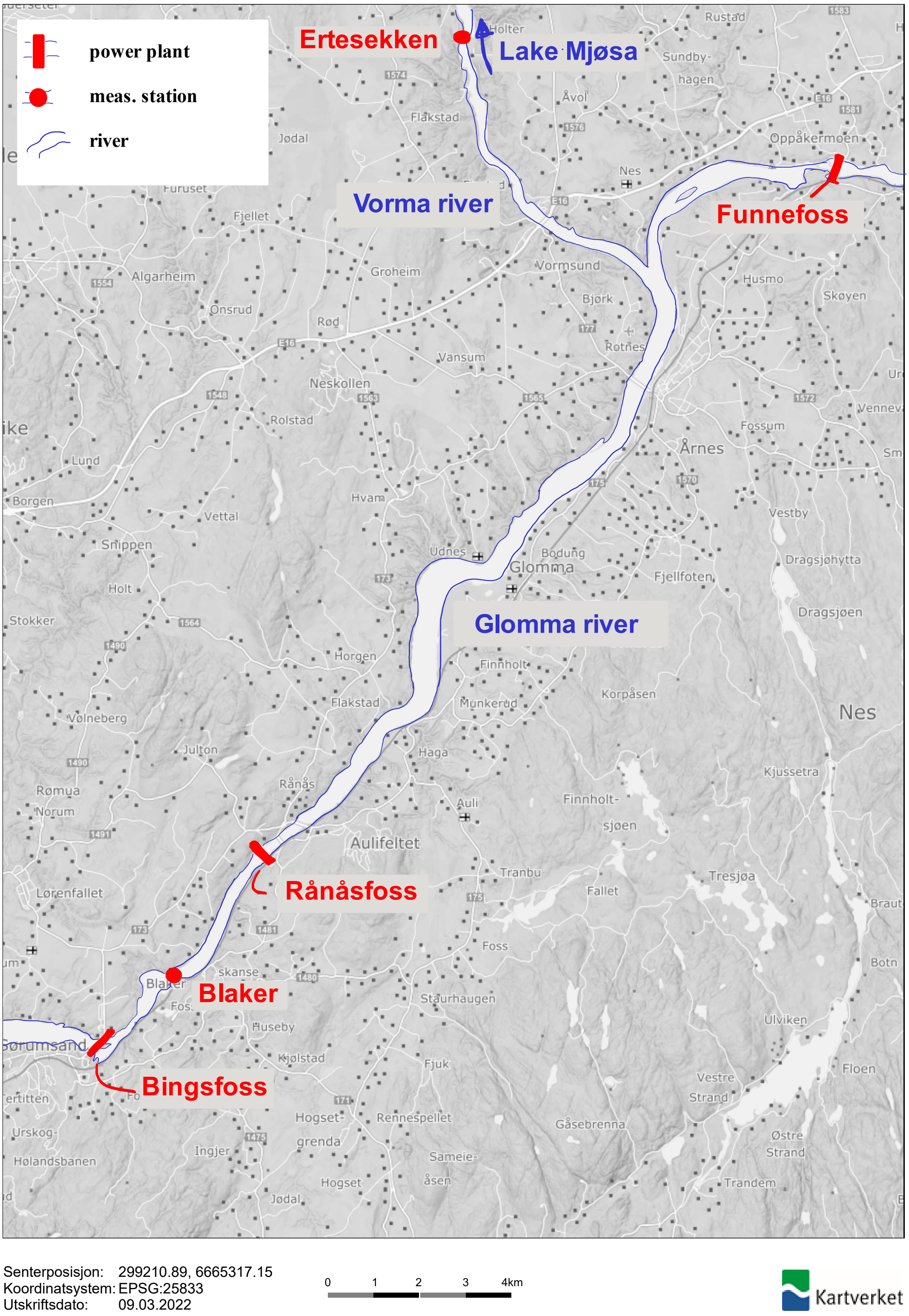
## 1.3 Approach

This thesis is based on the results of a pre-project about models that can be used to describe a run-of-river power plant system. The pre-project was written as a part of the TTK4551 course at NTNU, in autumn 2021. The project has shown promising results about St. Venant equations (SVEs) as a way to model river dynamics. Lack of data was the main limitation of the project, and the results have not been accurately verified. The first part of this thesis is therefore a continuation of work from the pre-project and verifying the equations by comparing them to the real data.

The information gathered about the used models comes mainly from the pre-project. The information about how the power plants operate currently has been provided by contact persons at Voith Hydro and Akershus Energi. These were important to understand the problem from a practical point of view, which was necessary to create a solution that can be used in a real system. The theory about the controller and estimator has been mainly found by online literature search using Google Scholar. I had also some basic knowledge about the MPC from previous courses at NTNU. However, the practical implementation and some advanced techniques were new to me, and I learned a lot of the techniques while writing his thesis. In this process, the inputs from the supervisor at NTNU were very helpful in connecting the theory with an actual solution.

All the experiments in this thesis have been done by simulating the developed models. These have been run using a Python library, GEKKO. Python has been chosen as a programming language for this thesis since it's free and accessible for scientific work. It also doesn't require a license as for instance MATLAB does and might therefore make it more attractive to try out new and unconventional solutions, that have been presented in this thesis. For data manipulation, I used mainly python libraries "SciPy" and "pandas", and Excel.

Data used in this thesis has been acquired from different sources. Access to data from power plants has been provided by Voith Hydro and Akershus Energi. Data from measurement stations that are not owned by Akershus has been collected from "NVE Sildre". Other data like terrain and river width measurements were acquired from digital maps at "NVE Atlas" and hoydedata.no.



Senterposisjon: 299210.89, 6665317.15
Koordinatsystem: EPSG:25833
Utskriftsdato: 09.03.2022

Figure 6: Map of the Glomma river

# 2 Theory

## 2.1 Saint-Venant equations

Saint-Venant equations (SVEs), also called shallow water equations, "are equations of unsteady flow for shallow water conditions. The equations cover the movement of water in rivers, canals, overland flow as well as circulation in lakes and oceans"[16]. The SVEs has been first presented by Berre de Saint-Venant in 1871. The fundamental assumptions underlying the development of SVEs as shown by Mahmood [16] are:

- Preassure distribution along vertical axis is hydrostatic, which means that the vertical acceleration is small.
- Friction losses in unsteady flow are not significantly different from those in steady flow.
- Velocity distribution across the wetted area does not substantially affect the wave propagation.
- The wave movement can be considered two dimensional with the effect of eventual difference of levels in cross sections negligible
- The average slope of the channel bottom is so small that $sin\alpha$ may be replaced by $tan\alpha$, and $cos\alpha$ as unity.

One dimensional SVEs can be written in follwing form, as shown by Mahmood [16]:

$$\frac{\partial A}{\partial t} + \frac{\partial (Av)}{\partial x} = Q_l \tag{3a}$$

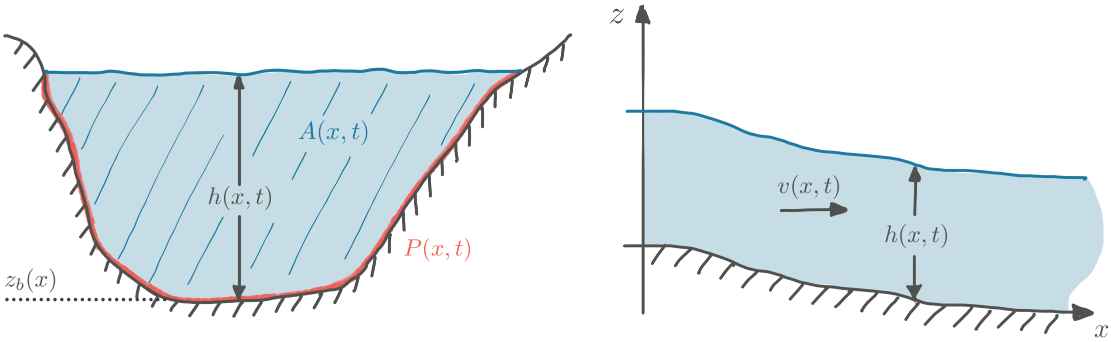
$$\frac{\partial v}{\partial t} + v\frac{\partial v}{\partial x} + g\frac{\partial h}{\partial x} + g(S_f - S_x) + \frac{q}{A}(v - v_l) = 0 \tag{3b}$$

where $S_x$, $S_f$ and $R$ are given by:

$$S_x = \frac{\partial z_b}{\partial x} \tag{4a}$$

$$S_f = \frac{|v|v}{(k_s R^{2/3})^2} \tag{4b}$$

$$R = \frac{A}{P} \tag{4c}$$



(a) River cross section

(b) River along the x axis

In the equations, the coordinate system is placed such that $x$ axis is in the direction of the primary flow, $y$ in the horizontal direction normal to the primary flow, and $z$ in the vertical direction.[16]

The equation 3a is the continuity equation and states that the net rate of flow into the volume equals storage change in the volume. This equation expresses the same dynamics as the mass balance equation 1. The equation 3b is the momentum equation and states that the net rate of momentum entering the element plus the sum of the forces acting on the element is equal to the rate of accumulation of momentum.[16]

Terms in the momentum equation can be divided into conservative terms and non-conservative terms. Conservative terms are local acceleration: $\partial v/\partial t$ and connective acceleration: $v(\partial v/\partial x)$. These terms conserve the momentum fluxes. The non-conservative terms are friction and gravity term: $g(S_f - S_x)$ and hydraulic pressure gradient: $\partial h/\partial x$. Those terms are capable of creating or destroying momentum (source and sink) [14]. The non-conservative terms describe forces that act on the fluid in $x$ direction. The gravitational term is the gravitational force acting on the fluid, simplified for a small slope. The friction term is derived from the Guackler-Manning formula. This contains a Strickler coefficient $k_s$ which varies according to the surface. In some literature it is also used Maning coefficient $n$ where, $k_s = 1/n$. The coefficient values can be found in multiple sources, the most common are coefficients decried by Chow [7].

## 2.2 Optimization

Mathematical optimization problems include three main components, an objective function, decision variables, and constraints. The objective function is a scalar function that describes a property that we want to minimize or maximize. [10] The goal of the optimization is to find decision variables that optimize the objective function. The variables are often restricted in some way by the constraints. [21] This background allows us to define an optimization problem, the nonlinear program (NLP):

$$\min_{z} J(z,\theta) \tag{5a}$$

$$\text{s.t.} \tag{5b}$$

$$c_m(z,\theta) = 0, \qquad m \in \mathcal{E} \tag{5c}$$

$$c_m(z,\theta) \geq 0, \qquad m \in \mathcal{I} \tag{5d}$$

The objective function $J$ takes the vector of decision variables $z \in \mathbb{R}^n$ and vector of parameters to the problem $\theta$ and projects them onto the real axis. The decision variables might be control trajectory parameters, intermediate states, slack variables etc. and the parameters can include initial states, parameters of reference trajectories, exogenous inputs etc. [15]. $\mathcal{E}$ and $\mathcal{I}$ are disjunct index sets for the equality and inequality constraints $c$ respectively. Due to the constraints, the decision variables $z$ may be selected from a subset of $\mathbb{R}^n$. This subset is called the feasible region and is defined by: [10]

$$\Omega = \{z \mid (c_m(z,\theta) = 0, m \in \mathcal{E}) \wedge (c_m(z,\theta) \geq 0, m \in \mathcal{I})\} \tag{6}$$

The feasible set is a result of the dynamics of the system and all design parameters of the NLP. It is desired to make this set as large as possible while fulfilling the physical and operational constraints of the system. [15] The objective function and the constraints can be chosen in such a way that the optimization problem can be used as a controller or a state estimator as it is shown in the next sections.

## 2.3 Notation and system formulation

The next sections use a system model to formulate the MPC and the MHE optimization problem. The system formulation is a general nonlinear system that can be described using the equations:

$$\dot{x}(t) = f(x(t), u(t), d(t), w(t)) \tag{7a}$$

$$y(t) = h(x(t), u(t)) + v(t) \tag{7b}$$

where $x$ are states, $u$ are control inputs, $d$ are known disturbances, $w$ ar unknown model errors and disturbances, $y$ are measurements, and $v$ unknown additive measurement errors. The system is discretized at time instants $t_k = k$, and the notation is simplified such that $x(t_k) = x_k$. The disretized system takes following form:

$$x_{k+1} = F(x_k, u_k, d_k, w_k) \tag{8a}$$

$$y_k = H(x_k, u_k) + v_k \tag{8b}$$

Norm notation is often used in describeing the objective function presented in this thesis. For a vector $x \in \mathbb{R}^N$ a $\ell_2$ norm of $\|x\| = \|x\|_2 = \sqrt{x^T x}$, and $\ell_1$ norm of $\|x\|_1 = |x_1| + \ldots + |x_N|$. A weighted norms with a symmetric matrix $W \succeq 0$ is given as $\|x\|_W = \sqrt{x^T W x}$. In all formulations in this thesis the weight matrix $W$ is a diagonal matrix. Vectors $x_1, x_2, \ldots, x_N$ are stacked into one large vector $x$ by the notation $x = \text{col}(x_1, x_2, \ldots, x_N)$. Estimated values are denoted with a hat symbol as: $\hat{x}$

## 2.4 Model predictive control

Model predictive controller (MPC) is a form of control in which the current control action is obtained by solving on-line, at each sampling instant, a finite horizon open-loop optimal control problem, using the current state of the plant as the initial state. The optimization yields an optimal control sequence and the first control in this sequence is applied to the plant. An advantage of an MPC is its way of handling constraints. Nearly every application imposes constraints; actuators are naturally limited in the force they can apply, and safety limits dictate the system's behavior. There is however lack of control methods that can handle the constraints, which results in the use of ad hoc methods. Another advantage of an MPC is the ability to control nonlinear plants, for which off-line computation of a control law usually requires the plant dynamics to possess a special structure. [18]

The specification of the MPC control functionality and dynamic performance is provided through the cost function and the constraints[15]. A typical MPC cost function consists of a stage cost function $L$ and a terminal cost $S$. A general MPC can be described with following optimization problem:

$$\min_{z} \sum_{i=0}^{N_c} L(x_i, u_i) + S(x_{k+N_c}) \tag{9a}$$

subject to

$$\begin{aligned}
x_{i+1} &= F(x_i, u_i, d_i, w_i) && \forall i = k \ldots k + N_c - 1 && \text{(10a)}\\
y_i &= H(x_i, u_i) && \forall i = k \ldots k + N_c && \text{(10b)}\\
x_0, d_0, w_0 &= \text{given} && \forall i = k \ldots k + N_c && \text{(10c)}\\
c_m(z) &= 0 && m \in \mathcal{E} && \text{(10d)}\\
c_m(z) &\leq 0 && m \in \mathcal{I} && \text{(10e)}
\end{aligned}$$

### 2.4.1 Stage cost

MPC can be designed to handle different types of control problems. A common problem, which is the main focus of this thesis is reference tracking. One of the most widely used objective functions to solve this problem is the $\ell 2$ type cost function that minimizes the squared difference between a given reference setpoint $y^{\text{ref}}$ and the measured values $y$ and penalizes change in the control inputs $\Delta u$.

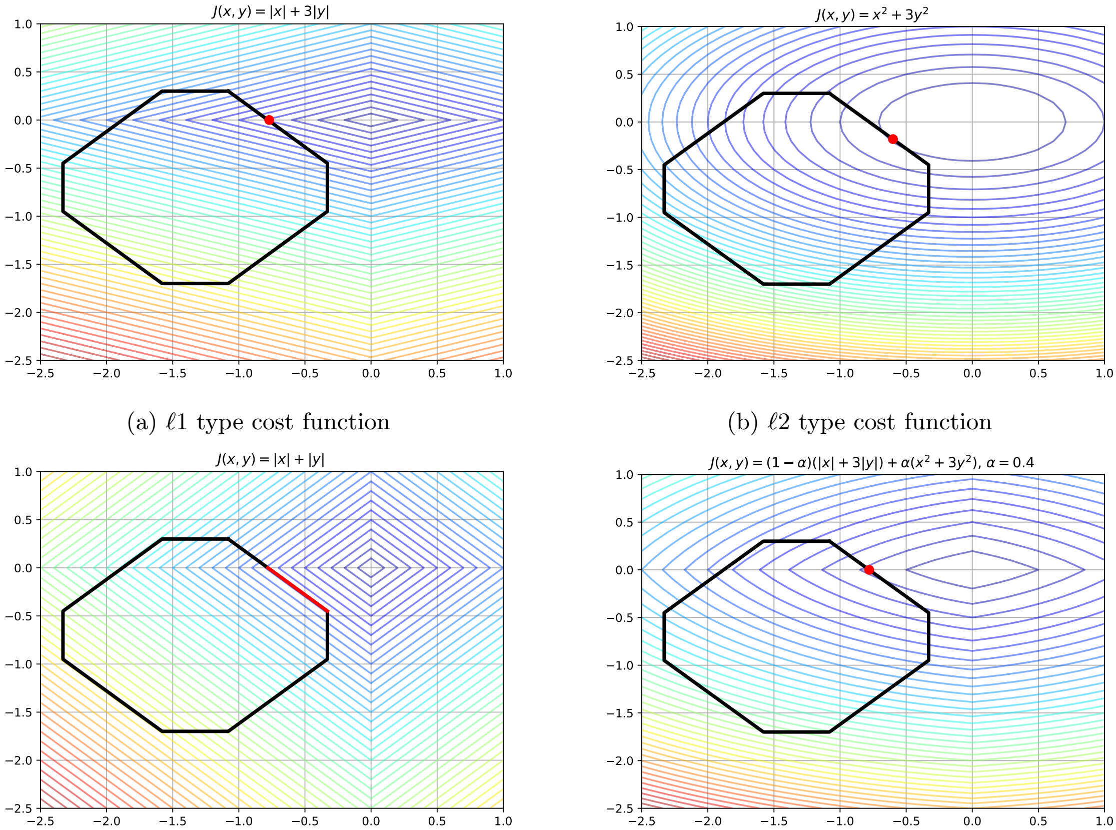
$$L(x_i, u_i) = \|y_i - y_i^{\text{ref}}\|^2_{W_{\text{SP}}} + \|\Delta u_i\|^2_{W_{\Delta u}} \tag{11}$$

A desirable property of a control system in a hydropower plant is to keep the control input constant for as long as possible. One way of achieving such behavior is the use of $\ell_{\text{asso}}$-MPC, which uses $\ell 1$ regularized least-squares in the objective function. The stage cost proposed for a $\ell_{\text{asso}}$-MPC in [11] can be written as follows:

$$L(x_i, u_i) = \|y_i - y_i^{\text{ref}}\|^2_{W_{\text{SP}}} + \alpha \|\Delta u_i\|^2_{W_{\Delta u}} + (1 - \alpha)\|\Delta u_i\|_{1, W_{\Delta u}} \tag{12}$$

This formulation is called elastic net in [11], and introduces a variable $\alpha \in [0, 1]$, which can be used to control the trade-off between smoothness of the cost ($\ell 2$ norm) and the sparsity of the solution ($\ell 1$ norm). Compared to the standard expression in [11], the $u$ has been replaced with $\Delta u$, as the goal is to minimize the change rate of the control input, not the magnitude.

How the $\ell 1$ regression imposes the sparsity of the solution, can be illustrated with an example. The example presented here is a summarized version of the one shown in [11]. In figure 8 different cost function has been plotted, with the same linear constraints marked with black line. The optimal solution within the constraints has been marked with red.



(a) $\ell1$ type cost function

(b) $\ell2$ type cost function

(c) $\ell1$ type cost function with multiple solutions

(d) $\ell_{\text{asso}}$ type cost function

Figure 8: Effects of different objective functions on the solution

By comparing figure 8b and 8a, one can clearly see that $\ell1$ cost function prioritizes the solutions where one of the variables is zero, while $\ell2$ cost function will chose solutions that are closest to the origin. There can however appear some unwanted behaviors if a pure $\ell1$ cost function is used. It is illustrated in figure 8c. Here the contour lines are parallel with the constraints. It means that the optimization problem does not have a unique optimum, as all points along the marked red line are an optimal solution. This can cause chattering in the control signal, which is the opposite of what we wish to achieve. To guarantee a unique solution and preferably a sparse solution, these two ideas are combined into a $\ell_{\text{asso}}$ cost function. As shown in figure 8d the $\ell_{\text{asso}}$ cost function achieves exactly that. $\ell_{\text{asso}}$ cost function is smoother than the pure $\ell1$ function, which means that small perturbations of the cost and constraints have a smooth effect on the solution. It keeps however the non-smooth vertices of $\ell1$ cost function, which is the key to finding a sparse solution. $\ell_{\text{asso}}$-MPC can also achieve the same noise rejection as LQ-MPC, which is superior to the $\ell1$ case. These properties are described in more detail in [11].

### 2.4.2 Terminal cost

The terminal cost $T$ is chosen to approximate the cost-to-go, i.e $S(x_{N_c}) \approx \sum_{k=N_c}^{\infty} L(x_i, u_i)$ such that the total cost function approximates an infinite horizon cost. The cost-to-go is generally hard to compute and simple approximations are usually chosen. One such approximation is terminal set constraints of the type $x_i \in \Omega$ that ensures that the state is regulated close enough to the setpoint such that after $N_c$ steps it is a priori known that there exists a feasible and stabilizing controller that will ensure that $x_i, i > N_c$ never leaves $\Omega$ and eventually goes asymptotically to the set point. Another method is terminal equality constraints of type $x_{N_c} = y^{\text{ref}}$, that ensures convergence in a finite time. This implies that the cost after timestep $N_c$ is zero. [15]

## 2.5 Moving horizon estimation

The states of a system are most often not directly measured, due to physical or practical implications. To solve this problem the states of the system need to be estimated. The state estimation problem is to determine the current states $x$ based on a sequence of past and current measurements $y$ at discrete time instants, and the use of a dynamic model. [15] It is assumed that the system can be described as a discrete-time nonlinear model using equation 8.

The MHE is formulated as a non-linear optimization problem, that can be solved using numerical optimization. A general MHE optimization problem can be described as follows:

$$\min_{z} \sum_{i=k-N_e}^{k} L(v_i, w_i) + Z_{k-N_e}(x_{k-N_e}) \tag{13a}$$

subject to

$$\begin{aligned} x_{i+1} &= F(x_i, u_i, d_i, w_i) && \forall i = k - N_e \dots k-1 && (14a)\\ y_i &= h(x_i, u_i) && \forall i = k - N_e \dots k && (14b)\\ y_i, u_i, d_i &= \text{given} && \forall i = -k - N_e \dots k && (14c)\\ c_i(z) &= 0 && i \in \mathcal{E} && (14d)\\ c_i(z) &\leq 0 && i \in \mathcal{I} && (14e) \end{aligned}$$

where $L(v_i, w_i)$ is a stage cost, and the $Z(x_0)$ is arrival cost. The cost function is minimized over a time horizon $N_e$ subject to the systems model equations, and equality and inequality constraints. The additional constraints in the MHE are a way to include additional information about the system, that are not described in the model equations, for instance, the upper and lower limits of a variable.

### 2.5.1 Least squares cost function

The stage cost penalizes the model disturbance $w_k$ and measurement error $v_k$,which are the errors in the model equations 8. The measurement error can then be reformulated as: $v_k = y_k - h(x_k, u_k)$. There are usually infinite choices of $w_k$, $v_k$ and initial states $x_0$ that are consistent with a given measurements $y_k$. It is therefore necessary to establish criteria for calculating the best estimates of these unknown variables. One common way of doing this is by using the least-squares method. This way we are trying to obtain the "best fit" of the state trajectories to the observations subject to the model. [22]

$$L(v_i, w_i) = L(y_i - h(x_i, u_i), w_i) = \|y_i - h(x_i, u_i)\|^2_{Q^{-1}} + \|w_i\|^2_{R^{-1}} \tag{15a}$$

Such a choice of the objective function seems natural as we are trying to minimize the difference between predicted and measured values, which is described by the first term. In addition, we want to minimize the solver's use of unknown disturbances $w$ to fit the data, which is described by the second term.

There is also a probabilistic view on the least-squares formulation that can give more insight into the objective function, described by Robertson [22]. All of the information about the states that is contained in the measurement sequence can be written in terms of the conditional joint density function $p(x_0, \dots, x_{N_e} | y_0, \dots, y_{N_e})$. Using the Bayes' theorem the conditional density can be described using densities of $x_0^e$, $v_i$, and $w_i$, which make the problem easier to work with. Term $x_0^e$ describes the error in the initial values. Then by assuming that $x_0^e$, $v_i$ and $w_i$ are uncorrelated, zero-mean Gaussian sequences with covariance $P$, $Q$, $R$ respectively, the joint conditional probability density function can be written as:

$$p(x_0, \dots, x_{N_e} | y_0, \dots, y_{N_e}) = C\exp[-0.5\|x_0^e\|^2_{P^{-1}} \sum_{k=0}^{N_e} + \|v_i\|^2_{Q^{-1}} + \|w_i\|^2_{R^{-1}}] \tag{16a}$$

where $C$ is a constant independent of $x$. Our goal is to maximize the conditional probability density, which corresponds to minimizing the least-squares equation 15a.[22] Because of this connection we can see that by choosing the weight matrices $W$ equal to the inverse of covariances of measurements and disturbances, the MHE tends to the maximum likelihood estimator for the most likely trajectory. [27] This provides a method for choosing weight matrices based on knowledge about the signals, instead of arbitrary values.

### 2.5.2 Arrival cost

A term $Z(.)$ is also included in the cost function (equation 13). Its purpose is to include information from data before the start of the horizon used in the estimation. Using only data from the horizon is a weakness especially when the information content in the data is low due to noise or other uncertainty. In other words, the estimation formulation contains no other mechanism to introduce filtering of noise or regularization than to increase the horizon $N_e$. This also increases the computational complexity of the optimization problem and may still be insufficient. [15]

There are several ways of choosing the term $Z(.)$. The simplest way is to set $Z(.) = 0$, which means that no information about data prior to the horizon is included. It comes however with the drawbacks described above. One way to formulate the term is by penalizing the deviation from the prior estimate. Another more complex formulation include estimating the arrival cost $Z_{k-N_e}(x_{k-N_e}) = \sum_{i=0}^{k-N_e} L(v_i, w_i) + \Gamma(x_0)$ using Kalman filter. References to more detailed information can be found in [15]. These will not be described here in more detail as they have not been explicitly implemented in this thesis, since the software used had some built-in functions that achieve this.

### 2.5.3 Parameter estimation

The formulation of the MHE can also include parameter estimation in the optimization problem. The arguments presented here are based on those presented in [15]. A common approach to the joint state and parameter estimation problem is to augment the state space with constant parameters. Assuming a vector of constant parameters $\theta^*$ appears in the model equations:

$$\xi_{i+1} = F(\xi_i, u_i, d_i, \gamma_i, \theta^*) \tag{17a}$$

$$y_i = H(x_i, u_i) + v_i \tag{17b}$$

with new notation where $x_i$ is the state and $\omega_i$ is the disturbance. An augmented state-space model assumes that the parameters are constant or slowly time-varying by the following model of the unknown parameter vector:

$$\theta_{i+1} = \theta_i + \rho_i \tag{18}$$

Combining these leads to:

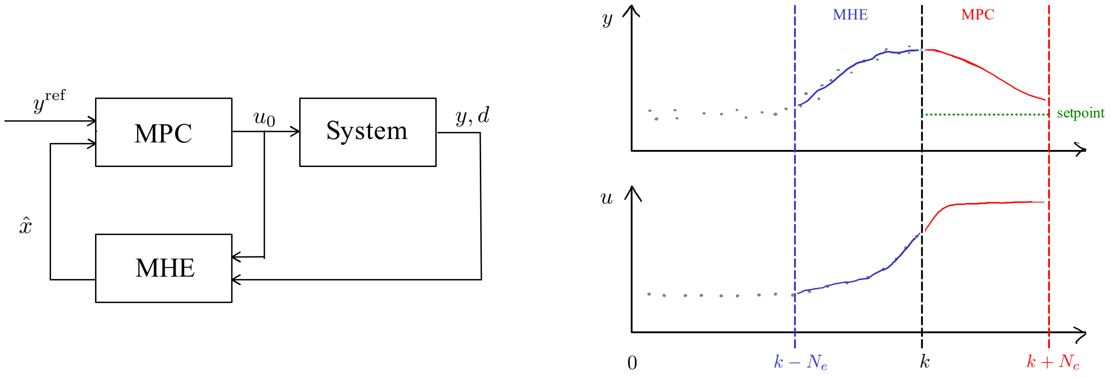
$$\begin{bmatrix} \xi_{i+1} \\ \theta_{i+1} \end{bmatrix} = \begin{bmatrix} f(\xi_i, u_i, d_i, \gamma_i, \theta) \\ \theta_i + \rho_i \end{bmatrix} \tag{19a}$$

$$y_k = h(x_i, u_i) + v_i \tag{19b}$$

With augmented state $x = \text{col}(\xi, \theta)$ and augmented disturbance vector $w = \text{col}(\omega, \rho)$ we observe that these equations are in the assumed form (equation 8), such that MHE algorithm formulation can be applied without any modifications.

## 2.6 Combining MPC and MHE

How the MPC and the MHE er connected into a control loop is shown in figure 9a, and is very similar to the control loop of any other observer and controller. The formulations of an MHE and an MPC can be seen as dual problems, and work in a similar manner. Figure 9b shows how the MPC and MHE work at a given time instance $k$. At a time instant $k$, the MHE uses the measurements and control inputs from $k - N_e$ to $k$ to estimate states in that time period. If arrival-cost is used its goal is to include information about the states before time $k - N_e$. From the results returned by the MHE, the estimates from the latest timestep $k$ are sent to the MPC. Then the MPC finds the control inputs that will result in an optimal trajectory in the time window $k$ to $k + N_c$. In this case, the optimal trajectory is one that achieves the smallest possible difference between the measurements and the setpoint with small changes in the control inputs. If the terminal cost is used its goal is to assure that the system ends up close enough to the setpoint after time $k + N_c$. Then the first of the calculated control inputs are applied to the system.



(a) Structure of the control loop

(b) MPC and MHE at a time instant k

# 3 Methodology

## 3.1 River model implementation

To be able to simulate the SVEs equations the model is discretized using finite volume method. The river is divided into $N$ cells, also called control volumes, which allows us to discretize the model along the $x$ axis. Each cell holds physical values (velocity and/or area) that are time-varying. Output fluxes from the previous cell are input fluxes for the next cell. This allows us to approximate spatial derivatives using finite differences along the $x$ axis. This results in $N$ ordinary differential equations (ODEs) that can be solved using the preferred ODE solver. This method presented below is based on the method described by Vytvytskyi et al. [28].

There are several ways to create a cell grid to discretize the equations. One way is to use a collocated grid, where all variables are stored at the same grid position. A disadvantage of this method is the checkerboard problem that arises if it is used to simulate an incompressible flow. The problem is that the velocity in a cell does not depend on the pressure in the cell, only on pressures in the neighboring cells. The problem is often illustrated with a pressure field with alternating pressure values like on a checkerboard. With such a setup the discretized gradient becomes zero and gives the same results as a uniform pressure field. [9, p.616] In practice this method creates unwanted oscillations in the simulation. In the first simulations of SVEs, a collocated grid was used. This method made the simulation become easily unstable and another solution needed to be found.

An alternative to a collocated grid is a staggered grid. This approach uses two grids that are placed with an offset of a half cell length to each other. This method is similar to Arkawas C-grid [3] that is used in two-dimensional cases. The main advantage of the staggered grid is "that it allows for the pressure difference between two adjacent grid points to be the driving force for the velocity component located between these grid points"[9], and it eliminates the checkerboard problem.

The staggered grid method eliminated the unstable behavior that was observed when using the collocated grid. The grids are implemented as follows. The first grid stores values for cross-section areas (A-grid) and has $N+1$ cells. The second grid stores values for velocities (v-grid) and has $N$ cells.

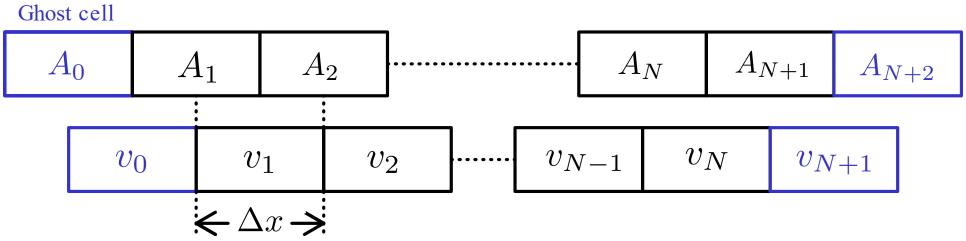


Figure 10: Staggered grid

An arbitrary position on the grid is denoted with index $n$. Derivatives along the x-axis can be approximated by using the values at the boundary of each cell ($y_{n+1/2}$, $y_{n-1/2}$) to calculate the finite differences for distance $\Delta x$. The value at the cell boundaries are approximated as $y_{n+1/2} = 0.5(y_n + y_{n+1})$. Since a staggered grid is used, the values stored at v-grid, will describe velocities ($v$) at the boundaries of cells on A-grid and vice versa. Therefore it is only necessary to approximate boundary values for $A$ at A-grid and boundary values for $v$ at v-grid. Using those assumptions the equations can be described as follows:

$$
\begin{aligned}
\frac{\partial A_n}{\partial t} &= -\frac{A_{n+1/2}v_n - A_{n-1/2}u_{n-1}}{\Delta x} + Q_{l,n} && n = \{1 \dots N+1\} && \text{(20a)}\\
\frac{\partial u_n}{\partial t} &= -u_n\frac{u_{n+1/2} - u_{n-1/2}}{\Delta x} - g\frac{h_{n+1} - h_n}{\Delta x}\\
&\quad - g\left(\frac{\tanh(Kv_n)v_n^2}{(k_s(R_n)^{2/3})^2} - \frac{\partial z_b}{\partial x}\right) - \frac{Q_{l,n}}{A_{n+1/2}}(v_n - v_{l,n}) && n = \{1 \dots N\} && \text{(20b)}\\
\frac{\partial Q_{l,n}}{\partial t} &= \dot{Q}_{l,n} && n = \{1 \dots N\} && \text{(20c)}
\end{aligned}
$$

Compared to equations presented in the theory an additional equation has been added where lateral inflow is described with an extra variable $\dot{Q}$. This is done because the lateral inflow is a disturbance that is going to be estimated and used in the MPC to predict future behaviors. By using a derivative to describe this term, the whole system can still be described using general equations 7 presented in the theory. It will also make it easier to implement freezing the derivative of the variable in some situations, which are described in more detail in section 3.5.4.

Another change from the equations presented in the theory is the term $|v_n|v_n$, that has been replaced by $\tanh(Kv_n)v_n^2$. The absolute value is not continuously differentiable which may create problems in a optimizer, since optimizers use often gradients to find the optimal solutions. The original term can be expressed as $\text{sgn}(v_n)v_n^2$, and the sign function can be approximated with a continuously differentiable $\tanh(.)$ function. The new formulation includes a constant $K$, which can be used to adjust steepness of the tanh function. The variable $K$ should be chosen such that the tanh approximates the sign function for the range of values of $v$ that will occur in the simulations. In the model implementation, the value has been chosen to be $K = 100$.

Boundary conditions are implemented using ghost cells, that are used as padding for the grids. The ghost cells are removed before returning the values of the differential equations to the solver. In the pre-project there were defined two types of ghost cells depending on the type of the boundary condition.

- Free out/in-flow boundary condition: $A_0$, $A_{N+2}$ is set to the same value as neighboring cell, $v_0$, $v_{N+1}$ is set to average of velocity of $k$ neighboring cells. Using average to dampen noise that can occur at the boundaries. This method does not work well for flat slopes at the outflow boundary.
- Controlled out/in-flow boundary condition: $A_0$, $A_{N+2}$ is set to the same value as the neighboring cell, $v_0$ and $v_{N+1}$ are calculated from specified $Q_{\text{in}}$ or $Q_{\text{out}}$ using $v = Q/A$.

In this implementation all boundary flows are either controlled or measured, and only the "controlled out/in-flow boundary condition" are used.

Measurements used in the model are headwater and tailwater level measurements. These are measured in meters above sea level. The model developed here uses cross-section area $A$ as a way to describe the amount of water at each river segment. The areas need to be converted to water height above the river bed using function $h(A_n, n)$. Then terrain level $z_b(n)$ is added, such that measurements describe level above the sea level. This results in the following equation for the measurements:

$$y = h(A_n, n) + z_b(n) \tag{21a}$$

where $n = 1$ for measurements at the tailwater and $n = N + 1$ for measurements at the headwater.

### 3.1.1 River geometry

To be able to use the river model presented in this thesis, it is necessary to describe the river bed's slope and form. These parameters are rarely measured accurately, which makes it more challenging to develop a model of a river. The easiest way to acquire data about the terrain slope along the river is to use digital maps. For this thesis, the elevation data along the river has been collected from hoydedata.no, which provides elevations along a chosen trajectory. The resulting data are often noisy and in some cases might include big spikes in elevation, that often correspond with the placement of bridges and powerlines crossing the river.

Some data manipulation is necessary to use the river bed measurements in the model. The terrain is resampled such that the elevation measurements correspond to the number of elements in the spatial grid. This is done by calculating the average elevation for each river grid cell, which also filters away the noise. This gives elevations that can be used in the model. An additional implementation of this method has also been developed, where the elevation is not averaged/filtered the boundary cells. For the upstream boundary, the most upstream elevation has been applied for the whole cell, and similarly, the most downstream elevation has been applied for the downstream boundary cell. The idea behind this is that elevation at those points is used to calculate the measured water levels, and averaging the elevation at the boundaries would change the elevation at the measurement point. The last method for implementing the river bed slope is by simply using a linear slope for the river segments. The result of the downsampling for the grid that was used in the implementation is shown in figure 11 The elevations for method 'linear' and 'resampled (unfiltered boundaries)' overlap at the boundaries, while the two 'resampled' methods overlap everywhere except at the boundaries.

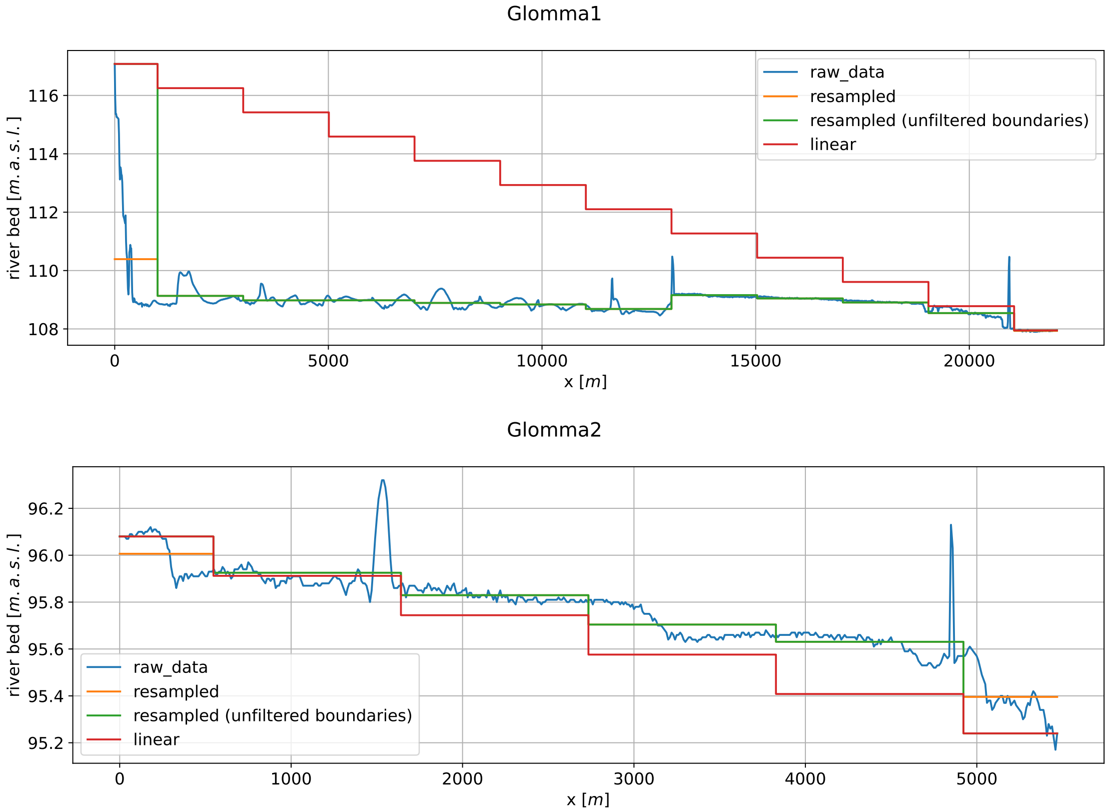


Figure 11: River bed profiles

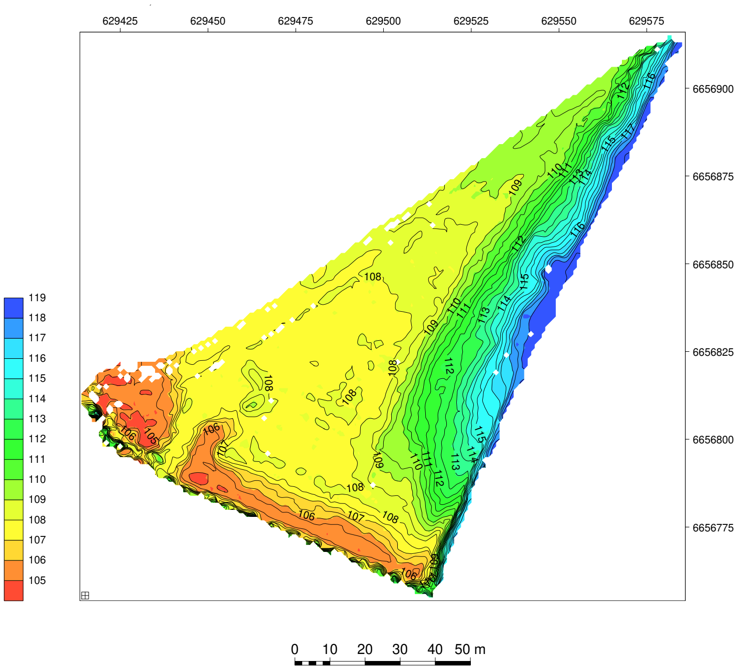


Figure 12: River bed scan at headwater of Rånåsfoss

One issue with the presented methods is that the digital maps give elevation at the terrain's surface, which is why the noise from bridges and power lines is visible. For a river, it means that the data show the elevation at the water's surface and not at the river's bed. This method will however still give a somewhat correct slope if we assume that the water level is in steady-state on the provided map. The river bed elevation is however necessary for the measurements equation 21. How to acquire this information will depend on what is available at the current power plant. For this case, there has been carried out river bed scanning at a small river segment at the headwater of the Rånåsfass power plant. Those were done as part of another project. Results from the scans can be seen in figure 12. In the figure, the lower-left boundary is the dam at Rånåsfoss. From the figure 12, we can read that the river bed is at about 108 m.a.s.l. The digital map data from terrain surface elevation at Rånåsfoss headwater is 119.5 m.a.s.l. That indicates that the river bed should be 11.5m lower than measured. This elevation offset is applied for all elevation measurements, as this is the best approximation that is available.

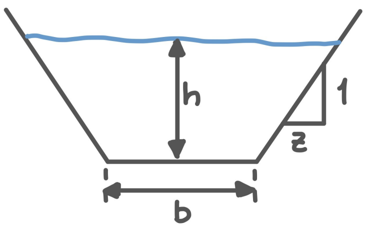


Figure 13: Trapezoid river cross-section

Another parameter needed to describe river geometry is the shape of the cross-section of the river bed. This is necessary to compute water level $h$ and wetted perimeter $P$ based on the cross-section area $A$. The trapezoid shape is a popular choice for modeling river bed geometry. In the pre-project, it has been concluded that the trapezoid shape is the most versatile and gives good results with a fast solving time. One advantage is that it can be tuned using only two parameters. Area of a trapezoid is given by $A(h) = (b + zh)h$. Using a quadratic equation we can find an expression for $h$. In the derived formulation 22a, an alternative quadratic formula (sometimes called "Citardauq formula" [1]) has been used, which allows to set $z = 0$ without resulting in division by zero. Trapezoid river geometry at a section $n$ can be described as:

$$h_n(A_n) = \frac{-2A_n}{-b_n - \sqrt{b_n^2 + 4z_n A_n}} \tag{22a}$$

$$P_n(A_n) = b_n + 2h_n\sqrt{1 + z_n^2} \tag{22b}$$

The main goal of the developed control loop is to keep a stable water level at the headwater of the power plants. By looking at the water level data at these points, it can be seen that the water level varies by just a few centimeters. The channel shape used in the simulation is set to rectangular, which should be a good approximation since the water level variations are small. This is achieved by setting $z = 0$ in the equations 22.

The river width $b$ can vary significantly along the river. In the current case, the river is very wide ahead of the Bingsfoss power plant compared to some other sections. This might create extra dynamics, and therefore the possibility to include varying river width $b(x)$ is added to the model. There is no easy way to get the measurements of the river width. The method used in this thesis is simply measuring different points along the river manually by using a digital map. Here arises the problem of how the measurements should be used to describe the width of each discretized grid cell. Several methods have been tried out to see if some might give better results than others. The first method assumes constant width along the whole river length, where the width is the average of all the measurements. Another method that uses the obtained width measurements averages the measured width for each grid cell. The last method is the same as the previous but the width of the first and last cell is the same as the first and last width measurement. The idea is that the water levels at these points are of the highest interest and should be described most accurately, same as in the method used for river bed profile.

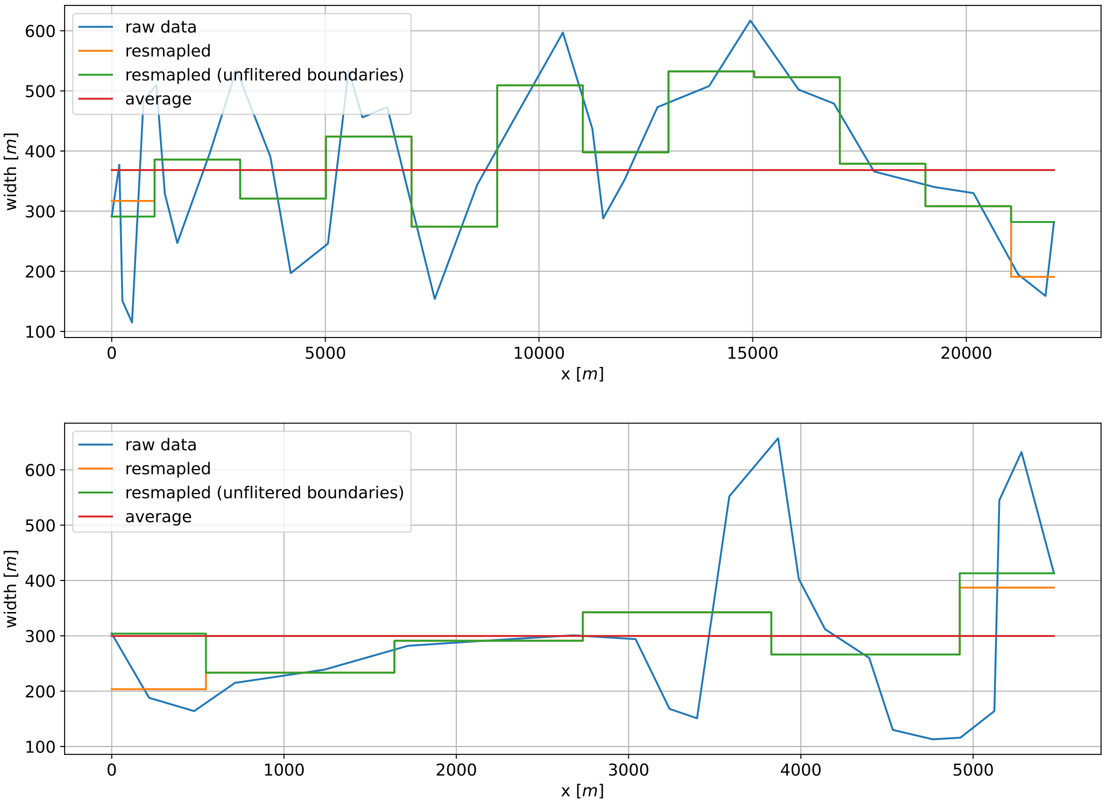


Figure 14: River width

## 3.2 Plant model implementation

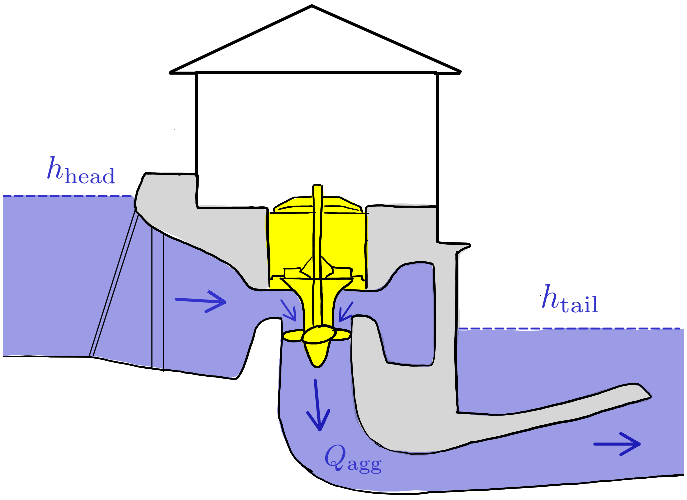


Figure 15: Schematic of a power plant with kaplan/propeller turbine

In this project, the power plant is modeled as a flow that is divided into different power plant components like aggregates, gates, and overflow. The power plant model does not contain any additional dynamics as the power plants that are discussed in this thesis lay directly on the river. The change in the flow between the power plant's intake, turbines, and output is very short, and the change in flow can be assumed instantaneous for simplicity. If the power plants had a long intake tunnel the dynamics of the tunnel should be included as it would create a considerable time delay, between flow at the power plant intake and aggregates. Model equations that can describe such power plants have been presented in the pre-project and are described by Nielsen [20].

$$Q_{tot} = Q_{\text{gates}} + Q_{\text{aggs}} + Q_{overflow} + Q_{\text{bias}} \tag{23a}$$

$$\frac{\partial Q_{\text{bias}}}{\partial t} = \dot{Q}_{bias} \tag{23b}$$

In addition to common plant components, a bias term has been added to the equations. The reason behind this is that in some cases the measured flows didn't add up. In a case where the water level is relatively constant, the flow into the river segment should be equal to the flow out of the same river segment. If the segment has a lot of unmeasured lateral inflow, one could also expect that the flow through the downstream power plant would be slightly bigger than through the upstream power plant. Based on the plots in the figure 16, it can be seen that this is not the case for the data from the powerplants discussed here. The flow through the downstream power plant is lower than through the upstream one for a long time period, while the water level stays relatively stable. One explanation for such behavior can be flow through the gates. Gates are big structures that are not controlled as accurately as the aggregates, and the flow calculated through these can quickly become slightly wrong. In the plots in figure 16, this is indeed the case where there is a lot of flow through the gates at Bingsfoss. This "missing" flow is less common when the total flow is lower and gates are closed, but some small deviations can however still be observed. As a percentage of the total flow, the amount of "missing" flow is similar for cases with the big and small total flow. But in the river model, it creates enough error for the estimator to struggle. This is especially when this disturbance can not be compensated with lateral inflow that is lower bounded at $0m^3/s$. To solve this problem a $Q_{\text{bias}}$ term has been added, as a slow varying disturbance.

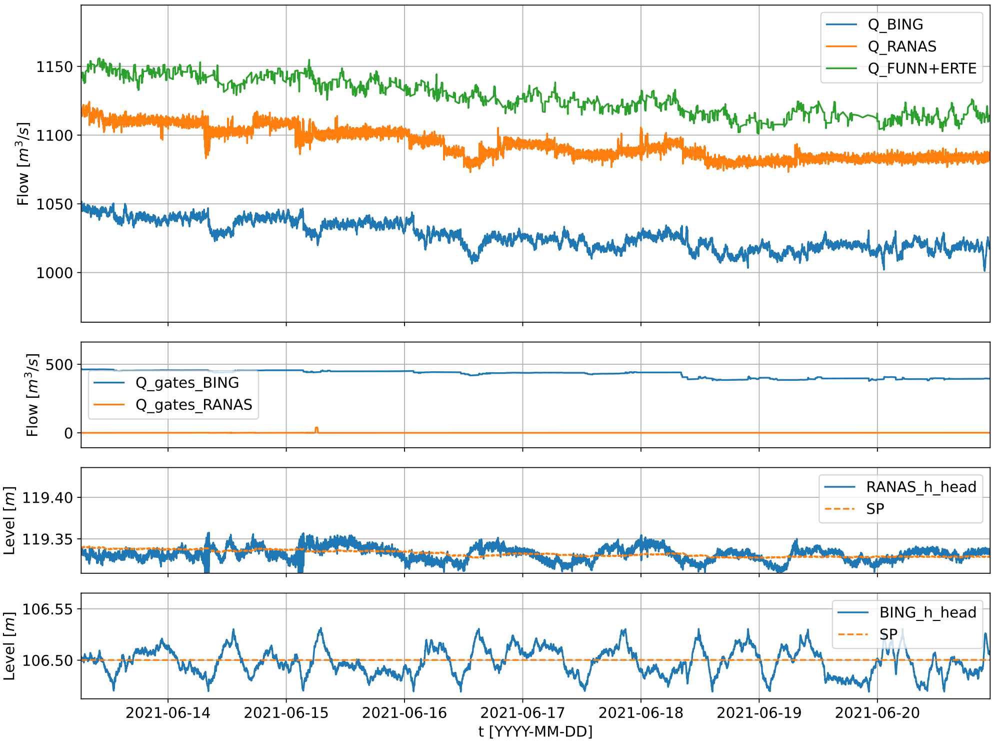


Figure 16: caption

## 3.3 System choice

The power plants Funnefoss, Rånåsfoss, and Bingsfoss have been proposed as a potential testing case for the MPC. A lot of data from those power plants was made available and the system needed to be limited to a case that suited the testing process. The system could be made more or less complex by choosing which river segment is to be considered, as well as what measurements to be assumed available. For this thesis the system to be controlled has been chosen as follows: the headwater level at Rånåsfoss and Bingsfoss is to be controlled to a setpoint by varying the flow through these power plants. This is the same as the objective of currently implemented water level controllers in the power plants. The lateral inflow from the Mjøsa lake is assumed unknown to the control system. There is however a measurement station at Ertesekken, which's measurements should be somewhat similar (but delayed due to travel distance) to the lateral inflow into the Glomma river. This can be used to compare the estimated lateral inflow with the measured one in the analysis. The flow out Funnefoss power plant is measured and assumed known by the control system. By choosing the system this way, different aspects of the control system can be tested such as estimation of unknown disturbance, utilizing information about measured disturbance, control of two systems in a cascade, and model accuracy on two physically different river segments.

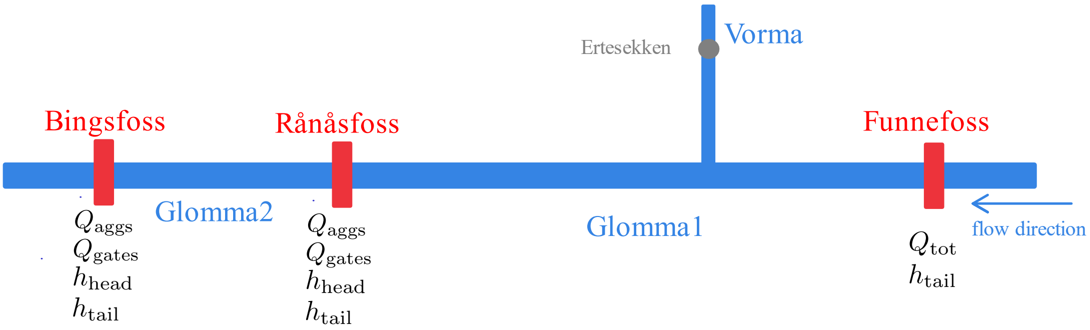


Figure 17: River schematic

## 3.4 Model implementation

When all the individual parts of the system are described, they need to be connected to form the whole system. To achieve that the river from Funnefoss to Bingsfoss is divided into parts that are limited by the power plants. It results in two segments, one between Funnefoss and Rånåsfoss, called from now on Glomma1; and another segment from Rånåsfoss to Bingsfoss, called Glomma2. Each of these segments is modeled using separate SVEs.

The uppermost inflow to the Glomma1 segment is the outflow from the Funnefoss power plant. This plant is not to be controlled on this implementation, but the outflow measurement from it is known. The other inflow into the Glomma1 segment is the flow from the tributary Vorma river that connects Glomma and Mjøsa. Inflow from the tributary is assumed unknown and this lateral inflow is modeled as an unknown disturbance in the model. The Vorma flows into the Glomma about 13km downstream Funnfoss power plant and is added to the corresponding grid cell in the SVEs. The outflow from Glomma1 is the intake of the Rånåsfoss power plant. The Rånåsfoss power plant is to be controlled and modeled using equations in section 3.2. This power plant is also a coupling point between the segments Glomma1 and Glomma2. Therefore the variables are set such that the outflow from Glomma1 is equal to the inflow to the Rånåsfoss plant, and the outflow from Rånåsfoss is equal to the inflow to Glomma2. The outflow from segment Glomma2 is the Bingsfoss power plant, which is a power plant that is controlled and modeled. The segment Glomma2 does have a little stream 4km from the Rånåsfoss, which has been modeled as the lateral inflow. It is very small compared to the total flow but has been added as an DOF for this segment.

## 3.5 Controller implementation

### 3.5.1 LASSO formulation

For the controller the $\ell_{\text{asso}}$-MPC formulation (section 2.4.1) has been used:

$$\min \sum_{i=1}^{N_c} \|y_i - y_i^{\text{ref}}\|^2_{W_{\text{SP}}} + \alpha\|\Delta u_i\|^2_{2,W_{\Delta u}} + (1-\alpha)\|\Delta u_i\|_{1,W_{\Delta u}} \tag{24}$$

s.t.

$$\begin{aligned} x_{i+1} &= F(x_i, u_i, d_i, w_i) && \forall i = k \dots k + N_c - 1 && (25a)\\ y_i &= H(x_i, u_i) && \forall i = k \dots k + N_c && (25b)\\ \Delta u_i &= u_i - u_{i-1} && \forall i = k \dots k + N_c && (25c)\\ x_k, d_k, w_k &= \text{given} && && (25d) \end{aligned}$$

where $y$ is the system measurements. The tailwater measurements have the corresponding weights set to zero since the tailwater is not controlled. The $u$ in the equations is chosen such that $u = Q_{\text{aggs}} + Q_{\text{gates}}$. This is because the MPC has no specifications on how the flow should be divided, so the $u$ is set to the total controllable flow.

Formulation of the $\ell_{\text{asso}}$-MPC includes $\ell 1$ norm of the control input, which's calculation uses absolute value function. The absolute value function is non-smooth and not continuously differentiable, which makes it problematic as most optimization techniques use Jacobians to find the optimal solution. To be able to use the absolute value in the objective functions, it needs to be reformulated. One way is to use mixed-integer programming to define the absolute value. This will require using a solver that supports integer values. Such solver is available in GEKKO, it is however generally slower than the IPOPT solver that is used. In some cases, the absolute value can be relatively easily expressed using continuous algebraic equations, without integer values. The continuous formulation can be written in the following way as shown in [6].

$$\begin{aligned} |u_i| &= u_i^{abs} && (26a)\\ u &\leq u_i^{abs} && (26b)\\ -u &\leq u_i^{abs} && (26c) \end{aligned}$$

The absolute value is replaced with a new variable $u^{abs}$. Then the inequality constraints force the $u^{abs}$ to be equal to the absolute value. Note that this approach works only because we are minimizing $u^{abs}$ in the objective function. Inequality constraints impose that $u^{abs}$ should always be greater or equal to $u$ and $-u$. Since $u^{abs}$ is minimized it will either, be equal to $u$ if $u$ is positive or equal to $-u$ if $u$ is negative. These are exactly the properties of the absolute value function.

### 3.5.2 Hard constraints

The MPC should also satisfy multiple constraints that exist in the system. Hard constraints that need to be satisfied are positive water level and lateral inflow, and upper and lower bounds on the flow through the power plant components. Lower bound on the control inputs can be used if some gates are controlled manually and not by the controller. In other situations this bound is set to zero.

$$0 \leq A_{n,i} \tag{27a}$$

$$0 \leq Q_{l,i} \tag{27b}$$

$$u^{\text{MIN}} \leq u \quad \leq u^{\text{MAX}} \tag{27c}$$

### 3.5.3 Soft constraints

For the power plants discussed in this thesis, some requirements imply that the water level can not change more than 10 cm a day. This requirement can be imposed in the MPC as a soft constraint. A soft constraint is used such that the MPC does not become infeasible, in a case if the water level happens to change faster than the constraint limits it to. The constraint relaxation is implemented using quadratic penalty as shown in [8]:

$$J+ = p \left\| \epsilon_i \right\|_2^2 \tag{28}$$

and the soft constraint can be described with:

$$\Delta y_i \leq \Delta y_i^{\text{MAX}} + \epsilon_i \tag{29a}$$

$$\Delta y_i = y_i - y_{i-1} \tag{29b}$$

where $p$ is a scalar penalty parameter, while $\epsilon$ represents a measure of the constraint violation. When the penalty parameter $p \to 0$ we obtain the unconstrained solution, when $p \to \infty$ the soft constraint becomes a hard constraint.

### 3.5.4 Disturbance

The MPC gets estimates of multiple disturbances from the MHE. These are then used in the MPC to predict the future behavior of the system. The easiest way to use those disturbances is to assume they stay constant for the whole prediction horizon. In many cases, it is however not realistic. Since the MHE estimates the disturbances using its derivatives, we can use that information to try to predict its future behavior. Therefore the disturbance in the controller has been described as:

$$w_{i+1} = w_i + \beta \dot{w}_i \tag{30}$$

where $w_i$ is the disturbances vector ($Q_l$ and $Q_{\text{bias}}$), $\dot{w}_i$ is the derivative of the disturbances from the MHE, and $\beta \in [0, 1]$ is a decaying factor. This factor decides how the derivative is used. If $\beta = 0$ the disturbance will stay constant for the whole horizon, while $\beta = 1$ will make the disturbance increase or decrease linearly with a rate equal to the derivative. Values between 0 and 1 make will make the disturbance decay slowly into a stationary value. When using $\beta \neq 0$ it is necessary to implement some mechanism to check if the disturbance stays within its bounds for the whole prediction horizon. If the disturbance goes outside the bounds the problem becomes infeasible and the MPC fails. In the current implementation, it is done for the lateral flow that has a lower bound at 0. This is done using an if-function that sets the derivative to zero when the lateral inflow approaches zero.

Note that such formulation has been developed just for the unmeasured disturbances, as the measured disturbance is the flow from the power plant Funnefoss in this case. The controller in the power plants keeps the flow constant for as long as possible, so one can assume that it stays constant for the whole prediction horizon.

### 3.5.5 Parameter choice

Choosing weights in the MPC's objective function is an important part of designing the controller as they have a big impact on the controllers' behavior. The terms $W_{\text{SP}}$ and $W_{\Delta u}$ in the developed formulation decide how well the MPC will follow the given setpoint trajectory. The important part to remember is that it is the ratio between these two weights that will decide the behavior. Increasing both the values will only scale up the value of the objective function. Choosing the weights is deciding the trade-off between fast convergence to the setpoint and limiting the use of the control inputs. Since a $\ell_{\text{asso}}$-MPC formulation is used, there is an additional parameter $\alpha$. As described in the theory section 2.4, it decides how hard the MPC should try to lock the control input on a constant value. Choosing it too low might make it hard for the MPC to find a solution, and has in some instances resulted in infeasibility in the simulations that have been run.

When implementing the MPC there are two sample time that needs to be chosen

- Prediction $T_s$ is the sample time of the internal prediction model. It defines how long each prediction step $i$ lasts. The product of prediction sample time and the horizon decides how long into the future the MPC is planning for, or in the case of the MHE how far into the past. The upper bound of prediction $T_s$ is determined by plant dynamics and control response time. Shorter prediction $T_s$ gives bigger control bandwidth but will require a longer prediction horizon to keep the prediction time constant. A longer horizon leads to a larger optimization problem, which is more complex and time-consuming to solve.[25]
- Control $T_s$ is the control sample time and defines how often the MPC is solved at runtime. It is typically equal to the prediction $T_s$, but can also be set to be faster. Faster control $T_s$ generally improves performance and robustness to some extent. This makes sense as the controller is able to respond faster to the changes in the environment. However as the control sample time becomes small, the computational effort increases as the optimization problem is solved more frequently. [25]

### 3.5.6 Complete MPC formulation

$$\min \sum_{i=1}^{N_c} \|y_i - y_i^{\text{ref}}\|^2_{W_{\text{SP}}} + \alpha\|\Delta u_i\|^2_{2,W_{\Delta u}} + (1-\alpha)\|\Delta u_i\|_{1,W_{\Delta u}} + p\|\epsilon\|^2 \tag{31a}$$

s.t.

$$x_{i+1} = F(x_i, u_i, d_i, w_i) \qquad \forall i = k \dots k+N_c-1 \tag{32a}$$
$$w_{i+1} = w_i + \beta \dot{w}_i \qquad \forall i = k \dots k+N_c-1 \tag{32b}$$
$$y_i = H(x_i, u_i) \qquad \forall i = k \dots k+N_c \tag{32c}$$
$$\Delta u_i = u_i - u_{i-1} \qquad \forall i = k \dots k+N_c \tag{32d}$$
$$\Delta y_i = y_i - y_{i-1} \qquad \forall i = k \dots k+N_c \tag{32e}$$
$$|u_i| = u_i^{abs} \qquad \forall i = k \dots k+N_c \tag{32f}$$
$$x_k, d_k, w_k = \text{given} \tag{32g}$$
$$u \leq u_i^{abs} \qquad \forall i = k \dots k+N_c \tag{32h}$$
$$-u \leq u_i^{abs} \qquad \forall i = k \dots k+N_c \tag{32i}$$
$$\Delta y_i \leq \Delta y_i^{\text{MAX}} + \epsilon_i \qquad \forall i = k \dots k+N_c \tag{32j}$$
$$A_{n,i} \geq 0 \qquad \forall i = k \dots k+N_c, \quad n = 1 \dots N \tag{32k}$$
$$Q_{l,n,i} \geq 0 \qquad \forall i = k \dots k+N_c, \quad n = 1 \dots N \tag{32l}$$
$$u^{\text{MIN}} \leq u_i \leq u^{\text{MAX}} \qquad \forall i = k \dots k+N_c \tag{32m}$$
$$\tag{32n}$$

## 3.6 Estimator implementation

### 3.6.1 General formulation

A starting point for implementing the MHE is the optimization problem presented in theory section 2.5. The objective function used is the least square cost function. Regularization term is added to the formulation, that penalizes the deviations from prior model predictions $\tilde{y}$. This is a built-in function in the GEKKO software that has been used. The objective function is constrained by the model dynamics, and the equality and inequality constraints.

$$\min_z J(z) = \min_z \sum_{i=k-N_e}^{k} \|\hat{y}_i - y_i\|^2_{W_{\text{meas}}} + \|\hat{y}_i - \tilde{y}_i\|^2_{W_{\text{model}}} \tag{33a}$$

s.t.

$$\begin{aligned}
\hat{x}_{i+1} &= F(\hat{x}_i, u_i, \hat{d}_i, w_i) && \forall i = k - N_e \dots k && \text{(34a)}\\
\hat{y}_i &= H(\hat{x}_i, u_i) && \forall i = k - N_e \dots k && \text{(34b)}\\
y_i, u_i, d_i &= \text{given} && \forall i = k - N_e \dots k && \text{(34c)}\\
& && && \text{(34d)}
\end{aligned}$$

### 3.6.2 Hard constraints

Hard constraints are constraints that can never be violated, and origin often from the physical characteristics of the system that is used. In the river system, the water level must be always positive. In addition, the flow through the power plants must be positive as the water can not be pumped upstream.

$$\begin{aligned}
A_{n,i} &\geq 0 && \text{(35a)}\\
Q_{\text{aggs},i} &\geq 0 && \text{(35b)}\\
Q_{\text{gates},i} &\geq 0 && \text{(35c)}\\
Q_{\text{overflow},i} &\geq 0 && \text{(35d)}\\
& && \text{(35e)}
\end{aligned}$$

Similary the later inflow should be positive. It means that the lateral inflow can only flow into the main river not out. This is also the most common scenario in a real river. We can also impose an upper bound on the lateral inflow to keep it within realistic values.

$$\begin{aligned}
Q_{l,i} &\geq 0 && \text{(36a)}\\
Q_{l,i} &\leq Q_{l,i}^{\text{MAX}} && \text{(36b)}\\
& && \text{(36c)}
\end{aligned}$$

### 3.6.3 Lateral inflow

Lateral inflow is an unknown disturbance in our system and needs therefore to be estimated as the lateral inflow can have a significant impact on the river dynamics. We need to constrain the lateral inflow such that the estimator doesn't overfit the estimates to measurements, since lateral inflow can be easily used to modify the dynamics of the model. It is necessary to limit how the estimator is allowed to use the lateral inflow to fit estimates to real data. The simplest way to punish excessive use of lateral inflow is to add its derivative to the cost function.

$$J(z) \mathrel{+}= \sum_{i=k-N_e}^{k} \left\|\dot{Q}_{l,i}\right\|^2_{W_{\dot{Q}_l}} \tag{37}$$

In addition, one can add punishment on change of $\dot{Q}_{l,i}$. This corresponds to setting punishment on the second derivative of the lateral flow. This variable has shown to be useful in imposing smoother estimates of the lateral flow.

$$\begin{aligned}
J(z) &\mathrel{+}= \sum_{i=k-N_e}^{k} \left\|\Delta\dot{Q}_{l,i}\right\|^2_{W_{\Delta\dot{Q}_l}} && \text{(38a)}\\
\Delta\dot{Q}_{l,i} &= \dot{Q}_{l,i} - \dot{Q}_{l,i-1} && \text{(38b)}
\end{aligned}$$

### 3.6.4 Q bias

The error in the flow through the power plant is another state that needs to be estimated to be able to correctly reflect systems dynamics. The use of this DOF needs to be punished, such that the estimator doesn't overuse it. It is done in the same fashion as the lateral inflow, by adding the state's derivative to the cost function.

$$J(z) + = \sum_{i=k-N_e}^{k} \left\| \dot{Q}_{\text{bias},i} \right\|^2_{W_{\dot{Q}_{\text{bias}}}} \tag{39}$$

Same as in the case with lateral inflow we can penalize the second derivative as well, to impose a smoother solution.

$$J(z) \mathrel{+}= \sum_{i=k-N_e}^{k} \|\Delta\dot{Q}_{\text{bias},i}\|^2_{W_{\Delta\dot{Q}_{\text{bias}}}} \tag{40a}$$

$$\Delta\dot{Q}_{\text{bias},i} = \dot{Q}_{\text{bias},i} - \dot{Q}_{\text{bias},i-1} \tag{40b}$$

In this case, we know that the bias should ideally be equal to zero. And in practice to be close to zero when there is no flow through the gates. We can therefore add a term in the cost function that tries to drive the bias to zero. It should be weighted lightly as we don't want it to be zero, but to be driven to zero if possible.

$$J(z) + = \sum_{i=k-N_e}^{k} \|Q_{\text{bias},i}\|^2_{W_{Q_{\text{bias}}}} \tag{41}$$

### 3.6.5 Parameters

In some cases, it can be advantageous to allow for parameter approximation. In this case, the change of parameter values needs to be penalized as well to prevent excessive use of those by the MHE. It can be done in a similar matter as lateral flow and bias flow.

$$J(z) \mathrel{+}= \sum_{i=k-N_e}^{k} \|\Delta\theta_i\|^2_{W_{\Delta\theta}} \tag{42a}$$

$$\Delta\theta_{l,i} = \theta_i - \theta_{i-1} \tag{42b}$$

In addition, hard constraints on the parameters are important to keep them within a reasonable range.

$$\theta^{\text{MIN}} \leq \theta \leq \theta^{\text{MAX}} \tag{43a}$$

$$\tag{43b}$$

#### 3.6.6 Complete MHE formulation

$$\min \sum_{i=k-N_e}^{k} \|\hat{y}_i - y_i\|^2_{W_{\text{meas}}} + \|\hat{y}_i - \tilde{y}_i\|^2_{W_{\text{model}}} + \|\dot{Q}_{l,i}\|^2_{W_{\dot{Q}_l}} + \|\Delta\dot{Q}_{l,i}\|^2_{W_{\Delta\dot{Q}_l}}$$

$$+ \|\dot{Q}_{\text{bias},i}\|^2_{W_{\dot{Q}_{\text{bias}}}} + \|\Delta\dot{Q}_{\text{bias},i}\|^2_{W_{\Delta\dot{Q}_{\text{bias}}}} + \|Q_{\text{bias},i}\|^2_{W_{Q_{\text{bias}}}} + \|\Delta\theta_i\|^2_{W_{\Delta\theta}} \tag{44a}$$

s.t.

$$\hat{x}_{i+1} = F(\hat{x}_i, u_i, \hat{d}_i, w_i) \quad \forall i = k - N_e \dots k \tag{45a}$$

$$\hat{y}_i = H(\hat{x}_i, u_i) \quad \forall i = k - N_e \dots k \tag{45b}$$

$$\Delta\dot{Q}_{l,i} = \dot{Q}_{l,i} - \dot{Q}_{l,i-1} \quad \forall i = k - N_e \dots k \tag{45c}$$

$$\Delta\dot{Q}_{\text{bias},i} = \dot{Q}_{\text{bias},i} - \dot{Q}_{\text{bias},i-1} \quad \forall i = k - N_e \dots k \tag{45d}$$

$$\Delta\theta_{l,i} = \theta_i - \theta_{i-1} \quad \forall i = k - N_e \dots k \tag{45e}$$

$$y_i, u_i, d_i = \text{given} \tag{45f}$$

$$A_{n,i} \geq 0 \quad \forall i = k - N_e \dots k, \quad n = 1 \dots N \tag{45g}$$

$$Q_{l,n,i} \geq 0 \quad \forall i = k - N_e \dots k, \quad n = 1 \dots N \tag{45h}$$

$$Q_{l,i} \leq Q_{l,i}^{\text{MAX}} \quad \forall i = k - N_e \dots k \tag{45i}$$

$$Q_{\text{aggs},i} \geq 0 \quad \forall i = k - N_e \dots k \tag{45j}$$

$$Q_{\text{gates},i} \geq 0 \quad \forall i = k - N_e \dots k \tag{45k}$$

$$Q_{\text{overflow},i} \geq 0 \quad \forall i = k - N_e \dots k \tag{45l}$$

#### 3.6.7 Parameter choice

As discussed earlier the system contains multiple disturbances that need to be estimated. The correct weighting of these in the objective function is important to make the MHE's results reflect a real system. All of the unknown disturbances are DOFs in the MHE that can be used to modify the results to minimize the estimation error. Based on knowledge about the system we can choose the weights in such a way that forces the MHE to use the provided DOFs as intended. For instance, the $Q_{\text{bias}}$ is a slow varying disturbance. Its weights should therefore be much higher than the weights used for the lateral inflow $Q_l$, which is a disturbance that can change quite fast. It is also desirable that both disturbance estimates should have a relatively smooth trajectory. In the current implementation, the penalization of the second derivative of the disturbance is useful for that purpose. By having a smooth trajectory we minimize the likelihood to estimate noise and should get better results when predicting the system's future behavior. The estimator contain also a part that tries to minimize the difference between the estimated values and the actual measurements. It is weighted using parameter $W_{meas}$, and should be chosen such that the estimation errors are within acceptable values. As described in the theory this term has a probabilistic interpretation, where it describes the correlation between the measurements and the estimates. If a measurement do not correlate with a modeled variable, or the measurement is corrupted or lost, the weight can be set to zero, which effectively discards that measurement. Note that increasing the $W_meas$ will have vare similar effect as decreasing penalization of the disturbance. All the terms are part of the same objective function and it is the weight ratio between different terms that determines the estimators' behavior. One must allow one term to get bigger such that another can be smaller, and the weights decide that balance. The terms in the objective function represent different physical quantities. A good starting point for choosing weights is to scale those into the same range. As an example, the water levels hav accuracy in the range of centimeters, and can be scaled up with factor $10^2$. Same logic can be used for the lateral inflow from Vorma river can reach values of few hundred $m^3/s$ and vary with few $m^3/s$ per sample. On the other hand the lateral inflow in Glomma2 is very small or non existing and its weight can be scaled up in relation to lateral inflow from Vorma. Corrcet scaling of the variables generally improves solvers convergence, and GEEKO has some options that enbales autmatic scaling to achieve this [12].

## 3.7 Software

All the simulations and test has been run using Python programming language. Python is a high-level free programming language and can be used in many different types of applications. To solve the optimization problems in Python, GEKKO Optimization Suite has been used. GEKKO provides a user-friendly, object-oriented Python interface to develop models and optimization solutions. Science python is not a very fast programming language, GEKKO converts models to a low-level representation in the Fortran back-end for speed in calculations. GEKKO then interacts with solvers for linear, quadratic, nonlinear, and mixed-integer programming in the back end. The solutions are then loaded back to Python for easy access and further analysis. [4] The IPOPT solver has been used to solve optimization problems in the presented implementation. It is importan to note that GEKKO supprts the IPOPT solver only on Windows operating system.

GEKKO Optimization Suite supports several modes of operation. The two main categories are steady-state and dynamic solutions. The core of all modes is the nonlinear model, which does not change between a selection of different modes. Thus, once a model is created it can be implemented as MPC or MHE by setting a single option. [4] How different modes are utilized in the simulations is described in section 3.9.

Table 4: Modes avaible in GEKKO (from [4])

| | **Non-dynamic** | **Simultaneous dynamic** | **Sequential dynamic** |
|---|---|---|---|
| Simulation | Steady-state simulation | Simultaneous dynamic simulation | Sequential dynamic simulation |
| Estimation | Model parameter update (MPU) | Moving Horizon Estimation (MHE) | Sequential dynamic estimation |
| Control | Real time optimization (RTO) | Nonlinear control (MPC) | Sequential dynamic optimization |

Gekko provides different variable types for building the model:

- `SV` - state variables, as the name suggests used to describe the states in the model.
- `CV` - controlled variables, used to describe measurements in the model.
- `MV` - manipulated variables, used to describe control inputs and disturbances in the model.
- `FV` - fixed variables, same as `MV` but the value is fixed for the whole horizon.

How these variables are used in the solver, is mainly decided based on two parameters:

- `STATUS` - allows the optimizer to change the value to minimize the objective function if `STATUS` is equal to 1. For `CV` adds a term in the objective function that minimizes the `CV` error. This term will differ for estimation and control. This term will differ for estimation and control mode. The variables with `STATUS` equal to 1 can be seen as the decision variables $z$ in the optimization problem.
- `FSTATUS` - allows updating the value of the variable using an external measurement.

Variables in GEKKO has also many others built-in properties, that can simplify defining the optimization problem. Here are described some that have been used, but many more are explained in GEKKO's documentation [13].

- `LOWER` - the lower limit of a variable.
- `UPPER` - the upper limit of a variable.
- `DMAX` - Applies a hard constraint that prevents the `MV` from being changed by more than the specified value in one time step
- `DCOST` - Adds a term to the objective function that gives a penalty for changing the `MV`. This parameter was mainly use to implement the $\Delta$ variables.
- `WMEAS` - A weighting factor to penalize deviation of current model predictions from measured values (only for the MHE).

- WMODEL - A weighting factor to penalize deviation of current model predictions from prior model predictions. Similar to technique mentioned in section 2.5.2 (only for the MHE).
- WSP - A weighting factor to penalize a squared error from the setpoint trajectory with the final target SP (only for the MPC).

One advantage of GEKKO is its ability to discretize continuous equations. To achieve this GEKKO uses orthogonal collocation on finite elements. Orthogonal collocation represents the state and control variables with polynomials inside each finite element. This is a form of implicit Runga-Kutta methods, and thus inherits the benefits associated with these methods, such as stability [4]. More detailed information about the orthogonal collocation method can be found at [24] and [5].

### 3.7.1 Model implementation in GEKKO

- Measurements $y$: CV with STATUS=1 and FSTATUS=1.
- Control inputs $u$: MV with FSTATUS=1, and STATUS=1 for MPC and STATUS=0 for MHE.
- States $x$: SV with FSTATUS=1 for MPC and FSTATUS=0 for MHE. Does not haveSTATUS prioperty as it is always allowed to be manipulated by the optimizer.
- Known disturbances $d$: MV with STATUS=1 and FSTATUS=1. Similar to control input but the optimizer can not adjust these in MPC mode.
- Unknown disturbances $w$: MV with STATUS=0 for MPC and STATUS=1 for MHE, and FSTATUS=1 for MPC and FSTATUS=0 for MHE. In MPC mode those are provided from the MHE and should not be a decision variable. In MHE mode those are unknown and should be a decision variable to estimate those.
- Paramters $\theta$: FV with STATUS=0 for MPC and STATUS=1 for MHE if it is to be estimated, and FSTATUS=0 for both MHE and MPC

## 3.8 Control system implementation

The control solution developed in this thesis is meant to replace the currently used water-level controller. It has been designed such that no change to the other parts of the system should be necessary. How the new control solution would look like is shown in figure 18. Here the MHE and MPC take all the available measurements from the existing system and outputs control input in form of total flow that should be implemented for a power plant. This is sent to the flow divider that further decides how the flow should be utilized. One downside of the developed solution is that it uses high-level programming languages like python. In addition, the GEKKO library needs to run on Windows operating system to be able to use the IPOPT solver. This means that this solution can not be implemented on a standard PLC unit. A possible solution could be to implement the controller on an Industrial PC (IPC), that can operate both as a PC with Windows and a PLC. Python has libraries that support a variety of communication protocols that can be used to communicate with a PLC that is used to run other parts of the control system. Note that other optimization software can be used to implement the MPC and MHE, so the implementation is not limited to GEKKO.

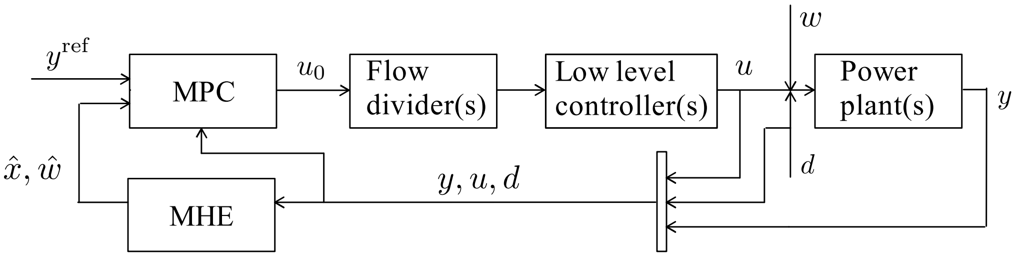


Figure 18: MPC and MHE in the control system

## 3.9 Simulation implementation

To test the developed solution different kinds of simulations have been used. To test the estimator the MHE has been simulated using only real data as input. To test the controller the river model has been run in parallel with the controller. The control inputs from the controller were then applied to the simulated system, such that MPC's behavior and performance could be tested. The simulated river model can also take real data as input to make the simulations more realistic.

### 3.9.1 Initialization

For nonlinear problems, initial values are an important factor that decides if the model converges to the indented solutions. In the case of an optimization problem, bad initial conditions may also result in an infeasible problem. To resolve that some initialization strategies are available. The purpose of initialization strategies is to find a solution close to the originally intended problem. These strategies may improve the performance and ability to find a feasible or optimal solution. One method for the initialization of nonlinear models is to simplify the model form so that a solution can be computed and used to seed the original problem with better initial values. One such method is steady-state initialization which is accomplished by setting all derivative terms $dx/dt$ and solving the resulting set of equations and objective functions. [23] Another way to initialize the application is by using a mode called in GEKKO "cold start". In a cold start all `MV`, `FV`, and `CV` variables have `STATUS` set to 0. This reduces temporarily the number of degrees of freedom to achieve a feasible, but a suboptimal solution. [2]

It is difficult to find reasonable initial conditions for St. Venant equations, as values such as cross-section area of the water and water velocity are not normally measured or commonly known. These values can also vary significantly when river geometry parameters are adjusted. Therefore some kind of initialization strategy is necessary. GEKKO library provides an easy way of running steady-state (non-dynamic) simulations by changing the MODE parameter (see section 3.7). Because of that, the steady-state simulations are used to acquire the model's initial conditions. This means that the model is assumed to be in steady-state at the beginning of the simulation. This will not always be the case, but the goal here is to provide initial conditions that are close, but not exactly equal to, the optimal solution.

In this thesis, the developed GEKKO model is used in an MHE, MPC, and to run open-loop simulations of the system. The initialization sequence is marked with blue on the flowchart diagram (figure 19). The values from the initialization sequence are updated only at the first simulation iteration.

### 3.9.2 Simulation

Simulations have been the main way of testing developed methods. The main simulation loop has been implemented as follows. The MHE receives measurements $y$ and control inputs $u$. The solution from the MHE is forwarded to the MPC which calculates the new control input $u$.

The measurements $y$ can be obtained in two different ways. The first method is when all measurements and control inputs are actual measurements taken at the power stations. The actual power plant data are represented by the block "data" in the diagram. This was the main way of testing the estimator. If the MPC is turned on for this simulation, its control input is saved in the results, but not implemented in any way. The second method is where a model of the system is simulated in parallel with the estimator and the controller (represented by block "System simulation", figure 19). In this case, the system measurement $y$ comes from this simulation and the control inputs $u$ calculated by the MPC are implemented in this simulation. This simulation uses the same equations as the MPC and MHE but has an option to add measurement and control noise to better reflect a real system. Running simulations in this way is the same as assuming that our model is an ideal reflection of the real system. This is of course not correct, but such simulations allow us to test if the MHE and MPC work in the ideal conditions. If it fails here it will most probably also fail in a real application. This was also a good way to test different behaviors of the MPC.

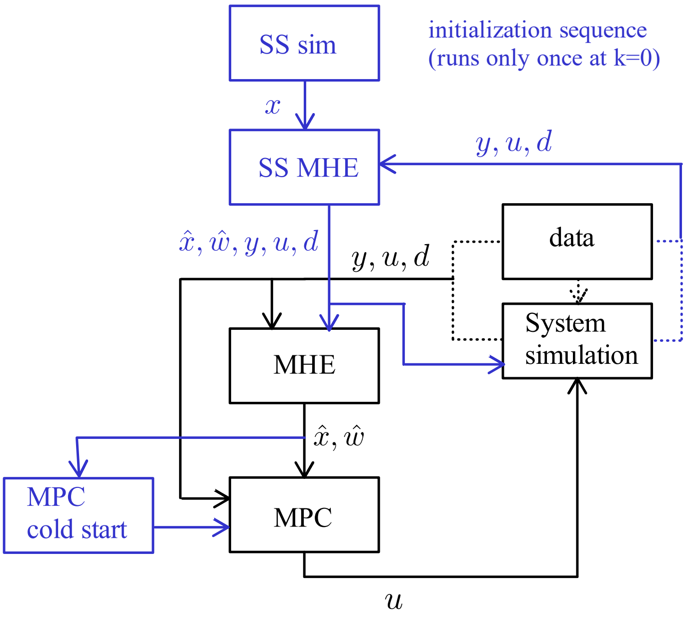


Figure 19: Simulation flow chart

#### 3.9.3 Data choice

The developed model has been tested on multiple periods of the provided data from the power plants. These have been divided into several cases:

- Case1 (2022-02-22 05:00:00): The system is in a steady state. The water levels are oscillating by a few cm due to how the control dead-band is implemented in the current solution. This is supposed to be an easy case to verify the basic functionality of the estimator.
- Case2 (2021-10-18 06:00:00): There is a sudden increase in the flow from the Mjøsa lake, which is an unknown disturbance. This is supposed to be a slightly harder case to see how well the estimator can estimate unknown disturbances.
- Case4 (2021-09-11 20:00:00): There is a step in the flow through the Funnefoss power plant. This data is meant to test how the control system uses information about known disturbances.
- Case5 (2021-10-01 00:00:00): Big increase in the flow with values over $1000m^3/s$. Used as validation case for long simulations.

#### 3.9.4 Simulation parameters

The river system in this thesis has very slow dynamics, therefore the prediction and control sample time has been chosen to be $5min$. The horizon has been chosen as $N_c = 12$ which gives a prediction time of one hour. The same horizon and sample time have been chosen for the MHE. The power plant data has been resampled using "pandas" `resample` function to $5min$ to match the timestamps of the control system. Such choice of the parameters resulted in solve time of a maximum of a few seconds for both the MPC and MHE. This was important to be able to run many longer simulations without needing to wait too long for the results. In an actual implementation, it might be a good choice to choose faster control sample time, to be able to react faster in case of unexpected critical events. Solving both MHE and MPC take up to 5 seconds in normal conditions, but in some more difficult cases, this might be higher. The MHE and MPC need to be able to keep up with the control sample time and therefore lower bound for the control sample time should lay at about 20 seconds. This value value is chosen based on the long simulations presented in section 4.1.8 and 4.2.5. Tests on a real system would be necessary to determine it, since the solution time will vary with computers performance. The value of $5min$ gave good results in the simulations and gave a faster execution time.

For the controller and estimator following parameters were used as baseline values:

Table 5: MHE parameters

| **Parameter** | **Value** |
|---|---|
| $W_{\text{meas}}$ | $3.5 \times 10^2$ |
| $W_{\text{model}}$ | $5.0$ |
| $W_{\dot{Q}_l}$ Glomma1 | $10.0$ |
| $W_{\dot{Q}_l}$ Glomma2 | $1.0 \times 10^3$ |
| $W_{\Delta\dot{Q}_l}$ Glomma1 | $2.0 \times 10^2$ |
| $W_{\Delta\dot{Q}_l}$ Glomma2 | $2.5 \times 10^4$ |
| $W_{\dot{Q}_{\text{bias}}}$ | $1.5 \times 10^4$ |
| $W_{\Delta\dot{Q}_{\text{bias}}}$ | $5.0 \times 10^6$ |
| $W_{Q_{\text{bias}}}$ | $1 \times 10^{-6}$ |
| $W_{b\_scale}$ | $1 \times 10^{-4}$ |
| $W_{k_s}$ | $1 \times 10^{-3}$ |
| $N_c$ | $12$ |

Table 6: MPC parameters

| **Parameter** | **Value** |
|---|---|
| $W_{\text{SP}}$ | $15 \times 10^2$ |
| $W_{\Delta u}$ | $0.1$ |
| $\alpha$ | $0.3$ |
| $\beta$ | $0.9$ |
| $N_e$ | $12$ |
| $p$ | $1.0 \times 10^{14}$ |
| $\Delta y^{\text{MAX}}$ | $\frac{10\times10^-2}{24\times60} \times 5$ |

In the initial testing, the estimator was not able to find any good solutions while both measurements of the tailwater and headwater were included. To solve this problem the tailwater measurements were rejected by setting $W_{\text{meas}} = 0$ for those measurements. This way only water levels that are controlled were used in the estimator, which gave good estimation results. This is not the best solution since information about the system is rejected. Possible ways to include this measurement have been discussed in section 5.

# 4 Testing results and analysis

This section presents some results of the testing that has been done throughout working with this thesis. The results presented in this section are not meant to show a perfect performance. Finding the MOST optimal parameters is very time-consuming and not practical. When working on this project, it became clear one can always find a slightly better solution over and over again, and the circle will never stop. The goal was however to find a solution that gives satisfactory results for this case, which is what I attempted to achieve. The results presented are also meant as guidelines for further developing and tuning such a control system in the future.

## 4.1 Estimation

Developing a good MHE was a crucial step for further implementation of the MPC. Most importantly the states need to be estimated, to be able to use the MPC. In addition good MHE performance confirms that the models used are able to reflect real dynamics. The same model is also used in the MPC to predict systems' future behaviors. If these predictions are correct the controller will be able to find an optimal control sequence that makes the system converge to a given setpoint. Estimators' goal is also to estimate the unknown disturbances, that can have a big impact on the behavior of the system. To test if the MHE fulfills its task it has been simulated with real measurements and control signals as inputs.

Results of the MHE simulations are presented as plots for each river segment, Glomma1, and Glomma2. All estimator plots have the same format that will be briefly explained here. The two first plots show the estimated inflow and outflow from the river segments. For Glomma1 the inflow will be equal to the provided measurements from the Funnefoss power plant. The rest of the in- and out-flows will be equal to the provided flow measurements from the two other power plants, plus the bias term calculated by the estimator. The next plot is the water level at the headwater of the power plant, which is the level that is to be controlled. With a dashed line the actual measured water level is also plotted. The two last plots are the estimated unknowns in the system, lateral inflow, and the bias flow in the power plants. For the Glomma1 the lateral inflow is measured at Ertesekken and this value is plotted with a dashed line. Note that the measured value is not used in the control system and is only used for comparison in the analysis. The measurements from

Ertesekken have much lower resolution than other measurements, and will therefore appear less noisy.

One way to determine estimator performance is to look at how the MHE uses provided DOFs to match the results. An estimator with a lot of freedom to choose the disturbances can perfectly match the measured values. In such a case, the model will not match the real system, and the disturbances will be used to estimate noise and all the inaccuracies which will result in an overfitted model. We can therefore look after points in the simulation where the estimated disturbances seem unnatural, for instance, oscillations in disturbances that happen exactly when the control inputs change.

### 4.1.1 Spatial discretization

Spatial discretization is a necessary step to implement the St. Venant equations. When the model is discretized it can be seen as dividing the whole river into smaller finite-size control volumes. A question that naturally arises from such a method is, how big should the control volumes be? The goal of the model is to run as quickly and accurately as possible. Dividing the whole river into fewer volumes will reduce the simulation time, as the number of equations is reduced. It is on the other hand difficult to say how the size of volumes affects the model's accuracy. To find the answer to this, a series of simulations have been run with different discretization settings. The parameter $dx$ is used to change the size of the volumes in the model, where $dx$'s value is the length of each control volume in meters. The average width and linear slope have been chosen for the simulations, such that these parameters are constant for all the simulations. Note that choosing the other methods that have been presented to describe width or slope would result in different values for different $dx$ as the measurements need to be modified to match the number of the control volumes. By using constant average width and linear slope the $dx$ should be the only parameter that varies across the simulations. As the measure of the model's accuracy, MHE's objective function value is used. This way both differences between the model and measurements, and the use of additional DOF in the model are taken into account. If the model is using $Q_{\text{bias}}$ or $Q_l$ less, means that it's more accurate and needs less correction.

Results are presented in the figure 20. Note that $dx = 5000$ is implemented only for Glomma1 as Glomma2 segment is too short for such a long control volume as it is equal to its total size. In this simulation Glomma2 segment usees $dx = 2000$ instead.

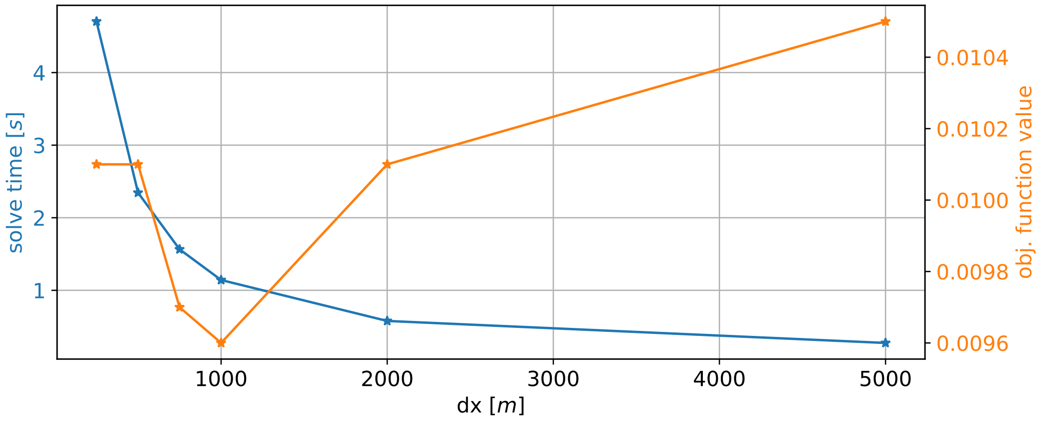


Figure 20: Average solve time and objective function value for different dx

The results show that the solving time decreases exponentially as the $dx$ becomes bigger. This is an expected result as shorter $dx$ results in fewer equations in the model. The differences in the average objective function values are so small that it has almost no effect on the estimator performance. This differs from what has been found in the pre-project where the accuracy decreased significantly when $dx$ became very big. The reason for that might be the use of better software that uses a solver created for solving nonlinear equations and the use of a better time discretization scheme than in the pre-project. The big increase in the average objective function value at $dx = 2000$ is most probably caused by the segment Glomma2, which short and such a long $dx$ might cause problems. Therefore for the rest of the simulations the $dx = 2000$ fro Glomma1 and $dx = 1000$ for Glomma2.

### 4.1.2 River slope

River slope is an important parameter in the St. Venant equations. In the pre-project, it has been shown that slope has a great effect on the waves' propagation time because a steep river bed will increase water velocity and decrease propagation time. Accurate data about the river bed are not common. Some different methods to obtain this data have been tested to see which provides the most accurate results. The methods used are described in section 3.1.1. It should be noted that the slope of the Glomma river is very gentle, and there should not be any big differences between these methods. The height difference between Funnefoss tailwater and Bingsfoss headwater is about 22 m over a distance of 28 km.

Table 7: Estimation results for different slope sampling methods

| **Slope** | **avg. solve time** | **avg. obj cost** |
|---|---|---|
| Linear | 0.8124 | 0.01005 |
| Resampled | 0.7995 | 0.01044 |
| Resampled (unfiltered boundaries) | 0.8028 | 0.01037 |

As shown by the results the way the slope is implemented has almost no effect on the models' performance. This is as said above mostly due to how gentle the slope is. Because of the gentle slope, the implementation method has not enough effect to change flow velocity in the different cases. Since in most applications the water level at the upstream and downstream power plant is of interest, the method that keeps unfiltered boundary measurements is mainly used as it preserves elevations accurately at the model's boundaries.

### 4.1.3 River width

River width is another parameter that is necessary to run St. Venant equations. In the pre-project, it has been concluded that width variations along the river had very little effect on the flow. Because of the lack of real data, it was impossible to verify how the river width affects the water levels, and how accurate width data is necessary for satisfactory results. The river segment considered in this thesis is good for such tests as the width variations on the Glomma2 segment are quite big. To test the significance the MHE has been run with different width models that are described in section 3.1.1. As a measure of accuracy, the average objective function value has been used.

Table 8: Estimation results for different width sampling methods

| **Slope** | **avg. solve time** | **avg. obj cost** |
|---|---|---|
| Constant (Avg.) | 0.593 | 0.01005 |
| Resampled | 0.591 | 0.00928 |
| Resmapled (unfiltered boundaries) | 0.594 | 0.00958 |

Results in table 8 show a very small difference in the estimators' accuracy with these different approaches. There are some very small differences in the results, where the "resampled" method seems to do the best. In practice, however, the differences are so small that it doesn't matter which method is chosen for this case. It has been observed that what method is the "best" varies depending on how other parameters like disturbance weights and friction coefficients are chosen in the MHE. There are a lot of parameter combinations that can be tested and might result in better results. The gain is however very small in the actual estimation. For most of the simulations in the thesis, a constant average width has been used for simplicity.

How the width varies along the river has little effect on the estimation, but the width's value itself has a much greater effect. To demonstrate this the MHE has been simulated on real data with different configurations. For all the simulations a constant average width is used. The river width is then scaled down by different factors. Results from the simulations with different scaling factors are presented in figure 21.

The results show clearly how the choice of the parameters affects the use of the DOFs to match the measurements. Generally smaller variations in the DOFs mean that the model is estimating the model easier. For Glomma1 the smaller scaling factor reduces the jumps in the lateral flow estimation that happen exactly when the outflow changes. Too low values create however more oscillations in the bias flow. The values that keep the bias flow relatively stable are between 0.7 and 0.9. For Glomma2 it is more clear that the value of 0.8 gives the best performace. Scailing of 0.8 has been therefore chosen as the one giving best results and will be used for all the other simulations.

The reason for the better performance of the downscaled river width is that it is probably closer to the real river cross-section area. The model uses a rectangular river bed shape, while the real river bed is closer to a trapezoid shape. A trapezoid with a longer side equal to the length of the rectangle will have a smaller area than the rectangle. That's why the width is scaled down to achieve a better result. So why not use a trapezoid shape in the model? This would require us to know the length of one side of the trapezoid and the angle of the sides. The length can be obtained using digital maps as in the rectangular case. This leaves us with one variable, the angle of the trapezoid sides, as an unknown that needs to be found. It ends up either way with one unknown parameter, where we either need to find out how much the rectangular shape needs to be downscaled or find the angle of a trapezoid shape.

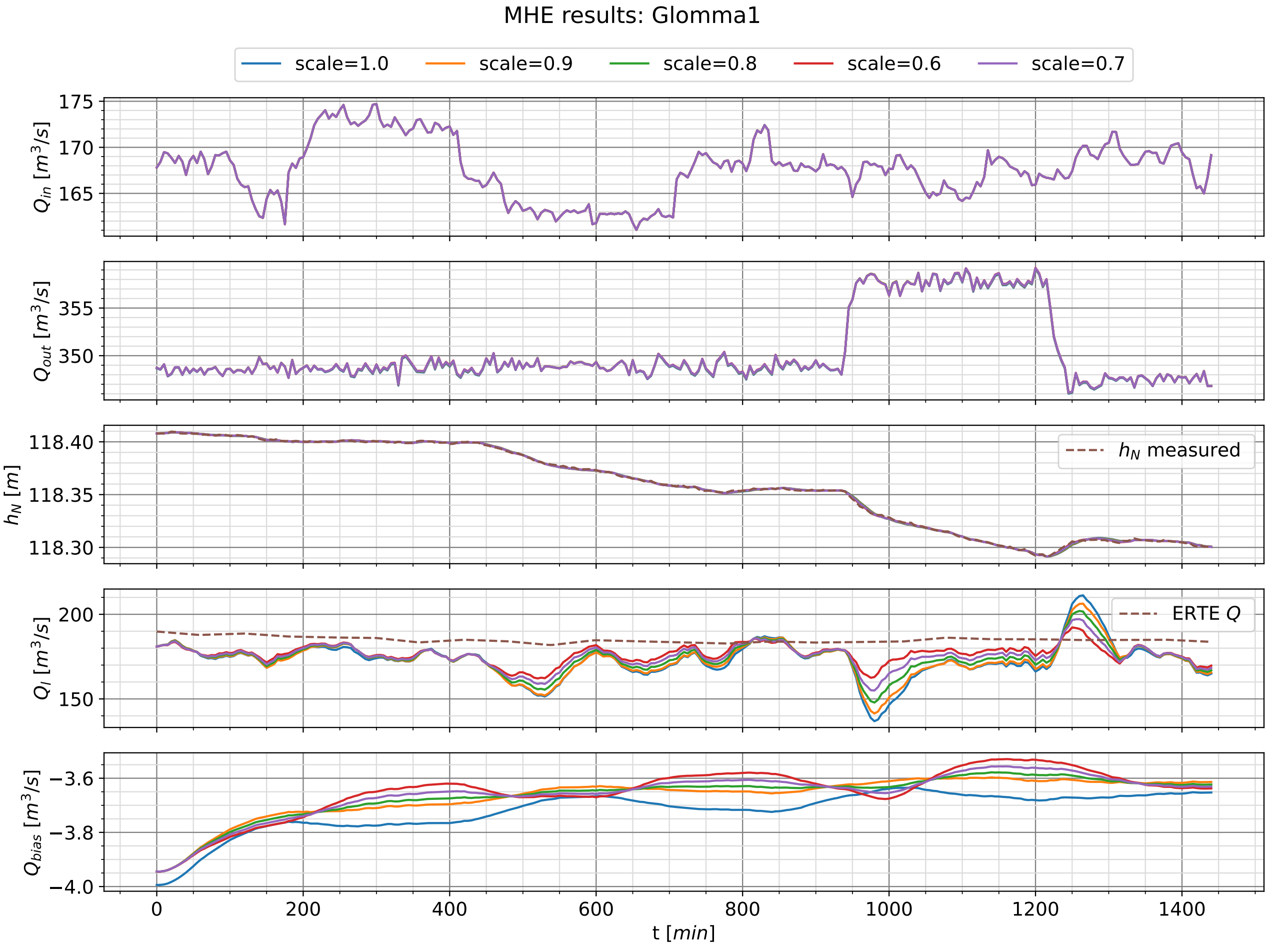


Figure 21: System's behaviour with different width scaling (case1)

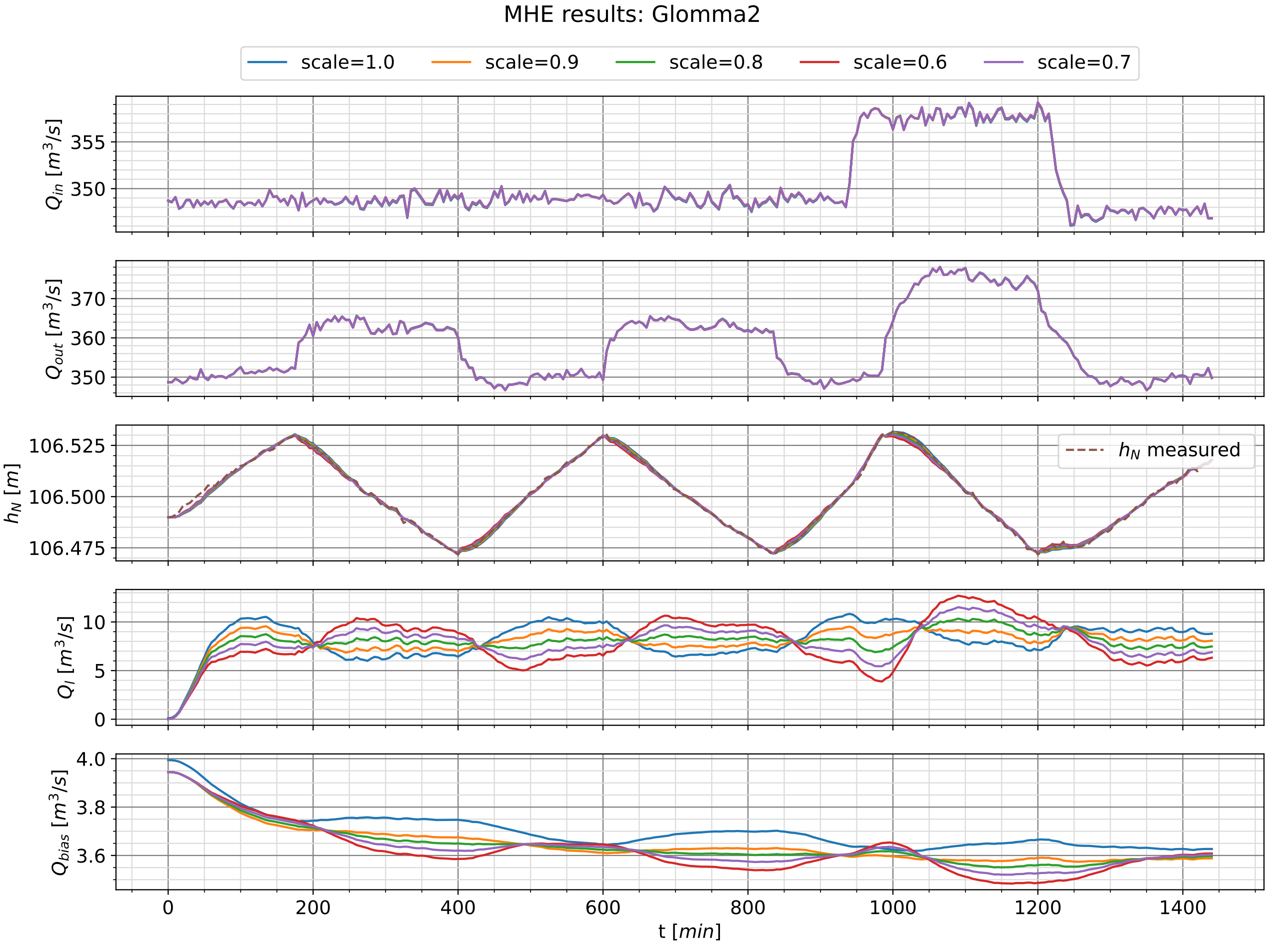


Figure 21: System's behaviour with different width scaling (case1), (cont.)

### 4.1.4 Friction coefficient

Another parameter that is included in the model and might be challenging to obtain is the friction coefficient $k_s$. As described in the theory section 2.1 the friction coefficient can be found in various tables, for instance, manning's values at [17] (convert from manning's values using $k_s = 1/n$). In this table, the values for the main channel can vary from 40 for a "clean, straight, full stage, no rifts or deep pools" channel, to 6 for a channel with "very weedy reaches, deep pools, or floodways with heavy stand of timber and underbrush".

The previous simulations with varying width had some unnatural jumps in the DOFs. For instance spikes in the lateral inflow in figure 21 at about $t = 1000$ and $t = 1300$. These happen exactly after the flow through the Rånåsfoss power plant has changed significantly. This is probably caused by som inacuracies in the model. The system has been simulated in the same way as prevoiusly using differetn $k_s$ values, to check it's effect. The results are presented in the figure 22. All these simulation has also width scaled down with factor of 0.8, as discussed in the previous section.

The results show some improvement in the simulations compared to the previous results. The best improvement was observed at Glomma1 with $k_s = 5$. The jumps in lateral inflow at $t = 1000$ and $t = 1300$ are slightly smaller than for the other values. By looking only at this the $k_s = 3$ reduces the jumps the most, it creates however bigger jumps in the lateral flow at other time instances, in addition to more oscillations in the bias flow. For Glomma2 the differences between different $k_s$ values are much smaller, except for $k_s = 3$ which worsens the performance. In the end the $k_s = 5$ has been chosen as the preferred one for Glomma1, and $k_s = 8$ for Glomma2. Choosing two different coefficients for the two segments isn't unnatural as these segments are different.

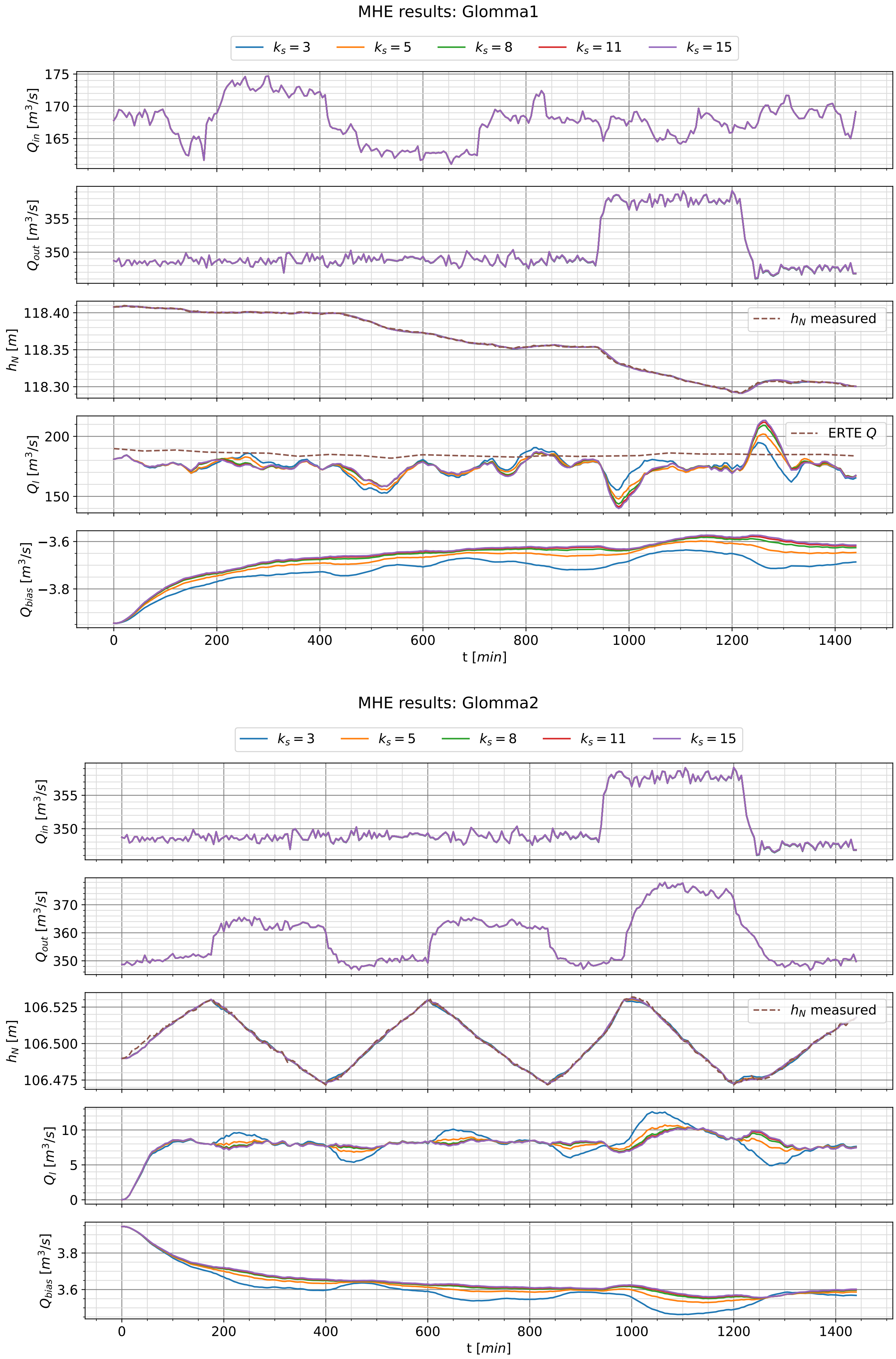


Figure 22: System's behaviour with different friction coefficient (case1)

### 4.1.5 Parameter estimation

Choosing the correct parameters for the model is important for its accuracy, but it can also be a very time-consuming process to do manually. In an attempt to simplify this process, the MHE has been run with the possibility to estimate the model parameters. Change in these parameters has also been penalized quite heavily in the objective function since other DOFs should be prioritized over the parameters to match the data. A simulation has then been run for a longer period of time than previously, to see if the parameters will approach some values that correspond to previous results. The initial values for $k_s$ has been set to 20 and width scale to 1.0, for both cases. Based on previous test, those values are not optimal and should be adjusted relatively quickly.

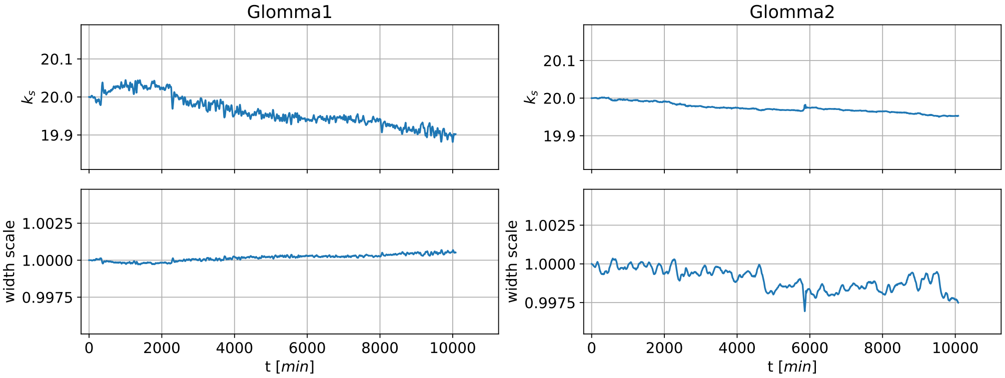


Figure 23: Estimated parameter's values

The results from the simulations are shown in figure 23. The parameters do not converge to any specific value close to the previous findings. For segment Glomma1 the width scale is adjusted up, which is the opposite of what has been concluded earlier. The values are also noisy, which suggests that the model is overfitting the measurement noise. The method tried here is very simple, and some more advanced setup is needed to make the parameter estimation work. Running the parameter estimation beforehand on a very long data sequence and then using the found parameter as constants in the MHE is probably a better approach. More development and tests on the parameter estimation have not been done in this thesis, as the parameters from the previous tests gave good results.

### 4.1.6 MHE weight tuning

The choice of the weights in the objective function of the MHE affects how it estimates the disturbances. Results presented here show a part of the tuning process of the estimator, as well as problems that can arise. One property that might be desirable is to get smooth trajectories of the disturbances to avoid fitting data to the noise. To impose smoother trajectories the model has been simulated with different weights on the disturbances derivative $\dot{w}$ and "second derivative" $\Delta\dot{w}$. The results are shown in figure 24. The values for the cases with changed weights have been scaled up by a factor of 4. The blue plot (baseline) uses the values that are used for all the other simulations. Increasing the weight of $\dot{w}$ (green plot) makes the oscillations in lateral flow smaller. However, the bias flow is oscillating much more than for the baseline case. Based on results in the previous sections it can be solved by re-tuning the friction coefficient and the width scaling. But it demonstrates the difficulty with the tuning of this estimator, where all parameters are highly interconnected, and changing one of them might have a negative effect on the others. Another potential problem with increasing this weight is that it limits how fast the estimated lateral flow can change. In a case where the lateral flow changes rapidly, the estimator will not be able to keep up. The plot shows that the change rate of the lateral flow is lower than for the two other cases (at about 300 min). An attempt to allow for a fast change of the lateral flow and smoother estimates was the use of the weight on the $\Delta\dot{w}$ (orange plot). We can see from the plots that this is achieved and the disturbance changes as fast as for the baseline case. However since the second derivative is now penalized, stopping the change of the flow is much slower. This applies to all the jumps in the disturbances, making that all the jumps have slightly higher altitudes and oscillate more than in the other cases. Increasing the weight even more, will lead to even bigger oscillations, and in the worst case, the oscillations can become unstable. Similar to the previous attempt, the bias flow also starts to oscillate more, which might indicate that the model parameters must be re-tuned.

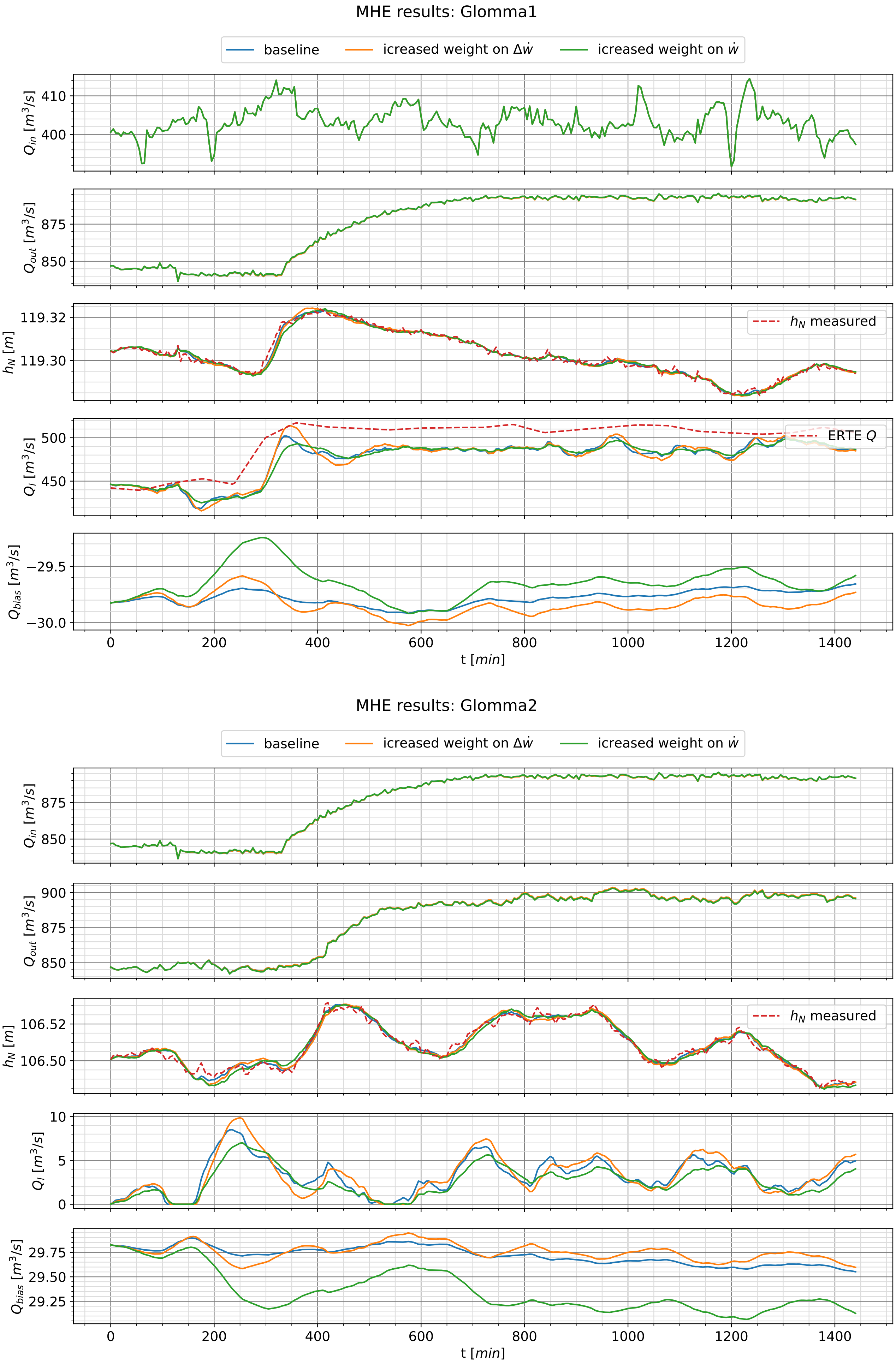


Figure 24: Effect of imposing smoother disturbance response (case2)

### 4.1.7 MHE and data loss

As an additional test, the MHE has been simulated with data loss at time $670min$ to check its performance in such cases. The data that is lost is the measurements of the water level. When the data loss occurs the estimator degrades to an open-loop estimation and runs like that for the rest of the simulation. This might not reflect a real-life scenario, but the goal here is to see how long the predicted values stay close to the real ones. If the open-loop stimulation corresponds well with the actual measurements it is an indication that the model reflects the real behavior well, and the model is not overfitted to a particular case. The open-loop estimation is also similar to how the MPC is going to predict the feature behavior of the system. When the data loss occurs the unknown disturbances can not be estimated anymore. Two strategies of what disturbance values should be used in an open-loop simulation have been simulated. In the first method, the values are frozen, and the last values that were calculated by the MHE are used. In the second method, the derivative of the disturbances is set to decay exponentially with a given factor. The idea behind this is that in the near future after the data loss event, the disturbance should keep the trend it had before. After a while, the derivative will approach zero and the value would be frozen.

The tests were run for case2 and case3 and the results has been plotted in figures 26 and 25. The blue plots represent how the estimator would run normally. In a normal case, the estimation works well and, the error between the measurements and estimates does not get bigger than 1cm. One aspect that is not ideal is all the small variations in the lateral flow, but we will see how it affects the open-loop simulations. The orange plots show the estimator with the disturbance decay, while the green plot shows the estimator with frozen disturbances. Based on the results in the presented cases the frozen disturbance performs much better in the long term. This is especially visible in the simulation of case2 (figure 26), where during about 12h long open-loop simulation the estimation error is not higher than about 2cm. The decaying disturbance gives acceptable open-loop estimates only for a short time period after the data is lost. It beats the disturbance freezing method in Glomma1 case3 (26) and Glomma2 case2 (25) in the short run. For the longer open-loop simulation it performs much worse. The presented results are very case dependent. If the data loss was simulated at another time instance, the results might be totally different. The decaying method would also benefit from smoother disturbance trajectories. As in Glomma1 case3, the data loss happen at a small spike in lateral flow which makes the open-loop estimation totally wrong. Smoother disturbance trajectories might however negatively affect the freezing method since as shown in the previous section the oscillations in the disturbances becomes slightly bigger. This makes that there is a higher chance for the disturbance to freeze at a peak of a wave. But again all this is very case dependent. The results show however that the open-loop simulation can be very accurate, which is a good property for using the model in the MPC.

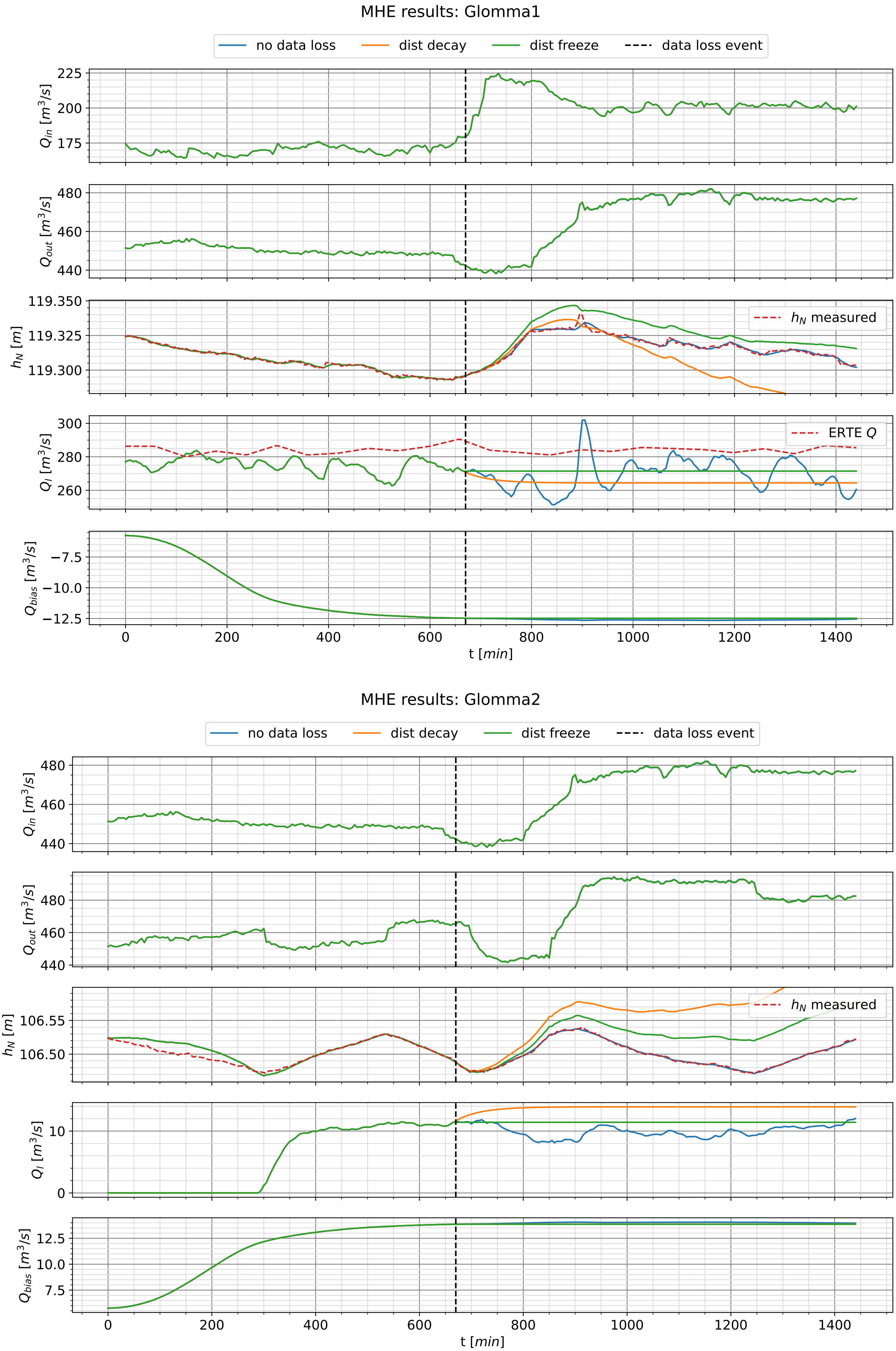


Figure 25: Simulation of data loss (case3)

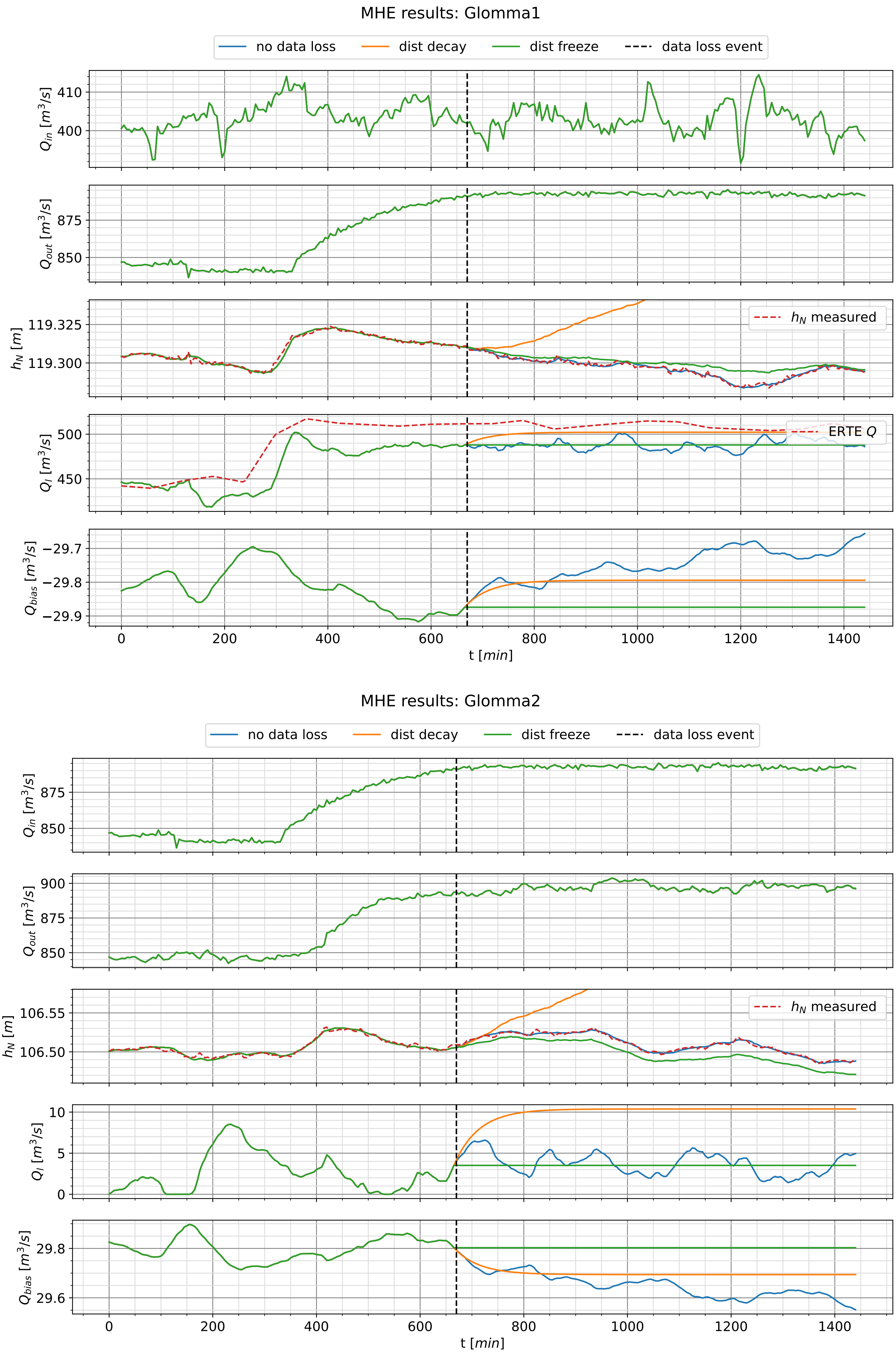


Figure 26: Simulation of data loss (case2)

### 4.1.8 Long-term performance

One of the last tests that were run was simulating the system over a longer period of time. The system was simulated over a period of 30 days. For the simulation data from case4 were used, where the flow reaches over $1000m^3/s$. This was also the first time the system was tested on this case, so it can be looked at as a validation case, that was not used for tuning the system. The goal of this test was to find some potential long-term problems with the current design and test the system for some more extreme flows that can occur. The results are presented in figure 29. The MHE has no problems with estimating the system. The error between measured and estimated values is generally under 1cm. There are few exceptions where the error reaches $2.5cm$, but those are rare and very short. The estimated lateral inflow from the Vorma river is very similar to the measured one. The average solve time for the MPC was $0.6s$, and the maximum solve time was $1.4s$.

#### The bias flow

In section 3.2, it was concluded that the main reason for this bias is the flow through the gates of a power plant. To see if this assumption is correct, the flow through the gates and the bias flow has been plotted in figure 27. The jumps in the bias flow seem to roughly correspond to the increase in the flow through the gates. One aspect of the bias flow that has not been discussed yet, is that the bias flows in the two power plants are mirrored versions of each other. This is because the estimator is dividing the flow correction needed equally between the power plants. For instance if Bingsfoss is ”missing” $50m^3/s$ of the flow, the estimator will set bias on Rånåsfoss to $-25m^3/s$ and $25m^3/s$ at Bingsfoss. Then correct the eventual error at Rånåsfoss with the lateral inflow from Vorma, which is so big that changing it has little effect on the cost function. This approach will result in a total lower objective function value because of how the objective function is formulated. This might result in some apparently unlogical results, for instance, the increase in Bingsofss's bias flow at $t = 170h$, which is caused by the opening of the gates at Rånåsfoss.

Another observed bias behavior is that it changes when the flow through gates increases, but not when it decreases. This might seem like incorrect behavior, but the lateral flow at Glomma1 matches the measurements at Ertesekken the best when the magnitude of the bias flow is at the highest at about $t = 200h$ and onward. After $200h$ the bias slowly decreases, which illustrates the effect of the term in the cost function that tries to minimize the bias flow. At $290h$ the bias jumps back, correcting the slow decrease. All that suggests that the $Q_{\text{bias}}$ approached a correct bias value needed to accurately describe the real system, and only increasing the bias is the correct solution in this case.

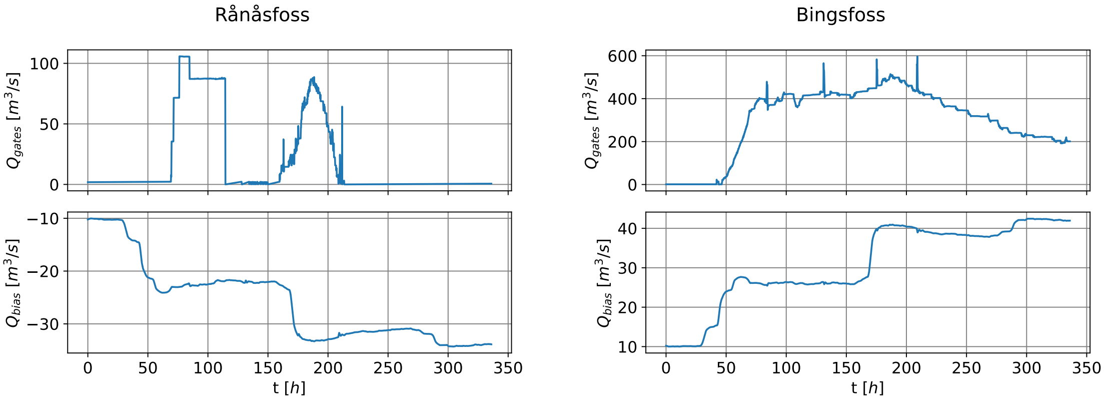


Figure 27: Comparassion between bias flow and gate flow

### Tailwater measurement

As described in section 3.6.7, the tailwater measurements were not used in the model as it resulted in the MHE not being able to find any solution to the optimization problem. To further alborate this problem, we can look at the difference between currantly estimated tailwater lavels, compared to the ones actually measured. This is presented in figure 28. For both measurements the estimates differ from the real values the most when the flow is very high. For tailwater at Funnefoss, $h_0$ at Glomma1, the difference is quite small at the beggining and end of the simualtion, when the total flow decreases. For tailwater at Rånåsfoss, $h_0$ at Glomma2, the differnce is much bigger and more noisy. Based on the error being relatively slow varying the difference could be eliminated using a bias term, in similar matter as for the flow measurement through the power plants. This will add another DOF to the model in the MHE, but will allow to utilize the information from those measurements.

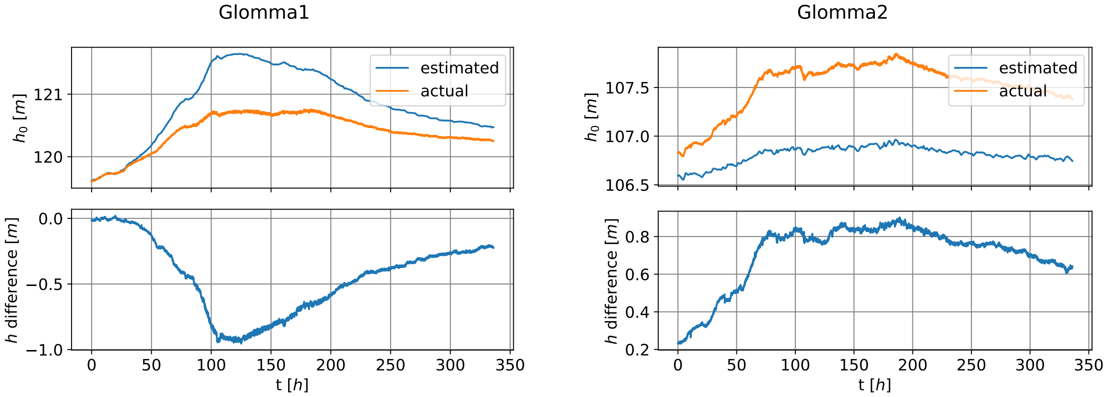


Figure 28: Difference between estimated and measured tailwater

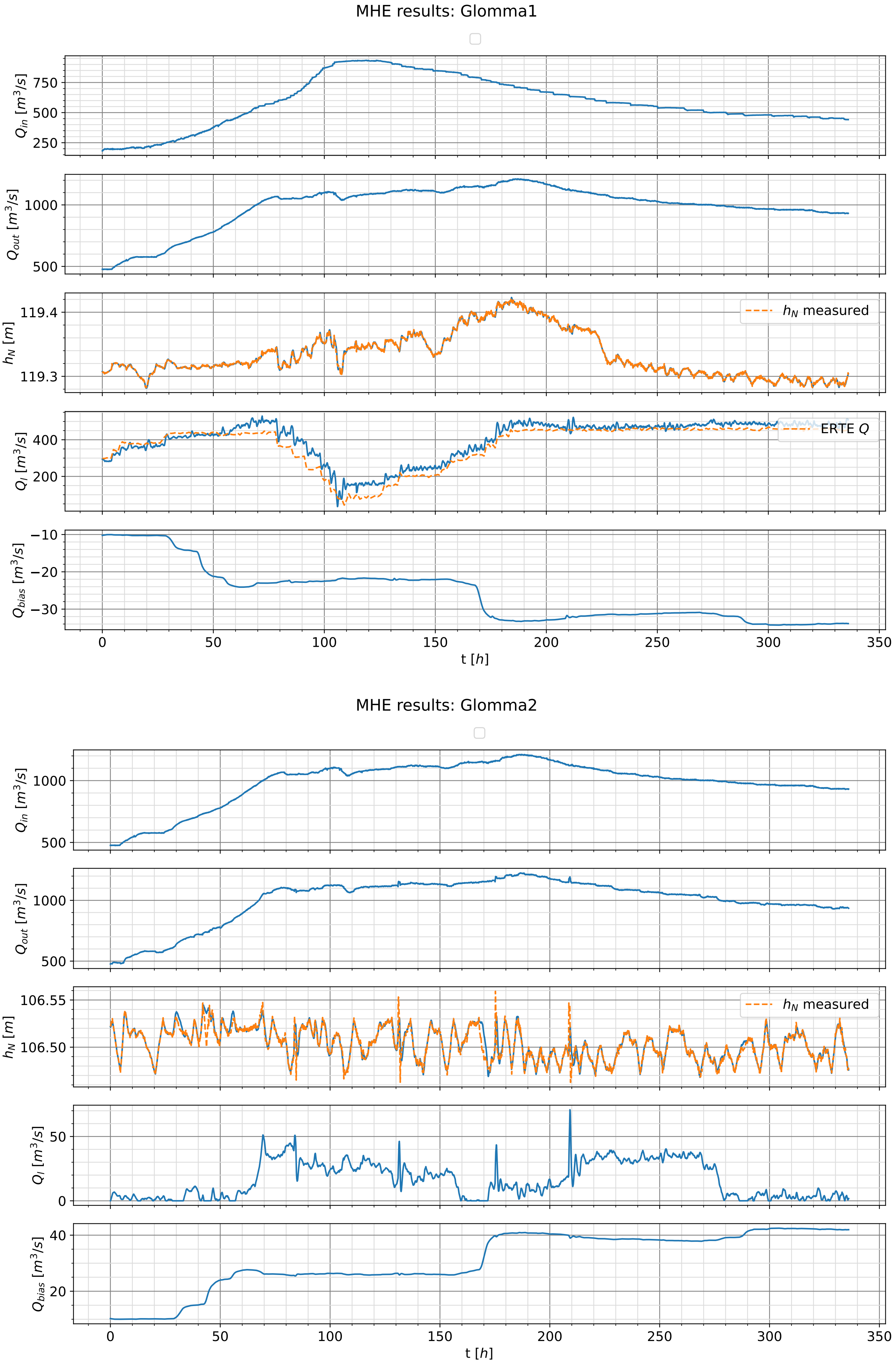


Figure 29: Long simulation results (case4)

## 4.2 MPC testing

Because of the time limitations, the controller has not been tested on the actual power plant. Testing controllers' actual performance is challenging without being able to implement the calculated control inputs on the actual plant. To demonstrate some of the controller's properties, the MPC has been tested on a simulated system. It is a good starting point for testing the MPC to detect faults without risking damage to the actual power plant. The system has been simulated using the same equations that are used in the MPC and the MHE. The inputs to the simulations were real flow measurements from Funnefoss and Ertesekken. The available flow measurements from Ertesekken are very useful here as it is used to simulate a disturbance that is very close to a real one. The setpoints in the simulations are the actual setpoints from the power plants. The flow through the power plants Rånåsfoss and Bingsfoss were calculated by the MPC.

All the controller results are presented in the following way. The first plot shows the inflow to a river segment, which is a measured disturbance in the MPC. The second plot shows the calculated control input, which is the same outflow from the river segment. The third plot shows the level at the headwater of the power plant and the setpoint. The last plot shows the absolute error, between water level and the setpoint.

### 4.2.1 Quadratic and LASSO formulation

The main goal of the first simulations was general testing of the MPC, and comparing differences between an MPC with $\ell 2$ cost function and a $\ell_{\text{asso}}$-MPC. The MPC was tuned such that it allows about the same errors as the current controller does, which is about 2cm. The simulation has been run for case3, to see controllers response to a measured distrurbance.

The results are presented in figure 30. In the plots, the MPC with a purely quadratic function is described as $\alpha = 1.0$, which turns off the $\ell 1$ regularization in the MPC. All the simulations were run with the same weights in the cost function and with the same parameters, the only difference being only the value of $\alpha$.

From the results, it is clear that the $\ell 1$ regularized MPC is able to find sparse solutions to the problem. As an example in the periods from 200 to 600 min, the value of the control inputs is constant for $\ell_{\text{asso}}$-MPC with $\alpha = 0.3$. In contrast, the values from quadratic MPC vary slightly all the time. In the simulated system, there is a constraint on how fast the water level can change, which is why the system uses some time, in the beginning, to converge to the setpoint. All these results match the described theory. In the simulation, the plots from Glomma2 are just delayed versions of Glomma1 plots and do not contribute with much new insight. This is due to running the controller in a simulated environment with a "perfect" model and no disturbance in the Glomma2 segment. The use of a LASSO-MPC comes at a cost of a slightly bigger control error, but it is a trade-off that is done for keeping a constant control input.

### 4.2.2 Simulated and real system

As an additional way to validate the results, the simulated system has been compared to the real system from the same period. A comparison is shown in figure 31. After the initial adjustment the MPC ends up with almost the same control input as the real one, and it continues up to 600 min.

The two controllers show the different approaches to handling the increase in the inflow. The MPC reacts to the incoming disturbance much earlier than the real controller. This shows the advantage the MPC controller has over a PI controller. The MPC has access to disturbance measurements and based on those and a system model it can predict how a disturbance will influence the controlled variables. The PI controller on the other hand reacts after the controlled variables have deviated from a given setpoint. As shown in the figure the MPC starts to increase the outflow a little after the inflow increases. The PI controller reacts about 150 min later, after a relatively big increase in the water level. The delay is so long because of the dead-band implemented on the PI controller, which waits until the error is big enough for the controller to react. When the inflow increased, the water level was in the lower end of the dead-band, which allowed it to increase more before the PI reacted. After that, the control inputs from both MPC and PI end up at the same values.

Since the system is simulated, with a "perfect" model in the MPC, we can not conclude that the MPC would actually be that much better than a PI controller. This should rather illustrate the difference in the control approach between an MPC and PI controller.

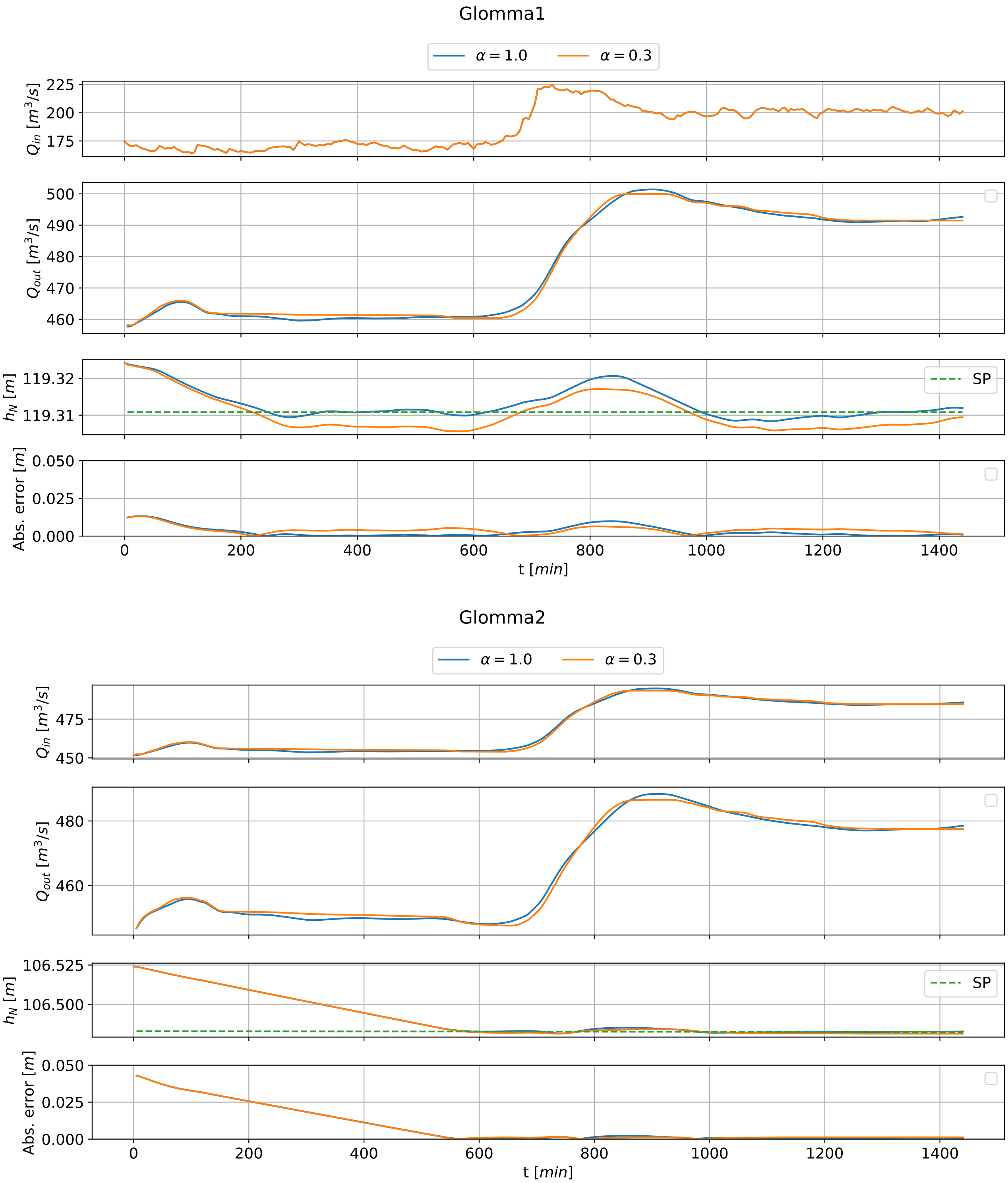


Figure 30: Comparassion between $\ell_2$ and $\ell_{\text{asso}}$ MPC formulation

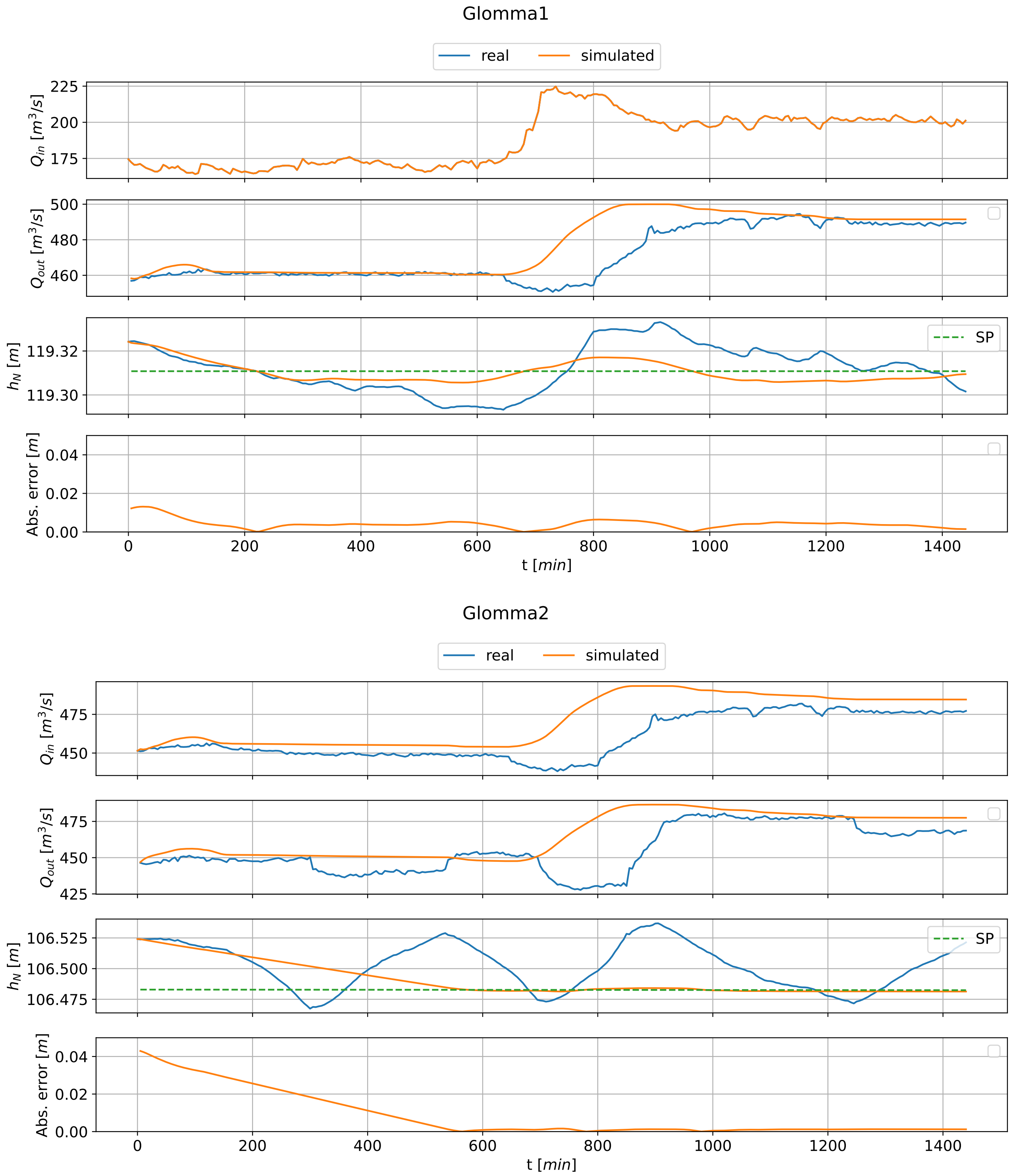


Figure 31: Comparassion between simulated $\ell_{\text{asso}}$-MPC and real system

### 4.2.3 SP step

One advantage of an MPC is the ease of implementation of some requirements in contrast to a PI controller. One such requirement is the maximum change rate of the water level. A proposed solution for implementing this requirement was constraining the water level change rate in the MPC's optimization problem formulation. To demonstrate how the MPC handles that, the system has been simulated with input data from the case3, with a 5cm step in the reference signal. For comparison, the system has also been simulated using the SP-ramp method, without any restriction on the water level change rate. The results are shown in figure 32. One difference between those two methods is that the MPC keeps the requirement of maximum change rate all the time, in contrast to the SP-ramping method where the requirement is only applied when the setpoint changes. This is most visible at the beginning of the simulation at Glomma2 where the initial water level is 4cm over the setpoint. The SP-ramp method converges fast to the setpoint, while the built-in constraint prevents the water level from changing too fast and results in a linear trajectory. This might improve the controller's performance also in other cases than changing the setpoint, as it forces the water level to be more stable. When the change in the setpoint occurs the behavior is quite different as well. The MPC with built-in constraint results in the water level increasing linearly to the new setpoint. The SP-ramp method results in the water level oscillating around the setpoint ramp. Overall, the built-in constraint follows the ramped setpoint better than the SP-ramp method itself. The reason for that is that the SP-ramp method doesn't utilize MPC's ability to predict the behavior of the system. In the current implementation, the setpoint is kept constant for the whole prediction horizon. It means that at each run the MPC will find optimal control inputs for the ”wrong” setpoint trajectory. Using built-in constraint and SP step makes that the MPC predicts control inputs that will make water level converge to the setpoint wanted in the long-term, not the temporary ”ramped” setpoint.

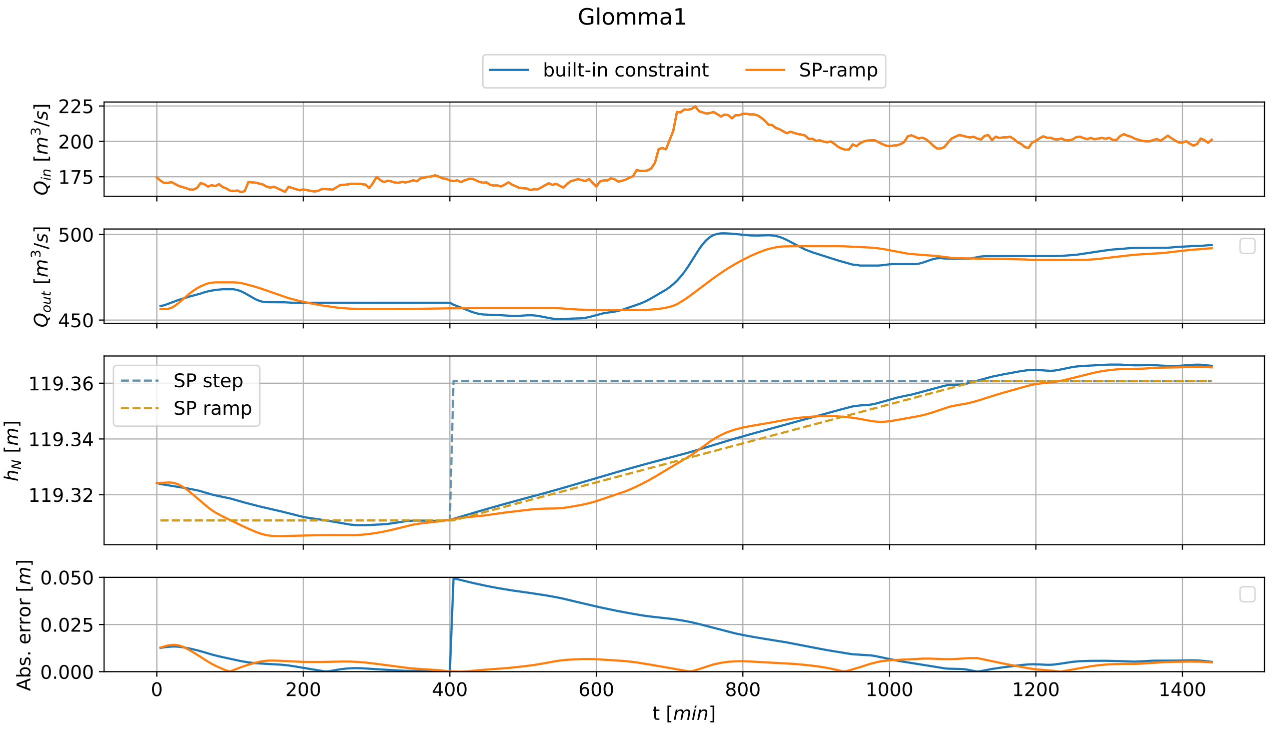


Figure 32: MPC with SP step using different ways to constrain maximum level change

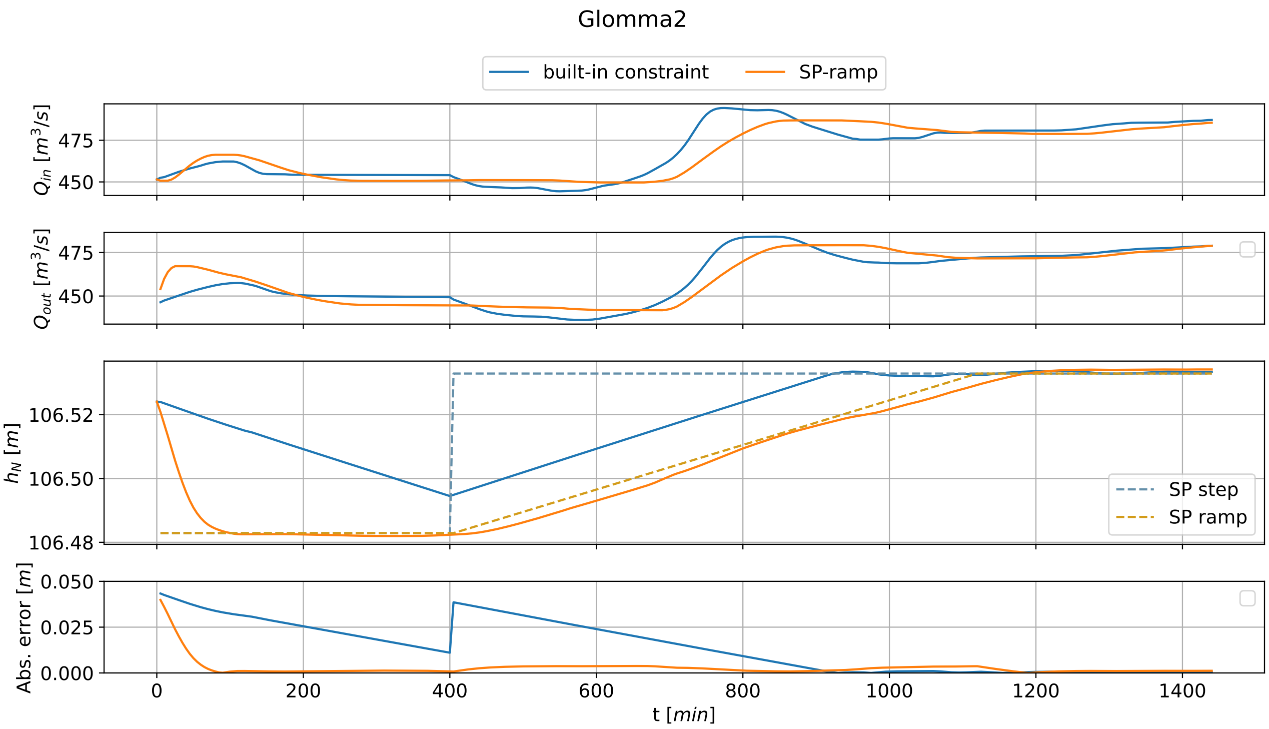


Figure 32: MPC with SP step using different ways to constrain maximum level change (cont.)

### 4.2.4 Disturbance decay

In section 3.5.4 a way to adjust how the MPC predicts disturbances have been described. The results should be similar to what was observed in the open-loop simulations of the MHE (section 4.1.7), where in some cases making the change rate decay could result in better results, than keeping the value constant. Some simulations have been run to test if this method has some effect on the performance of an MPC. Case2 has been chosen for this test as it includes a step in the lateral inflow, which will force MPC to predict the unknown disturbance, and should show the biggest difference between the two disturbance prediction methods. Results are presented in figure 33. Based on the results there is almost no difference in the MPC's performance. The method using disturbance decay is closer to the setpoint with few micrometers at some points, which in practice is not observable. This means that the predicted levels are almost the same for the chosen horizon. This corresponds with the results from section 4.1.7, where the difference between the two methods was visible only after 60 min or more (which is MPC's prediction horizon). This means that the difference might be bigger if a longer prediction horizon were used. This will come with a much longer solve time for the MPC, and was not tested. Therefore constant disturbance in the prediction horizon has been mainly used in other simulations as it is slightly easier to implement and doesn't risk violating the hard constraints by changing the disturbance's values.

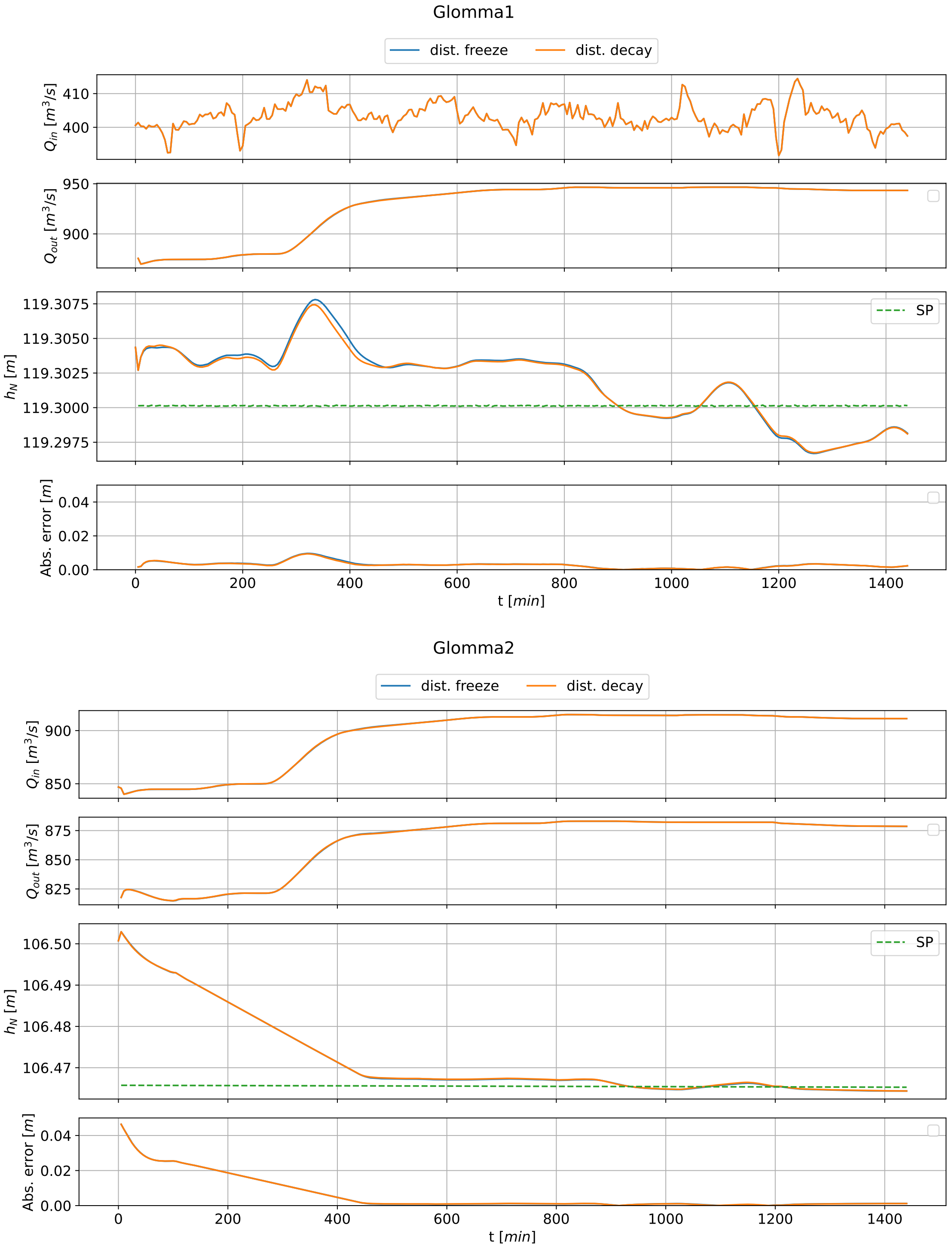


Figure 33: case2

### 4.2.5 Long-term performance

As a final test of the MPC, it has been simulated for a period of 31 days, to find some potential long-term problems. The case used is the same one that has been used for long-term testing of the MHE (section 4.1.8). The results are presented in figure 34. The actual values from this time period have also been plotted for comparison. Plots from segment Glomma2 have been omitted as without any disturbance it resulted in a ”perfect” performance.

On the presented plots, the simulated system performs much better than the real system in terms of the control error. This is to be expected as the MPC uses the perfect model of the simulated system, and doesn't say a lot about the actual performance. A better indication of the performance can be seen by looking at the control inputs. The simulated model uses very similar control inputs to the ones that have actually been used, which indicates that the control inputs calculated by the MPC are somewhat correct. The difference is biggest when the total flow is big, which is because with so big flows more disturbances that are not reflected in the model are acting on the system. The main difference between the responses is the reaction time on the incoming flow increases, which is what has been described in more detail in section 4.2.2. For this simulation the MPC had an average solve time of 2.4$s$, and a maximum solve time of 18.6$s$. This indicates how fast control $T_s$ can be chosen in the final implementation.

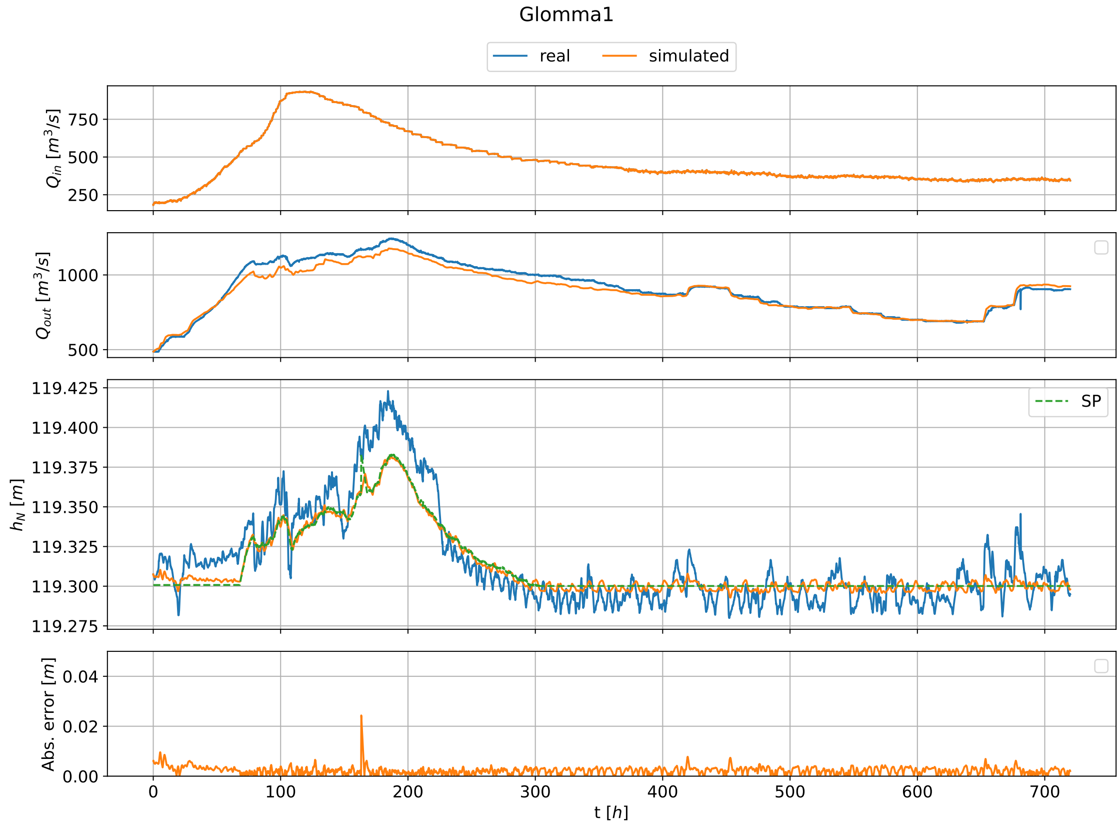


Figure 34: Long MPC simulation results

# 5 Discussion

The first step in the implementation of the MPC was the verification of the system model. This was mainly done by testing the MHE on different cases. The main difficulty with the river model used, was the choice of parameters, like river slope, river width, and friction coefficient. Access to digital maps makes this process easier. But, as presented in sections 4.1.2, 4.1.3, 4.1.4, the parameters are not ideal right out of the box and need some tuning for the model to perform well. It would be interesting to see if measuring river bed and river width accurately would result in much better performance. However, such a process can quickly become time-consuming and expensive and would make the control solution proposed in this thesis not vaiable. In the end, the presented solution achieved good results, using only data from digital maps to describe the model's parameters, which makes the proposed solution much easier to implement. Making the tuning of parameters even easier could be achieved using parameters estimation. A simple parameter estimation solution was tested in this thesis but didn't provide satisfactory results. It doesn't necessarily mean that it is not a valid solution, but more work is required to achieve this.

Creating the MHE was the most time consuming part of developing the control system. This step combined developing the MHE and confirming the model's validity, which resulted in testing a lot of different solutions. The MHE showed satisfactory results, with the estimation error being on average under 1cm and the main disturbance, the lateral inflow from the Vorma river, is estimated very close to the actual measurements. The open-loop simulation gave surprisingly good results in some cases. This further confirmed the models' validity but also shows robustness to potential corrupted or lost data.

The main disadvantage with the proposed solution is discarding the measurements of the tailwater level. This was done because the MHE wasn't able to find good solutions to the optimization problem when both the headwater and tailwater measurements were included. This is discarded measurements contain information that can be used to improve the estimation. Discarding those measurements was not an ideal solution, but at the time of development, it was the easiest option that made it possible to continue developing and testing other aspects of the system. Later the estimated tailwater levels were compared to the measured ones (section 4.1.8). The results showed that in some cases the difference is very small, and the error is slowly varying. This means that a slow varying bias term similar to the one used in the power plant flow can be used to compensate for this. The reason for the problem is probably that the model used is not accurate enough to accurately describe multiple water levels on the river with the provided DOFs. In the cases of the headwater level, it can be seen that the lateral flow is used by the estimator to adjust the flow to achieve the correct water level. The tailwater measurements are taken at the first grid cell in the model, and the current model implementation has no DOF that can be used to adjust for small differences in the tailwater level. In contrast, the headwater level can be adjusted both by bias flow and the lateral inflow. Therefore adding one lateral inflow per level measurement can be a valid solution. But it is important to remember that adding a lot of DOFs to the model to match the measurements can also be problematic as it might results in overfitting the model, so it should be kept to the minimum. In addition, the river parameters have a great effect on how many adjustments in the flow are necessary. Finding parameters that worked for headwater measurements was not a very easy process and finding parameters that work for multiple measurement points might be too challenging with the current method.

On the other hand, the fact that the MHE gives good results using only headwater measurement is a good property. The control solution presented in this thesis is meant to be able to handle different river systems. The dam headwater measurements are almost always available since it is the water level that is often controlled. But it doesn't mean that water level measurements will be available from upstream measurements stations. As an example, the measurement station at Ertesekken used in this thesis has only water flow measurements (at least at NVE Sildre). In this thesis, there were multiple powerplants in cascade which provided a lot of data, but this is not the case for all river systems. Therefore it is good to see that the MHE works with only the headwater measurements.

There are many different ways in which the model could be implemented. In retrospect, I see that multiple aspects could be chosen differently to achieve better estimation performance. For instance using two separate control systems for each river segment, instead of one big cascade system might be a better solution. In the start phase of the thesis, a cascade system was chosen to allow the controller to consider the actions of both power plants to achieve optimal control. However, the transport delay between the power plants is big enough for the power plants can be controlled individually. By using two separate models with the flow from the upstream power plant being a known disturbance and the downstream plant being controlled, the control system would be significantly simplified. This would result in a faster computation time for each power plant. In addition potential infeasibility at one segment would not result in the whole system failing, and the debugging process would be simplified. The division of bias flow between the power plant described in section 4.1.8 would also be eliminated. The bias term would be used as the intended correction of the gate flow at a single power plant.

The final MPC was able to satisfy all the requirements that were specified. The controller was able to find sparse solutions and lock the control input without using a dead-band. Using a dead-band is not ideal as it leaves the system practically uncontrolled while the water level is within some bounds. That's why in the simulations the MPC responded faster to incoming disturbances (section 4.2.2), which resulted in smaller water level variations. The MPC has a faster response time also because of the access to upstream flow measurements and a river model, which allows it to predict future dynamics, and react to the disturbances before they affect the water level. The MPC included a limit on water change rate, which showed better results than the method of SP ramping. Advantage of this implementation is that the MPC tries to keep that limit under all circumstances, not only when the setpoint changes.

To test the system on an actual power plant was a goal for this thesis, but because of time limitations, it was not achieved. Testing the solution thoroughly in simulations was however important to find any potential flaws, that could damage the real system. Without those tests, it is difficult to say how well the MPC would perform in an actual power plant. Based on the simulation results, the developed MPC has the potential to replace and improve the current control solution. This is especially in more challenging systems where a simple controller is not good enough. A drawback with the presented solution is of course amount of work it takes to implement compared to a PI controller. Generally, the MPC is harder to implement than a PI-controller, but the implementation of more advanced control strategies is in fact easier and can be done in a more systematic way in an MPC. For instance, keeping constant control input and limiting the change rate of the controlled variables. In this regard the MPC is much more customizable, as the optimization problem can be modified in a lot of different ways, to achieve different behaviors. The developed implementation was kept as general as possible to make the MPC solution applicable to other river systems. This should make the process easier, but as described earlier, finding correct parameters and tuning can still be challenging.

# 6 Conclusion

## 6.1 Summary and conclusions

In this thesis, a model-based control system, that can be used to control water level at a run-of-river power plant has been presented. The model used to describe the river system was St. Venant equations. The SVEs were able to accurately describe model dynamics and can be used in an MPC to predict future behaviors. To achieve that the MPC utilizes flow measurements taken upstream of the power plant, and disturbance estimates provided by an MHE. The MPC is a very versatile solution that allows using more advanced control mechanisms in a systematic and thoughtful way. The MHE can also be used in an open-loop mode with good results in case of data loss or corrupted measurements. This makes the solution more robust in such cases. The main drawback of the presented solution is its complexity, which makes it longer to develop and is harder to maintain and debug. The developed MPC can not be implemented on a standard PLC unit and will require more work to integrate into the current control system. The MPC has only been tested on the simulations and tests on an actual power plant are necessary to confirm its performance.

## 6.2 Further work

- In the presented solution there are no fail-safe methods implemented. In some cases, the optimization problem ends up at an infeasible or nearly infeasible point, and the optimizer is not able to find any solution or finds a bad solution. This happens rarely but is something that should be taken into consideration, and some checks on the quality of the solution should be implemented. Slacking some constraints or reinitializing the system for one iteration might solve the problem temporarily, but if it happens often the model equations should be reevaluated to find a long-term solution. For this reason another controller should be kept as a backup in case of MPC failure.

- In the simulations the control system was tested with the loss of water-level measurements of the headwater. Another case that can be tested is a loss of the upstream flow measurements. In such a case, the controller would not be able to predict the results of an incoming flow and would need to react more like a current control based on the changes in the water levels.
- An obvious next step in developing the presented solution is to test it on an actual powerplant. The presented results look good in simulations, but it is difficult to say how the control system will perform in a real system. The solving time will most probably differ from the ones presented here, as computers' performance will decide how long it takes to find a solution to the optimization problem. This will affect the choice of control and prediction sampling times. The code used in this thesis was made for simulations, so some changes are necessary to implement in a real system, for instance receiving live data instead of the historical data.

# References

[1] Milo Brandt (https://math.stackexchange.com/users/174927/milo-brandt). *A "new" general formula for the quadratic equation?* Mathematics Stack Exchange. URL: https://math.stackexchange.com/q/3619289 (visited on May 30, 2022).

[2] *APMonitor COLDSTAART.* URL: https://apmonitor.com/wiki/index.php/Main/OptionApmColdstart (visited on May 31, 2022).

[3] AKIO ARAKAWA and VIVIAN R. LAMB. "Computational Design of the Basic Dynamical Processes of the UCLA General Circulation Model". In: *General Circulation Models of the Atmosphere.* 1977.

[4] Logan Beal et al. "GEKKO Optimization Suite". In: *Processes* 6.8 (2018), p. 106. ISSN: 2227-9717. DOI: 10.3390/pr6080106. URL: https://dx.doi.org/10.3390/pr6080106.

[5] Lorenz T. Biegler. *Nonlinear programming: concepts, algorithms, and applications to chemical processes.* MOS-SIAM series on optimization. Philadelphia: Society for Industrial and Applied Mathematics: Mathematical Programming Society, 2010. ISBN: 9780898717020.

[6] Matthew Chan et al. *Optimization with absolute values.* Cornell University Computational Optimization Open Textbook. URL: https://optimization.cbe.cornell.edu/index.php?title=Optimization_with_absolute_values#Sign_Constraints_are_Satisfied (visited on June 6, 2022).

[7] Ven Te Chow. *Open-channel hydraulics.* eng. New York, 1959.

[8] Nuno M. C. De Oliveira and Lorenz T. Biegler. "Constraint handing and stability properties of model-predictive control". In: *AIChE Journal* 40.7 (1994), pp. 1138–1155. ISSN: 0001-1541. DOI: 10.1002/aic.690400706.

[9] Olav Egeland and Jan Gravdahl. *Modeling and Simulation for Automatic Control.* Jan. 2002.

[10] Bjarne Foss and Tor Aksel N. Heirung. *Merging Optimization and Control.* 2016.

[11] Marco Gallieri. *Principles of LASSO MPC.* Mar. 2016. ISBN: 978-3-319-27961-9. DOI: 10.1007/978-3-319-27963-3_3.

[12] *GEKKO docs: Global options.* URL: https://gekko.readthedocs.io/en/latest/global.html (visited on May 31, 2022).

[13] *GEKKO docs: Tuning parameters.* URL: https://gekko.readthedocs.io/en/latest/tuning_params.html (visited on May 24, 2022).

[14] Ben R Hodges. "Conservative finite-volume forms of the Saint-Venant equations for hydrology and urban drainage". eng. In: *Hydrology and earth system sciences* 23.3 (2019), pp. 1281–1304. ISSN: 1027-5606.

[15] "Selected Topics on Constrained and Nonlinear Control". In: ed. by T. A. Johansen. 2011. Chap. Introduction to Nonlinear Model Predictive Control and Moving Horizon Estimation.

[16] K Mahmood. *Unsteady flow in open channels : 1.* eng. Fort Collins, Colo, 1975.

[17] *Manning's n Values.* URL: http://www.fsl.orst.edu/geowater/FX3/help/8_Hydraulic_Reference/Mannings_n_Tables.htm (visited on May 24, 2022).

[18] D.Q. Mayne et al. "Constrained model predictive control: Stability and optimality". In: *Automatica* 36.6 (2000), pp. 789–814. ISSN: 0005-1098. DOI: 10.1016/s0005-1098(99)00214-9.

[19] German Ardul MUNOZ-HERNANDEZ, Sa'ad Petrous MANSOOR, and Dewi Ieuan JONES. *Modelling and Controlling Hydropower Plants.* Springer-Verlag, 2013.

[20] Torbjørn K Nielsen. *Dynamisk dimensjonering av vannkraftverk.* nob. Trondheim, 1990.

[21] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization.* 2e. New York, NY, USA: Springer, 2006.

[22] Douglas G. Robertson, Jay H. Lee, and James B. Rawlings. "A moving horizon-based approach for least-squares estimation". In: *AIChE Journal* 42.8 (1996), pp. 2209–2224. ISSN: 0001-1541. DOI: 10.1002/aic.690420811.

[23] Seyed Mostafa Safdarnejad et al. *Initialization strategies for optimization of dynamic systems.* 2015. DOI: https://doi.org/10.1016/j.compchemeng.2015.04.016. URL: https://www.sciencedirect.com/science/article/pii/S0098135415001179.

[24] Felix Fiedler Sergio Lucia. *Orthogonal collocation on finite elements.* 2021. URL: https://www.do-mpc.com/en/latest/theory_orthogonal_collocation.html (visited on Apr. 24, 2022).

[25] *Three ways to speed up model predictive controllers.* URL: https://www.mathworks.com/content/dam/mathworks/white-paper/3-ways-to-speed-up-model-predictive-controllers.pdf (visited on May 27, 2022).

[26] *Vannkraft.* Feb. 2022. URL: https://www.nve.no/energi/energisystem/vannkraft/ (visited on May 30, 2022).

[27] M. Vukov et al. "Real-time nonlinear MPC and MHE for a large-scale mechatronic application". In: *Control Engineering Practice* 45 (2015), pp. 64–78. ISSN: 0967-0661. DOI: 10.1016/j.conengprac.2015.08.012. URL: https://dx.doi.org/10.1016/j.conengprac.2015.08.012.

[28] Liubomyr Vytvytskyi et al. "Modeling for Control of Run-of-River Power Plant Grnvollfoss". In: Nov. 2015, pp. 237–246. DOI: 10.3384/ecp15119237.

NTNU

Norwegian University of
Science and Technology

VOITH