Privacy-Preserving Distributed Optimising Control of Water Supply

Master thesis
MSc. Electronic Systems
Aalborg University
Spring 2024

Abstract

Society relies on critical infrastructure, which, due to internet connections, is subject to an increasing threat of cyber attacks. This report presents a method for designing distributed optimising controllers for critical infrastructure, which keeps local cost functions and constraints private in case of passive and eavesdropping attacks, while protecting against specific active attacks. The method is based on distributing an Model Predictive Control problem using Alternating Direction Method of Multipliers, which is made privacy-preserving through the use of Secure Multi-Party Computation.

A Water Distribution Network is emulated in a laboratory, where PI-controllers are implemented such that the Water Distribution Network can be modelled while abstracting from pump, valve, and pipe dynamics.

Future actuation commands to the PI-controllers are found by optimising a non-convex Model Predictive Control problem with a cost function consisting of the electricity bill of the system, subject to system constraints. The optimisation problem is solved in a distributed manner, where the communication between the stakeholders is encrypted using Shamir's Secret Sharing Scheme to obtain privacy-preserving features.

The privacy-preserving controller finds minima with costs equal to an ordinary optimising controller. Thereby, the only cost of the privacy-preserving controller is longer computation time and increased communication.

Master thesis (30 ECTS)
Education: Master of Science in Engineering (Electronic Systems)
Project period: 1st of February 2024 – 31st of May 2024

Authors: Fie Ørum, Lau Lauridsen, Simon Johansen
Supervisors: Carsten Skovmose Kallesøe (AAU / Grundfos), Rafal Wisniewski (AAU)

AALBORG
UNIVERSITY

The Technical Faculty of IT and Design
Study board of Electronics and IT
Frederik Bajers Vej 7A
9220 Aalborg Øst
es.aau.dk

Preface

This project report is written as the Master thesis for the MSc. in Electronic Systems at Aalborg University. The report documents the main findings of the investigating on how to implement privacy-preserving control algorithms for Water Distribution Network.

A special thanks to Kathrine Tjell Mølgaard for sharing her unpublished slides [1], which provided invaluable inspiration for the project, despite her not participating in collaboration during the project. Furthermore, thanks to PhD student Saruch Satishkumar Rathore for sharing his know-how regarding the Smart Water Infrastructures Laboratory at Aalborg University.

Fie Ørum
farum19@student.aau.dk
fieoerum@outlook.com

Lau Lauridsen
llaur19@student.aau.dk
lau.lauridsen@hotmail.com

Simon Johansen
sjohan19@student.aau.dk
simonjohansen2k@gmail.com

Summary

Samfundet er afhængigt af kritisk infrastruktur, såsom elnet, vandforsyning og varmeforsyning. I mange tilfælde er infrastrukturen forbundet til internettet, hvilket øger risikoen for cyberangreb.

Denne rapport undersøger hvordan der kan udvikles modelprædiktive regulatorer til vandforsyningsnetværk, uden at informationer om pumpestationer såsom maksimalt flow og effektforbrug, samles et sted. Dermed mindskes risikoen for at alle pumpestationernes informationer lækkes i tilfælde af cyberangreb.

Rapporten beskriver hvordan et laboratorie benyttes til at emulere et vandforsyningsnetværk. Der udvikles PI-regulatorer til pumper og ventiler, som styres ved hjælp af kommandoer sendt over ModbusTCP protokollen. PI-regulatorerne lineariserer pumperne og ventilerne, således der kan laves en model af vandforsyningsnetværket i laboratoriet, baseret på strømninger og volumener. I laboratoriet skiftes elprisen hvert tiende minut, fremfor hver time som i virkeligheden, hvormed der kan simuleres i accelereret tid.

En modelprædiktiv styring som minimerer elregningen for vandforsyningsnetværket udvikles, hvilket i simulering resulterer i en 18.3 % reduktion af elregningen, sammenlignet med en tænd/sluk regulering af pumperne. Referencesignalerne til PI-regulatorerne der styrer pumperne beregnes i den modelprædiktive styring ved løsning af et optimeringsproblem. Optimeringsproblemet kan løses distribueret ved brug af konsensus ADMM. ADMM-algoritmen producerer i simulering løsninger med samme elregning som den modelprædiktive styring.

Interessenterne som deltager i ADMM-algoritmen skal i flere tilfælde beregne en sum af hver af deres bidrag. Interessenternes bidrag skjules for hinanden ved hjælp af Shamir's Scheme. For at denne beregning er fuldstændig sikker, benyttes moduloberegninger. Dermed skjules pumpestationernes begrænsninger og effektforbrug for hinanden.

Styringsalgoritmen implementeres i laboratoriet, hvor den som forventet på baggrund af simulering, fungerer identisk med den modelprædiktive styring, hvilke er det bedst opnåelige resultat. Den primære ulempe ved implementering af styringsalgoritmen er dermed øget kommunikation mellem interessenterne.

Contents

Introduction 1

Modern society relies on critical infrastructure such as energy, water networks, and communication to keep citizens safe. Even failures over a short period can have serious consequences, such as a lack of water for firefighting, loss of communication with emergency departments, or traffic breakdowns. Some infrastructures are dependent on an internet connection to function or utilise such a connection to optimise their operation [2].

Connecting critical infrastructure to the internet has heightened the susceptibility to cyber attacks [3, p. 2], resulting in an increasing threat, as cyber attacks are growing both in sophistication and number [4, p. 1]. Meanwhile, actors have shown a willingness to execute complex physical attacks on infrastructure, such as the attack on Nord Stream 2. Thereby, complex cyber attacks on infrastructure must be expected, making it crucial to obtain a high level of cyber security.

In some literature, [5, pp. 14-15], cyber attacks are categorised as passive, eavesdropping and active. Passive attacks infect a subsystem, letting it follow protocol while using it to gather information. Eavesdropping refers to attacks where a communication line is monitored to gather information. Active attacks infect a subsystem or communication line and make it deviate from protocol.

At the control and automation section at Aalborg University (AAU), it has been researched how it is possible to prevent passive and eavesdropping adversaries from obtaining information from optimising control systems [1], [5]–[7]. In addition, the section has researched within the fields of district heating, wastewater networks, and Water Distribution Networks (WDNs), utilising the world-class Smart Water Infrastructures Laboratory (SWIL) at AAU [8].

Tjell [5] has investigated the use of Secure Multi-Party Computation (SMPC) and Alternating Direction Method of Multipliers (ADMM) for hiding cost functions and local constraints in optimisation problems. Løvemærke [7] has demonstrated that Model Predictive Control (MPC) problems can be solved in the cloud without leaking private information using SMPC.

Finally, Tjell [1] has, in a non-published set of slides, suggested how distributed optimising control can be applied to a simple WDN, such as the one shown in Figure 1.1. Meanwhile, the pump stations' constraints and cost functions are kept private, using SMPC and consensus ADMM. The suggested control scheme aims to minimise the electricity bill of the WDN while complying with the constraints of the water tower's capacity, the pump stations' maximum flow, and the pump stations' Total Yearly Extraction Limit (TYEL).

The significant fluctuations in electricity prices, as shown in Figure 1.2, and the use of elevated water towers, presents an excellent opportunity to perform control to minimise the electricity bill.

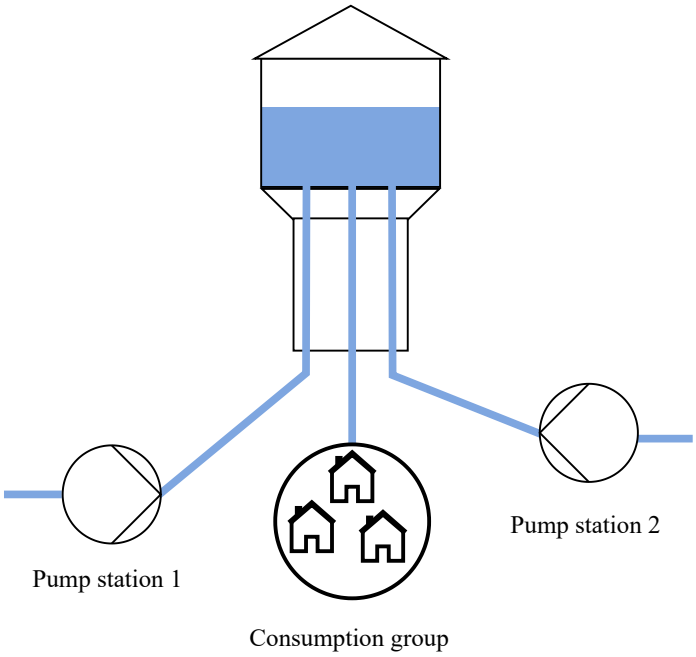


Figure 1.1. Sketch of a simple WDN with two pump stations, one consumption group, and an elevated water tower.

1.1 Problem Statement

This study sets its purpose as implementing privacy-preserving control, inspired by the suggestions in [1]. Therefore, the problem statement:

> *How can a distributed privacy-preserving optimising control algorithm be applied to a WDN such as the one shown in Figure 1.1?*

In this project, a privacy-preserving controller is defined as a controller where an adversary, cannot discern cost functions and constraints of non-infected pump stations by infecting another pump station or eavesdropping on a communication line. Optimising refers to minimising the WDN electricity bill by planning when to pump using MPC. The setup shown in Figure 1.1 is emulated by building a miniature WDN in the AAU SWIL to test the developed controller.

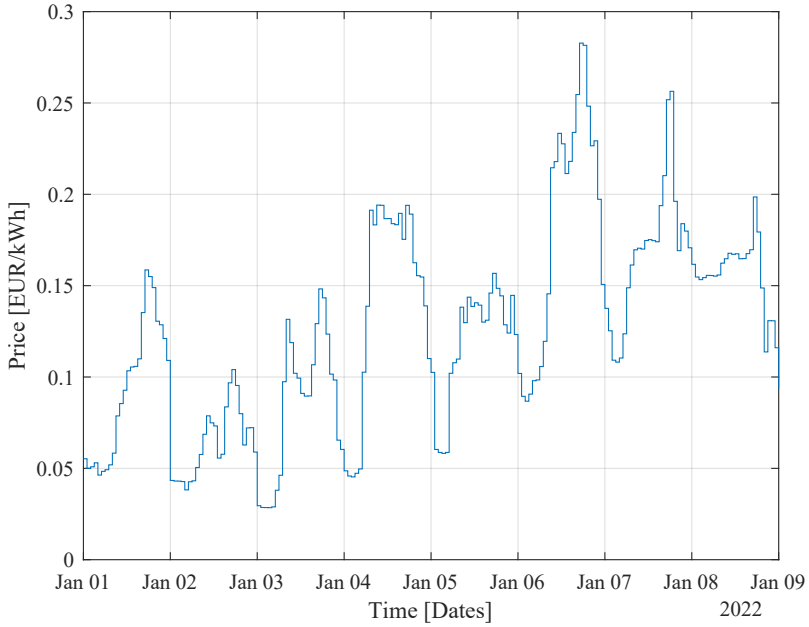


Figure 1.2. Electricity price in N1 distribution grid including network tariff (B-low) excluding taxes and subscription for the 1 to 9 of January 2022. Based on data from ENTSO-E [9] and N1 [10].

Laboratory Setup 2

The introduction stated a need for distributed privacy-preserving control algorithms for Water Distribution Networks (WDNs). This chapter describes a laboratory setup using the Smart Water Infrastructures Laboratory (SWIL) at AAU, which will be used to test the developed control algorithm. The setup will be made as simple as possible, as controller design of complex WDNs is not the scope of this project. Low-level pump and consumption flow controllers are implemented, such that the model for the optimising controller described in the next chapter can be based on flows from the pump stations, flow into the consumption group, and volume of water in the tower.

2.1 Setup

The SWIL is modular, meaning it can be arranged to fit the specific needs of different projects [8]. For this project, it is chosen to build a simple WDN like the one shown in Figure 1.1. To do this, five modules of three different types are used: Two pump stations, a pipe module to emulate pipe resistance, and two consumer modules, one to emulate the water tower and one to emulate the consumption group. A simplified diagram of the setup is shown in Figure 2.1, while detailed piping and instrumentation diagrams for the individual modules can be found in Appendix B. The main simplification of the diagram is that pumps and valves in parallel are drawn as a single component, and only selected sensors and actuators are shown.

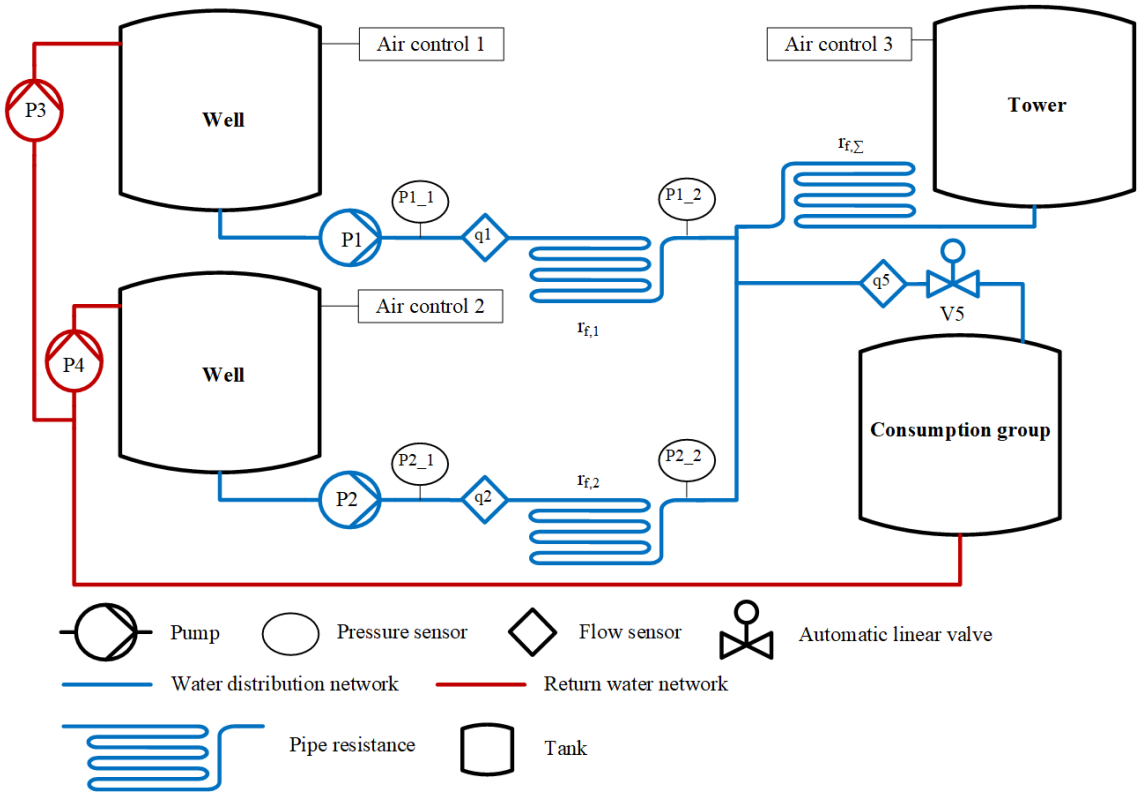


Figure 2.1. Simplified diagram of the laboratory setup.

Figure 2.1 shows a WDN and return water network, both consisting of pipes, pumps, valves, and a cylindrical water tower. The WDN is used to test control algorithms, while the return water network is present to pump water back into the wells. A switching-based control algorithm is implemented as a controller for the return water network, which will not be described in further detail.

The pipe resistance in and out of the tower $r_{f,\Sigma}$ is not desired, as such resistance is expected to be insignificant in real-world WDNs. It is present due to small non-changeable pipe dimensions at the bottom of the tower tank. This issue occurs, as the consumer modules are not designed to have water flowing into the bottom of the tank. Nevertheless, the setup is used this way, as it leads to the pump power consumption being a function of the water level in the tower, which is a system dynamic, as in some real-world applications.

The air control in Figure 2.1 increases pressure within the tanks, emulating elevation. Increasing the pressure inside both the tower and the pump station wells makes it possible to emulate different elevations between each pump station and the tower. All tanks are equipped with differential pressure sensors, not shown in the figure, measuring the height of the water within the tanks.

2.1.1 Communication with Sensors and Actuators

All sensors and actuators on each module are controlled using a Raspberry Pi 3 B+ with IO boards, which acts as a soft PLC, as shown in Figure 2.2. The Raspberry Pis run a 32-bit version of Raspberry Pi OS besides acting as a soft PLC. The PC on the figure is connected to all Raspberry Pis using ethernet and can be used to run Supervisory Control And Data Acquisition (SCADA). The sensor readings and actuator settings can be read and set by all PCs and Raspberry Pis in the network by sending Modbus over TCP commands to the soft PLCs. All sensors and actuators can be accessed through a Simulink program using ModbusTCP. However, in this project, the controllers are implemented as Python scripts communication over ModbusTCP. An example of the implementation of the communication can be found in Appendix C.

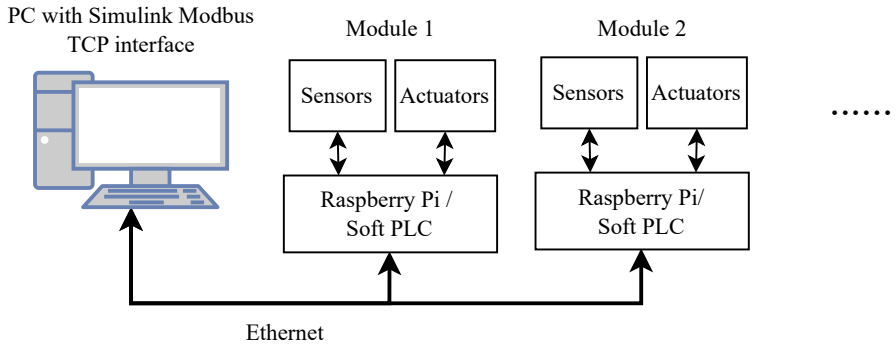


Figure 2.2. Communication setup in the SWIL.

2.1.2 Low-Level Controllers

In this section, low-level controllers are developed, with the intention of making it possible to derive a model for a high-level controller based on flows, thereby abstracting from pump, pipe, and valve dynamics. These controllers consist of a pump station controller, controlling the flow from the wells into the tower, and a controller to generate the flows into the consumer.

A Proportional Integral (PI) controller for each pump station is implemented as a Python script, which can run on both the PC and the Raspberry Pis. The script communicates with the soft PLCs over ModbusTCP using the Python library pyModbusTCP as described in Appendix C. The controllers are hand-tuned, aiming for low oscillations and rise time within a minute. The performance of these controllers is shown in Figure 2.3, where it is observed that both controllers oscillates to a large degree. However, these oscillations are deemed to be a result of the flows being sensitive to small changes in actuation and a cross-coupling between the two pump stations, caused by pumping water into the tower through the same entrance. Nevertheless, the mean flow follows the reference within approximately 30 s, thus the design is deemed acceptable. A detailed design description can be found in Appendix D.

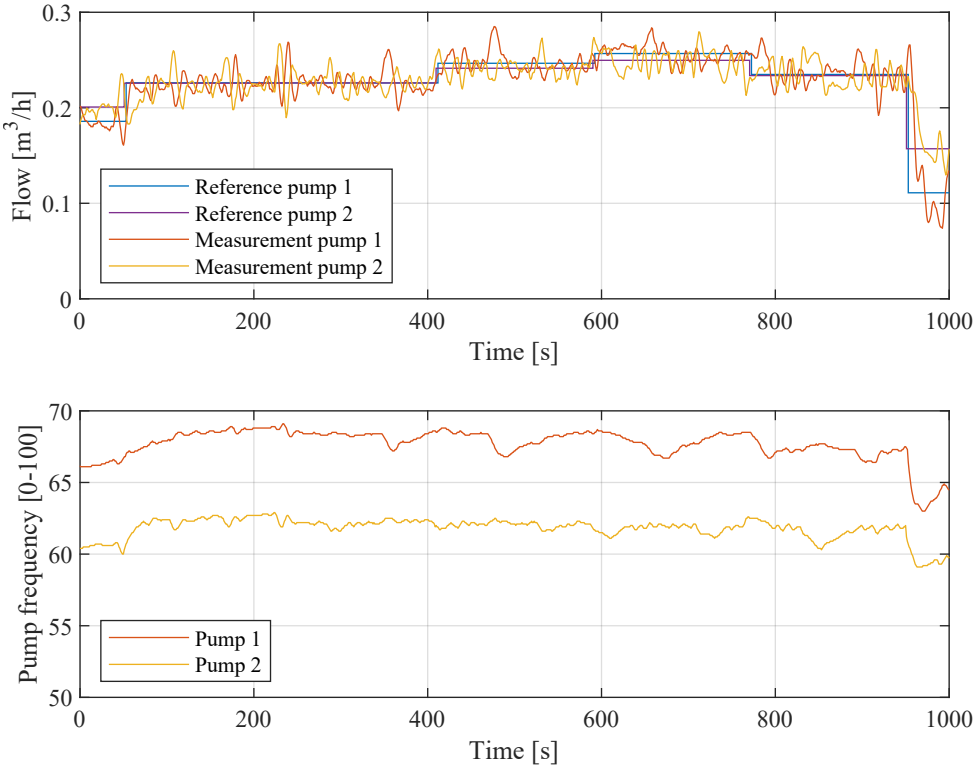


Figure 2.3. Measured step response of both pump station controllers running simultaneously.

Next, a controller for the consumption group is designed. This corresponds to controlling the flow into the consumer by actuating the valves at V5 in Figure 2.1. At V5, two valves and flow sensors are placed in parallel, where a PI controller is designed for each valve. The controllers are tuned for low oscillations and rise time within a couple of minutes. It was found that the valves perform best when the flows are neither high nor low. For this reason, the reference for each controller is set to half the reference from the high-level controller if the flow is above $0.275\,\mathrm{m}^3\,\mathrm{h}^{-1}$. Otherwise, one valve is assigned all the flow. The performance is shown in Figure 2.4, where low oscillations and a rise time below 50 s are observed. A detailed design description can be found in Appendix E.

In addition to the low-level controllers, a supervisory safety controller is implemented, shutting off pumps and closing valves, to prevent breach of minimum and maximum water levels in the tanks.

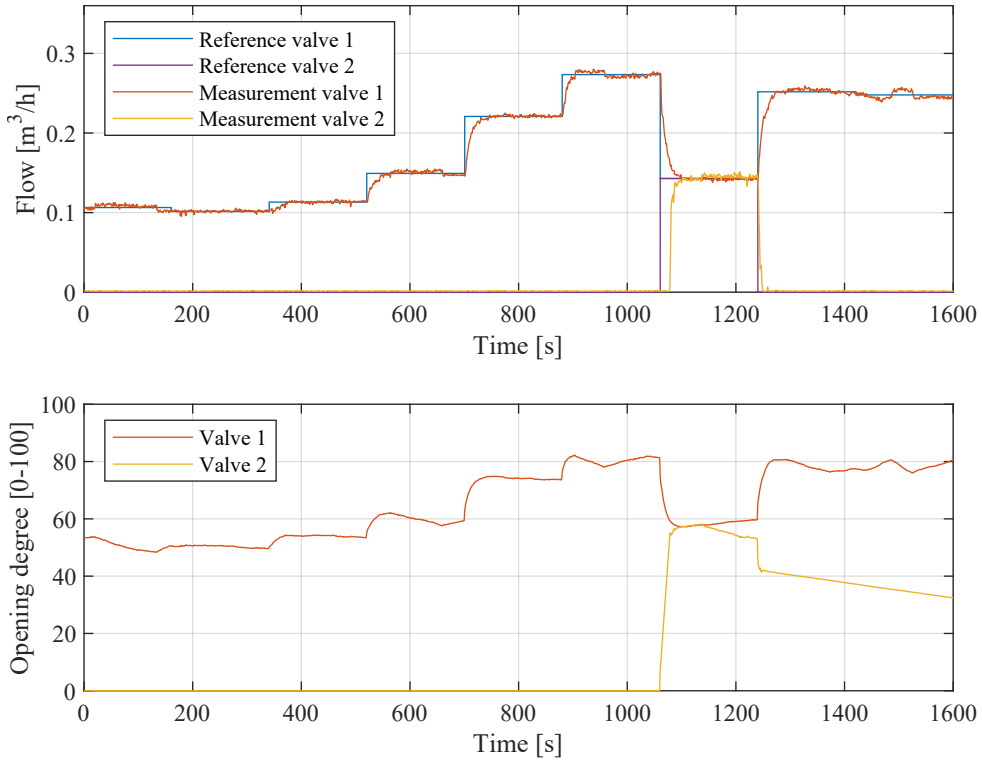


Figure 2.4. Performance of valve controllers.

2.1.3 Accelerated Time

In the SWIL, time can be accelerated to allow experiments to run faster [8]. The time acceleration is implemented by changing the electricity price and consumption at a higher rate than in the real world.

In this project, it is chosen to accelerate the time sixfold, resulting in changing electricity price every 10 min. This is a compromise between the amount of time required to perform tests and the frequency of new references from high-level control compared to the rise time of low-level controllers.

2.2 Summary

In this chapter, a laboratory setup for testing the privacy-preserving control algorithm was described. It was chosen to accelerate time within the laboratory, such that one physical hour corresponds to six laboratory hours. The setup consists of two pump stations, pipes, a water tower, and a consumption group. Low-level controllers were developed such that a model for a high-level controller can be based on flows and volume in the tower, abstracting from pump and valve dynamics. The controllers designed for the pump station flows reach the reference within approximately 30 s with significant oscillations. Furthermore, PI controllers for the consumption group were designed, which achieved rise time below 50 s and low oscillations.

Modelling 3

In this chapter, it is desired to derive a model of the setup shown in Figure 2.1. To provide a better understanding of the mathematical syntax, a new sketch of the Water Distribution Network (WDN) is shown in Figure 3.1. Low-level controllers, capable of following a reference flow, have been developed in Chapter 2. Thereby, the model in this chapter will include flow, volume in the tower, and power consumption of the pump stations while abstracting from pump, pipe, and valve dynamics.

While the WDN used in this project consists of two pump stations and a single consumption group, it is desired to derive a model that fits an arbitrary number.

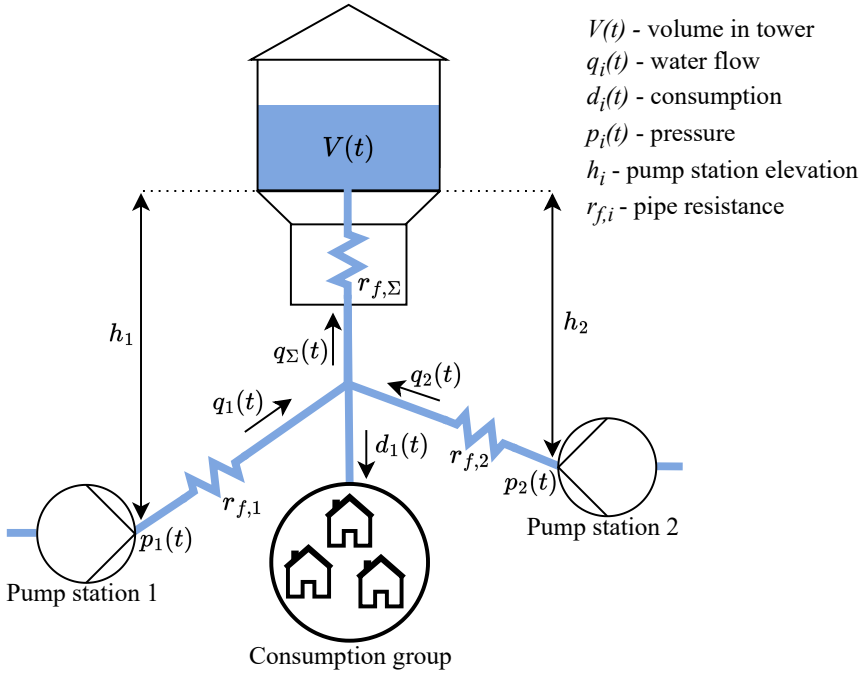


Figure 3.1. Sketch showing the mathematical syntax used to model the plant.

3.1 Volume Balance

The tower's water volume balance is modelled in (3.1) as the difference between the flow into and out of the tower.

$$\dot{V}(t) = \sum_{i=1}^{N_\mathrm{q}} q_i(t) - \sum_{i=1}^{N_\mathrm{d}} d_i(t) \tag{3.1}$$

where:

$V(t)$	Volume of water in the tower at time t	$[\mathrm{m}^3]$
N_q	Number of pump stations	$[\cdot]$
$q_i(t)$	Flow through pump station i at time t	$[\mathrm{m}^3\,\mathrm{s}^{-1}]$
N_d	Number of consumption groups	$[\cdot]$
$d_i(t)$	Flow into consumption group i at time t	$[\mathrm{m}^3\,\mathrm{s}^{-1}]$

Equation (3.2) shows the discrete-time model obtained using Forward Euler discretisation. For the discretisation to be exact, the consumption $d_i(t)$ and pump station flow $q_i(t)$, which are assumed to change slowly, must be constant within a sample time t_s.

$$V(t + t_\mathrm{s}) = V(t) + t_\mathrm{s} \sum_{i=1}^{N_\mathrm{q}} q_i(t) - t_\mathrm{s} \sum_{i=1}^{N_\mathrm{d}} d_i(t) \tag{3.2}$$

It is necessary to prevent overflow in the water tower, as well as to ensure that a minimum amount of water is available for emergencies, such as firefighting. This constraint for minimum and maximum volume, $\underline{V}$ and $\overline{V}$ respectively, can be written as:

$$\underline{V} \leq V(t) \leq \overline{V} \tag{3.3}$$

3.2 Pump Stations

The power consumption model of pump station i can be seen in (3.4). While the model is simple, it is deemed sufficient for this project.

$$P_i(t) = \frac{1}{\eta_i} q_i(t) p_i(t) \tag{3.4}$$

where:

$P_i(t)$	Power consumption of pump station i at time t	[W]
η_i	Efficiency of pump station i	$[\cdot]$
$p_i(t)$	Pressure delivered by pump station i at time t	[Pa]

The pressure delivered by the pump stations is determined as shown in (3.5), which includes the pipe resistances, pressure from the elevation of the tower, and water height in the tower. Remark that the second term in (3.5) will be negative when the flow is out of the tower. In some literature, a term taking the kinetic energy of the water inside the pipe into account is included. However, since flow dynamics are stable and significantly faster than tank dynamics, the flow will find a steady state quickly, therefore, the kinetic energy term is omitted.

$$p_i(t) = \underbrace{r_{\mathrm{f},i}|q_i(t)|q_i(t)}_{\text{Pipe resistance}} + \underbrace{r_{\mathrm{f},\Sigma}|q_\Sigma(t)|q_\Sigma(t)}_{\text{Combined pipe resistance}} + \underbrace{\rho_{\mathrm{w}} g_0 h_{\mathrm{V}}(t)}_{\text{Water height}} + \underbrace{\rho_{\mathrm{w}} g_0 h_i}_{\text{Tower elevation}} \tag{3.5a}$$

$$q_\Sigma(t) = \sum_{i=1}^{N_{\mathrm{q}}} q_i(t) - \sum_{i=1}^{N_{\mathrm{d}}} d_i(t) \tag{3.5b}$$

$$h_{\mathrm{V}}(t) = \frac{V(t)}{A_{\mathrm{t}}} \tag{3.5c}$$

where:

$r_{\mathrm{f},i}$	Pipe resistance of pipe i	[Pa s^2 m^{-6}]
ρ_{w}	Density of water	[997 kg m^{-3}]
g_0	Gravitational acceleration	[9.82 m s^{-2}]
h_i	Pipe elevation	[m]
$h_{\mathrm{V}}(t)$	Height of water inside the tower at time t	[m]
A_{t}	Water surface area in cylindrical tower	[m^2]

Combining (3.4) and (3.5) results in the pump station power consumption shown in equation (3.6).

$$P_i(t) = \frac{1}{\eta_i} q_i(t) \Big(r_{f,i}|q_i(t)|q_i(t) + r_{f,\Sigma}|q_\Sigma(t)|q_\Sigma(t) + \rho_w g_0 \left(h_{\mathrm{V}}(t) + h_i\right) \Big) \tag{3.6}$$

Real-world pump stations have a Total Yearly Extraction Limit (TYEL) $\overline{Q}_i$, which is the maximum amount of water allowed to be pumped over a year, to avoid over-absorption of the wells. The relationship between flow and TYEL will be described later.

In addition to a TYEL, real-world pump stations have a maximum pump capacity $\overline{q}_i$ and do not have negative flow. This results in the constraint:

$$0 \leq q_i(t) \leq \overline{q}_i \tag{3.7}$$

3.3 Water Consumption Model

A dataset of consumption from Bjerringbro Fælles Vandværk (Bjerringbro Water Distributor), supplying 3 700 consumers, is available. The dataset consists of data for 88 days sampled every 15 minutes.

Water consumption is known to be periodic within a day [2]. However, differences between weekends and workdays are expected. For simplicity, the used flow into the consumption group is chosen to consist of the measured water consumption. The prediction model is the mean flow into the consumption group in the specific 15-minute weekly interval for the whole dataset.

The prediction and consumption model have been scaled to the operating range of the consumer valve controller described in Section 2.1.2. The prediction and consumption model for seven days is shown in Figure 3.2. The figure shows a significant correlation between the prediction and consumption, though inaccurate, especially at peak flow into the consumption group.

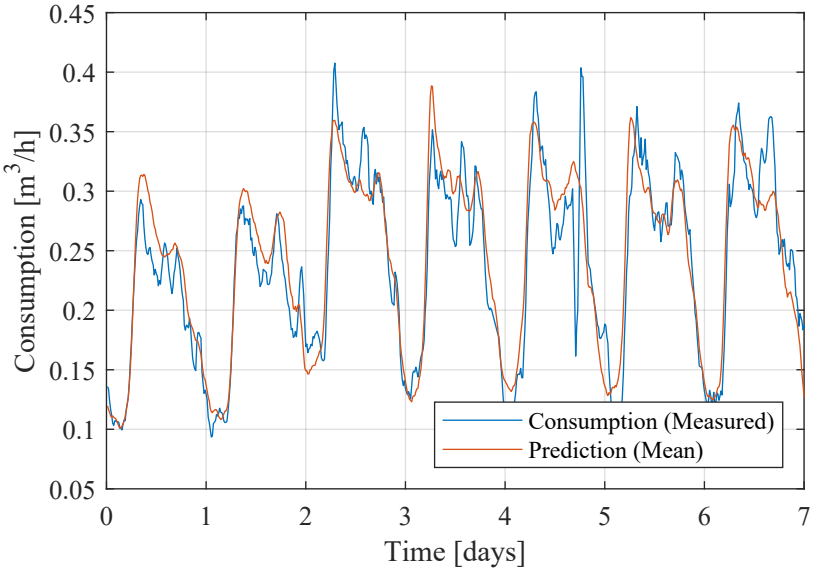


Figure 3.2. Current and predicted flow into the consumption group for the WDN in Bjerringbro, scaled to the operating range of the consumer valve controller.

3.4 Constants and Constraints

In this section, the constants and constraints to which the plant is subject are presented. Some values are based solely upon the physical constraints of the lab, while others are chosen.

The utilised constants are shown in Table 3.1. The elevations and pump station efficiency are chosen arbitrarily, given laboratory limitations. The cross-section of the water tower has been estimated based on its circumference. The pipe resistances have been estimated in Appendix F.

Parameter	Symbol	Value
Cross section of water tower	A_t	$0.28\,\mathrm{m}^2$
Elevation of tower compared to pump station	h_1	$2\,\mathrm{m}$
Elevation of tower compared to pump station	h_2	$1.5\,\mathrm{m}$
Pipe resistance	$r_{f,1}$	$0.35 \cdot 10^5$ Pa h^2 m^{-6}
Pipe resistance	$r_{f,2}$	$0.42 \cdot 10^5$ Pa h^2 m^{-6}
Pipe resistance	$r_{f,\Sigma}$	$0.29 \cdot 10^5$ Pa h^2 m^{-6}
Efficiency factor of pump station	η_1	0.9
Efficiency factor of pump station	η_2	0.8

Table 3.1. Constant values for the model.

The constraints for the plant are shown in Table 3.2. The TYEL has been chosen to be half the pump station's maximum flow. The water volume is chosen based on the volume of the tank.

Parameter	Value
Pump station flow	$0 \leq q_i(t) \leq 0.3\,\mathrm{m}^3\,\mathrm{h}^{-1}$
TYEL pump station 1	$3.6\,\mathrm{m}^3$ day^{-1}
TYEL pump station 2	$3.6\,\mathrm{m}^3$ day^{-1}
Water volume	$28\,\mathrm{L} \leq V(t) \leq 155\,\mathrm{L}$

Table 3.2. Constraints for the plant.

3.4.1 Summary

In this chapter, a simple model for a WDN was derived. The model includes the water volume balance for the tower, the power consumption of the pump stations, and models for the current and predicted flow into the consumption group. The model for the power consumption includes a common pipe resistance for all pump stations, unique pipe resistance to each pump station, water level, and pump station elevation. The predicted flow into the consumption group is based on average consumption data from Bjerringbro, whereas the current consumption corresponds to a measurement from the data set. In addition, the constants and constraints to which the plant is subject were introduced. With the models derived, it is now possible to implement Model Predictive Control (MPC).

Global Controller Design 4

This chapter describes and simulates a model predictive controller for the model derived in the previous chapter. The controller will act as a reference for a distributed privacy-preserving controller, which will be designed in the following chapters.

First, the concept of Model Predictive Control (MPC) is introduced and a cost function consisting of the plant's electricity bill is derived. The cost function, which is subject to plant constraints, is minimised to determine actuation commands. The global controller is compared with a simple ON/OFF controller, to evaluate the obtained electricity bill reduction.

4.1 Model Predictive Control

At the core of MPC is the minimisation of a cost function J_k defined over a control horizon N_c, where the optimisation variable is a vector of future actuation commands $\boldsymbol{\mathcal{U}}_k$ [11]. In this project, this vector consists of the actuation signals for all pump stations throughout the control horizon, which can be written as:

$$\boldsymbol{\mathcal{U}}_k = \begin{bmatrix} \boldsymbol{u}_k \\ \boldsymbol{u}_{k+1} \\ \vdots \\ \boldsymbol{u}_{k+N_\mathrm{c}-1} \end{bmatrix} \in \mathbb{R}^{N_\mathrm{c}N_\mathrm{q}} \qquad \boldsymbol{u}_k = \begin{bmatrix} q_1(t_0 + kt_\mathrm{s}) \\ q_2(t_0 + kt_\mathrm{s}) \\ \vdots \\ q_{N_\mathrm{q}}(t_0 + kt_\mathrm{s}) \end{bmatrix} \in \mathbb{R}^{N_\mathrm{q}} \tag{4.1}$$

where:

N_q	Number of pump stations	$[\cdot]$
$q_i(t)$	Flow through pump station i at time t	$[\mathrm{m}^3\,\mathrm{s}^{-1}]$
t_s	Sampling time	[s]

Once the optimal actuation commands $\boldsymbol{\mathcal{U}}_k$ are found by minimising the cost function J_k, only the actuation $\boldsymbol{u}_k$ corresponding to the current time is applied [11]. The control horizon is then shifted one time step forward, and the optimisation is repeated at the next time step [11].

Water consumption is known to be 24 h periodic as described in Section 3.3; likewise, electricity prices are roughly 24 h periodic as shown in Figure 1.2. For these reasons, a control horizon of 24 h is chosen. It is chosen not to address the issue that electricity prices are not always known 24 h ahead [12].

The hourly change in electricity prices suggests a sampling time of 1 h. A faster sampling time would be preferable, as it would lower the risk of breaking constraints due to inaccurate prediction of flow into the consumption group. Nevertheless, realising a higher sampling time would prove challenging due to the sixfold time acceleration, described in Section 2.1.3, and long rise time of the low-level controllers described in Section 2.1.2. Therefore, a sampling time of 1 h is chosen, corresponding to 10 minutes due to the time acceleration.

4.1.1 Cost Function

The objective of the cost function is to minimise the electricity bill as described in Chapter 1. The electricity bill for the control horizon, based on the model derived in Chapter 3, is shown in (4.2).

$$J_{\mathrm{p}} = \sum_{i=1}^{N_{\mathrm{q}}} \left(\sum_{\alpha=k}^{k+N_{\mathrm{c}}-1} P_i(\alpha) t_{\mathrm{s}} J_{\mathrm{e}}(\alpha)) \right) \tag{4.2}$$

$$P_i(\alpha) = \frac{1}{\eta_i} \Big(\underbrace{r_{\mathrm{f},i} |q_i(\alpha)| q_i^2(\alpha)}_{\text{Pipe resistance}} + \underbrace{r_{\mathrm{f},\Sigma} |q_\Sigma(\alpha)| q_\Sigma(\alpha) q_i(\alpha)}_{\text{Combined pipe resistance}} + \tag{4.3a}$$

$$+ \underbrace{\rho_w g_0 h_{\mathrm{V}}(\alpha) q_i(\alpha)}_{\text{Water height}} + \underbrace{\rho_{\mathrm{w}} g_0 h_i q_i(\alpha)}_{\text{Elevation}} \Big) \tag{4.3b}$$

$$q_\Sigma(\alpha) = \sum_{i=1}^{N_{\mathrm{q}}} q_i(\alpha) - \sum_{i=1}^{N_{\mathrm{d}}} d_i(\alpha) \tag{4.3c}$$

$$h_{\mathrm{V}}(\alpha+1) = \frac{V(\alpha+1)}{A_{\mathrm{t}}} \tag{4.3d}$$

$$V(\alpha+1) = t_{\mathrm{s}} q_\Sigma(\alpha) + V(\alpha) \tag{4.3e}$$

where:

J_p	Electricity bill for all pump stations	[Euro]
$J_\mathrm{e}(\alpha)$	Electricity price at time α	[Euro/Ws]
$P_i(\alpha)$	Power consumption of pump station i at time α	[W]
η_i	Efficiency of pump station i	$[\cdot]$
$r_{\mathrm{f},i}$	Pipe resistance	[Pa s^2 m^{-6}]
ρ_w	Density of water	$[997\,\mathrm{kg\,m^{-3}}]$
g_0	Gravitational acceleration	$[9.82\,\mathrm{m\,s^{-2}}]$
$h_\mathrm{V}(\alpha)$	Height of water inside the tower at time α	[m]
h_i	Pump station elevation	[m]
$d(\alpha)$	Flow into the consumption group at time α	$[\mathrm{m^3\,s^{-1}}]$
t_s	Sampling time	[s]
$V(\alpha)$	Volume of water in the tower at time α	$[\mathrm{m^3}]$
A_t	Water surface area in the cylindrical tower	$[\mathrm{m^2}]$

Tjell [1] suggests adding a term to the cost function which penalises the difference in water volume in the tower at the beginning and end of the control horizon. Without such a term, the volume of water in the tower will always be predicted as the minimum volume at the end of the control horizon to minimise the cost function. As the flow into the consumption group and electricity prices are roughly 24 h periodic, the same volume of water in the tower at the beginning and end of the control horizon will be a better prediction. The term suggested by [1] can be written as shown in (4.4), where κ is a tuning variable, and $V(k)$ is the volume of water in the tower at time k.

$$J_\mathrm{V} = \kappa \left(V(k) - V(k + N_\mathrm{c} - 1)\right)^2 \tag{4.4}$$

Should the electricity price become significantly larger, the contribution of J_V becomes insignificant. This issue is solved by scaling the vector of electricity prices over the control horizon, such that its 2-norm is always one.

This leads to the final cost function:

$$J_\mathrm{k} = J_\mathrm{p} + J_\mathrm{V} \tag{4.5}$$

Remark that some terms in J_p, (4.2), are not convex; therefore, J_k might not be convex. The first term, pipe resistance, is convex for positive flows. However, the second term, combined pipe resistance, is not convex since two different flows are multiplied. In the third term, past flows are multiplied by current flows, thereby resulting in the term being non-convex. In the last term, the elevation of the tower is linear.

A function is convex if and only if its Hessian is positive semi-definite [13, p. 71], given that the constraints result in a convex domain. Therefore, the convexity of the cost function J_k is tested by checking the Hessian for positive definiteness.

The Hessian is a function of the pump station flows; therefore, the test must be performed for multiple flows for each pump station at every hour within the control horizon. This requires a short control horizon to lower the number of combinations. The test is performed for $N_\mathrm{c} = 2$, which is the shortest horizon where the water height in the tower results in a non-convex term.

The test returns Hessians, which are not positive semi-definite for several flow combinations, meaning that the cost function is non-convex. It is assumed that the same results would be achieved for $N_\mathrm{c} = 24$ since more flows will be multiplied with each other in the water height term. Therefore, an optimiser must be expected to find a minimum, which is not necessarily the global minimum. Though not ideal, a local minimum is likely to reduce costs compared to non-optimising control methods.

The code for the test can be found at link 1 in Appendix A.

4.1.2 Conditioning of the Cost Function

The cost function defined in (4.2) consists of large numbers such as $r_{\mathrm{f},i} \approx 10^4$ $\mathrm{Pa}/(\mathrm{m}^3/\mathrm{h})^2$ and $\rho_\mathrm{w} g_0 \approx 10^4\,\mathrm{Pa\,m^{-1}}$, as well as very small numbers, i.e. flows. This has shown to pose problems for optimisers, such as Matlab fmincon. It is chosen to take a practical approach to the conditioning of the problem, namely ensuring that all inputs to non-linear functions and all terms in the cost function are close to one. This is achieved by choosing the unit of flow to cubic metre per hour, multiplying all terms in (4.3a)-(4.3b) with 10^{-4}, and omitting the multiplication with the sample time t_s. Since the electricity price is scaled such that its 2-norm becomes one, it should not result in ill-conditioning. This practical approach has significantly reduced the number of iterations required for the optimiser to solve the optimisation problem.

4.1.3 Constraints

The minimisation of the cost function is subject to constraints as described in Section 3.4. The minimum and maximum flow for each pump station can be written as shown in (4.6), where $\overline{q}_i$ is the maximum flow.

$$0 \leq q_i(t) \leq \overline{q}_i \tag{4.6}$$

The constraint ensuring that the water height in the tower is above a minimum and below a maximum can be written as shown in (4.7), where $\underline{V}$ and $\overline{V}$ are the minimum and maximum water volume allowed in the tower.

$$\underline{V} \leq V(t) \leq \overline{V} \tag{4.7}$$

The Total Yearly Extraction Limit (TYEL) constraint is enforced by setting an upper limit for the maximum pumped volume within the control horizon, as shown in (4.8), where $\overline{Q}_i$ is the TYEL over the control horizon for pump station i. In theory, this implementation does not guarantee the TYEL to be obeyed due to the control horizon being shifted at each time step. In reality, the TYEL is expected to be obeyed. However, investigating this further is considered outside the scope of this project.

$$\sum_{\alpha=k}^{k+N_c-1} q_i(\alpha) t_s \leq \overline{Q}_i \tag{4.8}$$

Before implementing the control algorithms, it is desired to rewrite the cost function and constraints such they are on matrix form as a function of $\boldsymbol{\mathcal{U}}_k$. This is done in Appendix G.

4.2 ON/OFF Controller

An ON/OFF controller is simulated to evaluate if the global model predictive controller reduces the electricity bill of the Water Distribution Network (WDN). The ON/OFF controller is based on the study case in Val Ledesma, Wisniewski, and Kallesøe [8, p. 13-14]. In the study case, the pump station starts pumping when the water volume falls below a specified lower limit and stops pumping when the water volume surpasses an upper limit. In this project, the flow for each pump station is set to $0.2\,\mathrm{m^3\,h^{-1}}$, which is half of the maximum consumption as shown in Figure 3.2. The sample time is chosen to be 5 minutes, corresponding to 30 s in accelerated time. The upper and lower limit is chosen to be the upper and lower water volume limits for the tower presented in Section 3.4. This implementation results in slightly breaking these constraints due to limited sampling time. If desired, the pumps could be turned on and off at slightly changed water volumes to avoid breaking the constraints, though not prioritised in this project.

After 240 simulated hours, the electricity bill is estimated to 1.09 Eurocent, using the electricity price vector and the model for pump station power consumption in (3.6). While this value is very low, it is not unexpected due to low flows and the accelerated time described in Section 2.1.3.

4.3 Global Controller

The controller is simulated in Simulink using the fmincon solver-based approach with sequential quadratic programming (SQP) as the solver. Remark that the solver might not return the global minimum since the cost function is not convex as described in Section 4.1.1. The implementation of the global controller can be found at link 2 in Appendix A.

The tuning variable κ in (4.4) is desired to be as low as possible. Simulations have shown that $\kappa = 900$ results in a mean difference in the volume of 2 %, when solving optimisation problems for 100 hours, which is considered acceptable.

Simulations of the global controller and ON/OFF controller using the exact plant model derived in Chapter 3, have resulted in the operation shown in Figure 4.1, where the prediction is plotted at hour80. The figure shows that the global controller breaks the water tower's volume constraint due to the difference between predicted and actual flow into the consumption group. This could potentially be solved by modelling the uncertainty and account for it using stochastic MPC. Alternatively, the controller constraints for the maximum and minimum water volume could be adjusted, such that the actual constraints are not broken. Lastly, it could be investigated whether a more accurate model for the predicted consumption could be derived. None of these suggestions are implemented, since solving the issue is not considered within the scope of this project.

Furthermore, the figure shows high pump station flows at low electricity prices for the global controller, which is the desired performance. The electricity bill of the global controller is estimated to 0.89 Eurocent, reducing the electricity bill by 18.3 %, which means a significant cost reduction can be obtained by introducing MPC. Remark that this is achieved even though the controller is not guaranteed to find the global minima of the cost function.

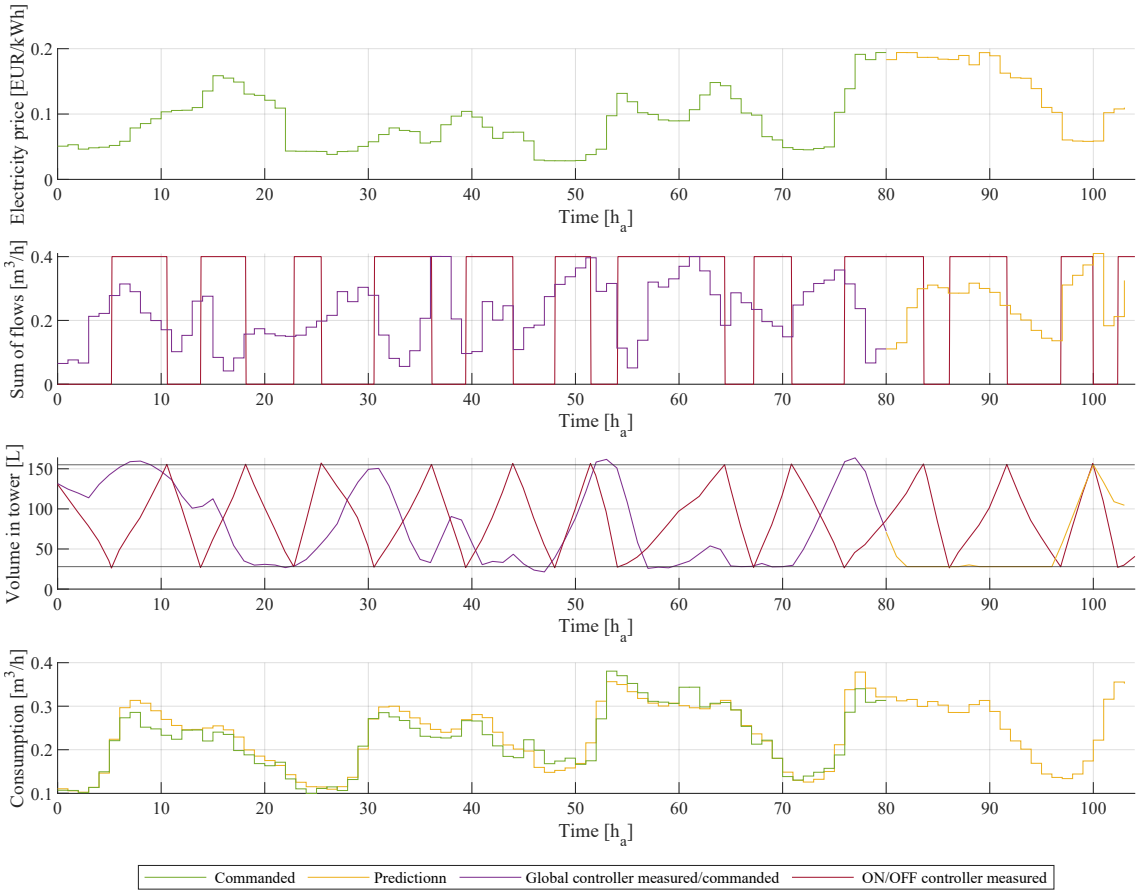


Figure 4.1. Results from the global controller in regard to summed pump station flow, consumption, and water volume, compared to the ON/OFF controller, where h_a refers to one accelerated hour.

4.4 Summary

In this chapter, cost function and constraints for a global model predictive controller were derived based on the model in Chapter 3. Simulations show that the global controller reduced the electricity bill of the Water Distribution Network by 18.3 % compared to an ON/OFF controller. Thus, significant cost reduction can be obtained by implementing the global controller.

The global controller holds all of the cost functions and constraints, which means all this will be leaked in case of a passive attack. In the following chapters, a distributed privacy-preserving controller is developed, which does not gather all cost functions and constraints at a single entity, thereby reducing the leaks in case of passive attacks. The success criteria for the privacy-preserving controller will be to have the same operation as the global controller, while being privacy-preserving.

Distributed Controller 5

Tjell [1] suggests keeping cost functions and constraints private by solving the Model Predictive Control (MPC) problem distributed using consensus Alternating Direction Method of Multipliers (ADMM). Meanwhile, communication between stakeholders will be encrypted using Secure Multi-Party Computation (SMPC), as described in Chapter 6. This chapter describes and simulates consensus ADMM. At the end of the chapter, a more privacy-preserving scheme is introduced as an inspiration, although it cannot be applied to the cost function in this project. An introduction of ADMM can be found in [14, Sec. 1-3].

5.1 Consensus ADMM

The consensus ADMM algorithm can be used to solve consensus problems on the form given in (5.1). The cost function is a sum of the N_s sub-cost functions f_i. The constraint defines that all vectors $\mathbf{x}_i$ must be equal to $\mathbf{z}$, i.e., there must be consensus between all $\mathbf{x}_i$.

$$\min_{\mathbf{x}_i,\mathbf{z}} \quad \sum_{i=1}^{N_\mathrm{s}} f_i(\mathbf{x}_i) \tag{5.1a}$$

$$\text{s.t.} \quad \mathbf{x}_i - \mathbf{z} = 0 \tag{5.1b}$$

The consensus ADMM algorithm given in (5.2) [14, pp. 48-49] is a special case of ADMM. In (5.2a), the argument $\mathbf{x}_i$ is determined, minimising the augmented Lagrangian $\mathcal{L}_{\rho,i}$ for each sub-cost function. The middle term in (5.2a) penalises not having consensus by a factor defined by the Lagrange multiplier $\boldsymbol{\lambda}_i$. The last term, which leads to convergence under milder conditions for f_i, is another penalty for not having consensus, weighted by the penalty parameter ρ. In (5.2b) and (5.2c), the consensus variable is calculated, and the Lagrange multipliers are updated, respectively. The minimisation in (5.2b) can be derived by differentiating $\mathcal{L}_{\rho,i}$ with respect to $\mathbf{z}$ and setting the result equal to zero, and solving for $\mathbf{z}$. The optimisation problem can be solved by making initial guesses for vectors $\mathbf{z}$ and $\boldsymbol{\lambda}_i$ before iterating through (5.2).

$$\mathbf{x}_i^{k+1} = \arg\min_{\mathbf{x}_i} \mathcal{L}_{\rho,i}(\mathbf{x}_i, \boldsymbol{\lambda}_i^k, \mathbf{z}^k) = \arg\min_{\mathbf{x}_i} \left(f_i(\mathbf{x}_i) + \boldsymbol{\lambda}_i^{k\top}(\mathbf{x}_i - \mathbf{z}^k) + \frac{\rho}{2}||\mathbf{x}_i - \mathbf{z}^k||_2^2 \right) \tag{5.2a}$$

$$\mathbf{z}^{k+1} = \arg\min_{\mathbf{z}} \left(\sum_{i=1}^{N_\mathrm{s}} \mathcal{L}_{\rho,i}(\mathbf{x}_i^{k+1}, \boldsymbol{\lambda}_i^k, \mathbf{z}) \right) = \frac{1}{N_\mathrm{s}} \sum_{i=1}^{N_\mathrm{s}} \left(\mathbf{x}_i^{k+1} + \frac{1}{\rho}\boldsymbol{\lambda}_i^k \right) \tag{5.2b}$$

$$\boldsymbol{\lambda}_i^{k+1} = \boldsymbol{\lambda}_i^k + \rho \left(\mathbf{x}_i^{k+1} - \mathbf{z}^{k+1} \right) \tag{5.2c}$$

Boyd, Parikh, Chu, *et al.* writes that ADMM converges to modest accuracy within a few tens of iterations [14, p. 17], and that convergence in practice can be improved by proper choice of the penalty parameter ρ. However, convergence of consensus ADMM for $N_\mathrm{s} > 2$ is a debated theme even for convex functions [15]–[17].

Boyd, Parikh, Chu, *et al.* [14] does not comment on convergence for $N_\mathrm{s} > 2$. Chen, He, Ye, *et al.* [15] reports that the algorithm does not necessarily converge and sets up a necessary condition for convergence but comments that the condition is only of theoretical interest, as it is not easily satisfied in known applications. Tjell and Wisniewski [16] reports, based on He, Hou, and Yuan [17], that the just presented consensus ADMM algorithm is likely to diverge unless under-relaxation is applied. Under-relaxation reduces the rate of change for the consensus vector $\mathbf{z}$ and the Lagrange multipliers $\boldsymbol{\lambda}_i$. The under-relaxation in [16], modified to consensus ADMM, is shown in (5.3).

$$\mathbf{x}_i^{k+1} = \arg\min_{\mathbf{x}_i} \left(f_i(\mathbf{x}_i) + \boldsymbol{\lambda}_i^{k\top}(\mathbf{x}_i - \mathbf{z}^k) + \frac{\rho}{2}||\mathbf{x}_i - \mathbf{z}^k||_2^2 \right) \tag{5.3a}$$

$$\tilde{\boldsymbol{z}}^{k+1} = \frac{1}{N_\mathrm{s}} \sum_{i=1}^{N_\mathrm{s}} \left(\mathbf{x}_i^{k+1} + \frac{1}{\rho}\boldsymbol{\lambda}_i^k \right) \tag{5.3b}$$

$$\mathbf{z}^{k+1} = \mathbf{z}^k - \frac{1}{N_\mathrm{s}+1} \left(\mathbf{z}^k - \tilde{\boldsymbol{z}}^{k+1} \right) \tag{5.3c}$$

$$\tilde{\boldsymbol{\lambda}}_i^{k+1} = \boldsymbol{\lambda}_i^k + \rho \left(\mathbf{x}_i^{k+1} - \mathbf{z}^{k+1} \right) \tag{5.3d}$$

$$\boldsymbol{\lambda}_i^{k+1} = \boldsymbol{\lambda}_i^k - \frac{1}{N_\mathrm{s}+1} \left(\boldsymbol{\lambda}_i^k - \tilde{\boldsymbol{\lambda}}_i^{k+1} \right) \tag{5.3e}$$

5.1.1 Consensus ADMM in WDN

The cost function defined in Section 4.1.1 is already a sum of sub-cost functions, one for each pump station and one for the difference in the volume of water in the tower at the beginning and end of the control horizon.

Three stakeholders are introduced: the two pump stations and the water tower. These stakeholders each hold a part of the cost function and the constraints specified in Table 5.1. The stakeholders solve the optimisation problem, each having their own $\mathbf{x}_i = \boldsymbol{\mathcal{U}}_k$, being all predicted flows for the pump stations within the control horizon. All $\mathbf{x}_i$ should be equal when the optimisation has been performed, i.e. there is consensus. After solving the optimisation problem, the pump stations apply the flow defined in their own $\mathbf{x}_i$.

In this project, it is decided that each stakeholder applies their constraints directly on (5.3a). Boyd, Parikh, Chu, *et al.* suggests encoding constraints by setting the cost function $f_i(\mathbf{x}_i) = +\infty$ if constraints are violated [14, pp. 48-49], which should provide the same result as the applied method, though is more complex to apply.

The constraints and cost functions have been assigned to the stakeholders, based on their natural belonging, as shown in Table 5.1 and Figure 5.1. One could argue that constraint a: positive pump station flow, does not belong to the tower, but the constraint is trivial and, for this reason, does not make sense to keep it private. The part of the cost function concerning having the same volume of water in the tower at the beginning and end of the

control horizon J_V includes no system information. For this reason, it is chosen to place a third of the cost at every stakeholder.

Stakeholder	Constraints	Cost function
Water tower	a: $0 \leq q_1(t), q_2(t)$ b: $\underline{V} \leq V(t) \leq \overline{V}$	c: $\frac{J_V}{3}$
Pump station 1	d: $0 \leq q_1(t) \leq \overline{q}_1$ e: $\sum_{\alpha=k}^{k+N_c-1} q_1(\alpha) t_s \leq \overline{Q}_1$ f: $0 \leq q_2(t)$	g: $\sum_{\alpha=k}^{k+N_c-1} P_1(\alpha) t_s J_e(\alpha)) + \frac{J_V}{3}$
Pump station 2	d: $0 \leq q_2(t) \leq \overline{q}_2$ e: $\sum_{\alpha=k}^{k+N_c-1} q_2(\alpha) t_s \leq \overline{Q}_2$ f: $0 \leq q_1(t)$	g: $\sum_{\alpha=k}^{k+N_c-1} P_2(\alpha) t_s J_e(\alpha)) + \frac{J_V}{3}$

Table 5.1. Constraints and cost functions for each stakeholder. a: Non-negative pump station flow, b: Tower volume constraint, c: $J_V/3$ see Section 4.1, d: minimum and maximum flow through pump station, e: Total Yearly Extraction Limit (TYEL), f: Positive pump flow, g: Electricity bill plus $J_V/3$.

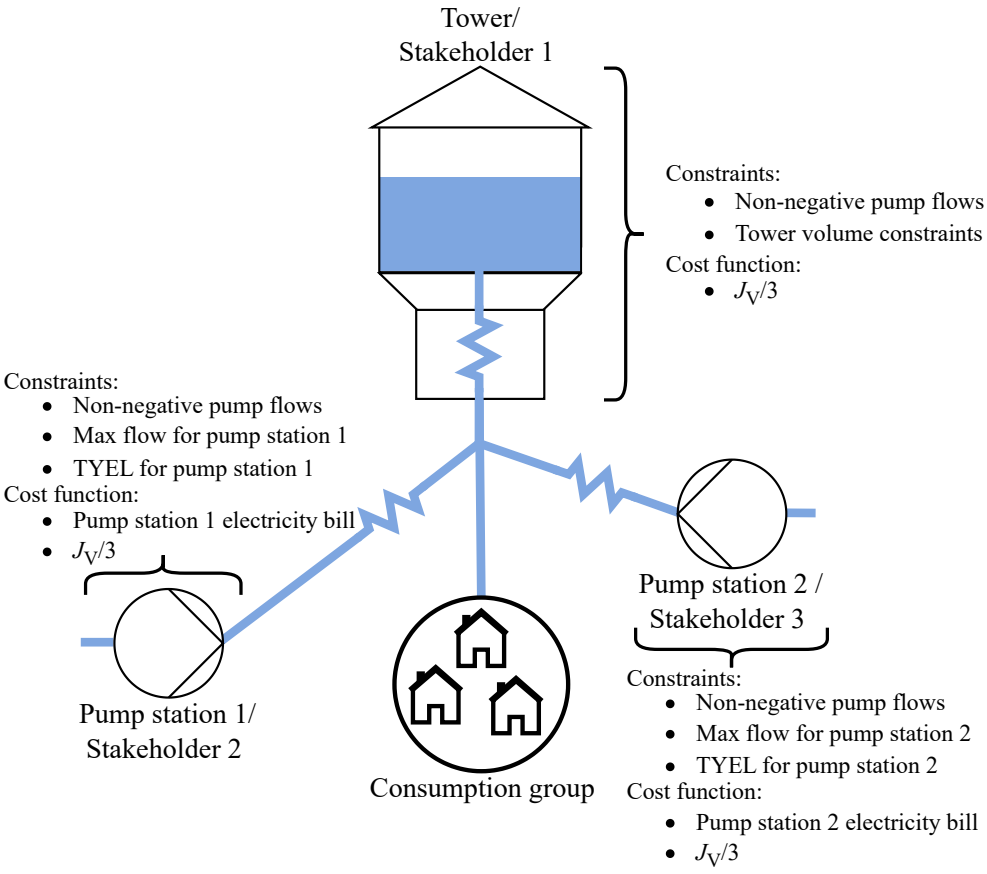


Figure 5.1. Illustration of distribution. For details, see Table 5.1.

When solving the optimisation problem, (5.2) can be calculated individually by each stakeholder, with the exception of (5.2b). Thereby, the stakeholder's cost function and constraints are private if (5.2b) can be calculated without disclosing each stakeholder's contribution. This is possible using Secure Multi-Party Computation, which will be described in Chapter 6.

The rest of this section will simulate consensus ADMM to determine whether under-relaxation should be applied, identify values for the penalty parameter ρ, and establish an appropriate stopping criterion.

5.1.2 Under-Relaxation

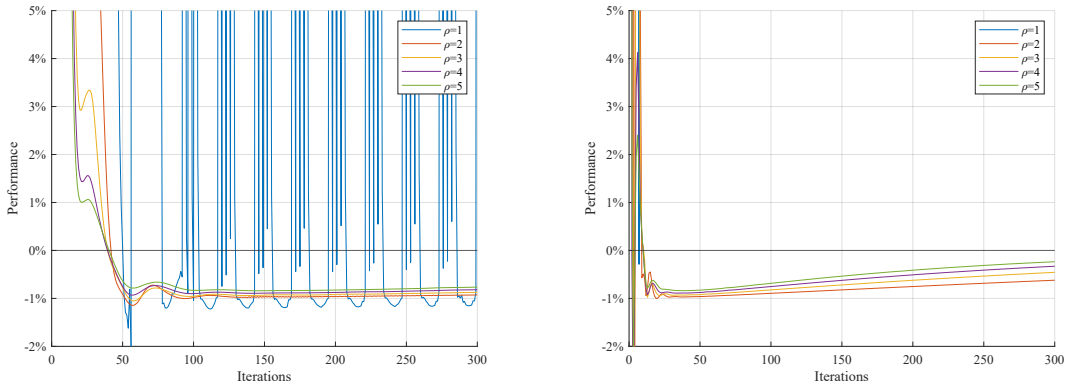
It has previously been described that the convergence of consensus ADMM is a debated theme and that Tjell and Wisniewski [16] claims that under-relaxation may be required for consensus ADMM to result in convergence. For this reason, it is chosen to test the influence of under-relaxation.

The MPC problem for a single hour has been solved using consensus ADMM with and without under-relaxation for different values of the penalty parameter ρ. The performance of the consensus ADMM algorithm is evaluated by comparing the cost of the solution $J_k(\mathbf{x}_i^{'})$ with the cost of the global controller $J_k(\mathbf{x}^*)$. Since the cost function is not convex, as described in Section 4.1.1, different solutions might be found, where neither is guaranteed to be the global minimum. If two different minima are to be found, it is expected that the cost of the solution will be constant but different after a finite number of iterations.

For consensus ADMM, performance is evaluated using the predicted flows belonging to each pump station. These predicted flows will be denoted as $\mathbf{x}^{'}$ from this point onward. In the case of full consensus, $\mathbf{x}^{'}$ is equal to all $\mathbf{x}_i$. The `performance` is evaluated as:

$$\texttt{Performance} = \frac{J(\mathbf{x}^{'}) - J(\mathbf{x}^*)}{J(\mathbf{x}^*)} \tag{5.4}$$

The `performance` with and without under-relaxation is shown in Figure 5.2. Remark that $\rho = 1$ does not result in convergence in either case. From the figure, it is observed that under-relaxation results in slower convergence towards 0 %. As a result, under-relaxation is not utilised.



(a) With under-relaxation. ***(b)*** Without under-relaxation.

Figure 5.2. `Performance` of ADMM with and without under-relaxation for hour 1.

5.1.3 Choice of Penalty Parameter

During tests, it was discovered that a penalty parameter resulting in convergence for one hour does not result in convergence for all hours. For instance, $\rho = 1$ results in convergence at hour 11 but not at hour one.

Ghadimi, Teixeira, Shames, *et al.* [18] states that while the algorithm, in theory, converges for any value of the penalty parameter ρ under rather mild conditions, the penalty parameter has a direct impact on the speed of convergence. Given that the cost function in this problem is not convex, as described in Section 4.1.1, increased sensitivity to the choice of penalty parameter is expected.

Boyd, Parikh, Chu, *et al.* [14, Sec. 3.4.1] suggests varying the penalty parameter, potentially using a different ρ at each iteration to improve the convergence. Additionally, this reduces the impact of choosing a less-than-ideal initial value for the penalty parameter. However, Boyd, Parikh, Chu, *et al.* does not provide any proof that a varying penalty parameter results in convergence, stating that proving convergence for a varying ρ would prove difficult. Instead, Boyd, Parikh, Chu, *et al.* assumes that convergence is the same as for a constant penalty parameter, given that the varying penalty parameter becomes constant after a finite number of iterations.

The algorithm for the varying penalty parameter presented by Boyd, Parikh, Chu, *et al.* [14, Sec. 3.4.1] is written in (5.5), where r^k and s^k are residuals for the primal and dual feasibility conditions respectively [14, Sec. 3.3]. The primal residual r^k becomes zero when $\mathbf{x}_i = \mathbf{z}$ is satisfied, whereas the dual residual s^k becomes zero when $\mathbf{x}_i$ becomes constant, indicating a minimum.

The idea behind the algorithm is to keep the residuals within a factor μ of each other, by increasing or decreasing the penalty parameter with a factor τ^{incr} or τ^{decr}. Boyd, Parikh, Chu, *et al.* [14, Sec. 3.4.1] suggests $\mu = 10$ and $\tau^{\text{incr}} = \tau^{\text{decr}} = 2$.

(5.5a) and (5.5d) can be calculated by each stakeholder. Whereas (5.5b) and (5.5c) can be calculated privacy-preserving using SMPC. Thereby, it is required to calculate two additional sums with SMPC. Using this method, $||r^k||_2^2$ and $||s^k||_2^2$ will be known by all stakeholders, however, this is considered insignificant.

$$\rho^{k+1} = \begin{cases} \tau^{\text{incr}}\rho^k & \text{if } ||r^k||_2 > \mu||s^k||_2 \\ \rho^k/\tau^{\text{decr}} & \text{if } ||s^k||_2 > \mu||r^k||_2 \\ \rho^k & \text{Otherwise,} \end{cases} \tag{5.5a}$$

$$\bar{\mathbf{x}}^k = \frac{1}{N_\text{s}} \sum_{i=1}^{N_\text{s}} \mathbf{x}_i^k \tag{5.5b}$$

$$||r^k||_2^2 = \sum_{i=1}^{N_\text{s}} ||\mathbf{x}_i^k - \bar{\mathbf{x}}^k||_2^2 \tag{5.5c}$$

$$||s^k||_2^2 = N_\text{s}\rho^2||\bar{\mathbf{x}}^k - \bar{\mathbf{x}}^{k-1}||_2^2 \tag{5.5d}$$

Since algorithm (5.5) triples the number of summations performed using SMPC, it also triples the need for communication between the stakeholders. Furthermore, a constant penalty parameter after a finite number of iterations might be necessary to guarantee convergence. For these reasons, it is chosen only to apply (5.5) for the first ten iterations.

The consensus ADMM algorithm using varying penalty parameters has been tested for the first 1 000 combinations of electricity price and predicted demand, using $\mu = 5$, $\tau^{\text{incr}} = \tau^{\text{decr}} = 1.5$, where the `performance` is shown in Figure 5.3. It was chosen to lower τ, as an increase or decrease by a factor of two was deemed too aggressive. In addition, μ was lowered, as $\mu = 10$ did not result in convergence for the first hour. In the simulation, the value of the penalty parameter is initialised as its value for the previous hour. Likewise, $\mathbf{x}_i$ and $\mathbf{z}$ are initialised as time shifted versions of the solution from the previous hour. The last entries in $\mathbf{x}_i$ and $\mathbf{z}$ are set equal to the second last entries. For the first hour, $\mathbf{x}_i$ and $\mathbf{z}$ are initialised as zero and $\rho = 1$.

In Figure 5.3, it is observed that the `performance` in a majority of the cases lies within $\pm 2\,\%$ after 50 iterations. A few outliers are present, which, based on the graph, are observed to converge very slowly. After ten iterations, the penalty parameter is 1.5 for all hours. This could indicate that a constant penalty parameter of 1.5 is sufficient; however, the algorithm is kept to ensure convergence with all electricity prices and predicted demands.

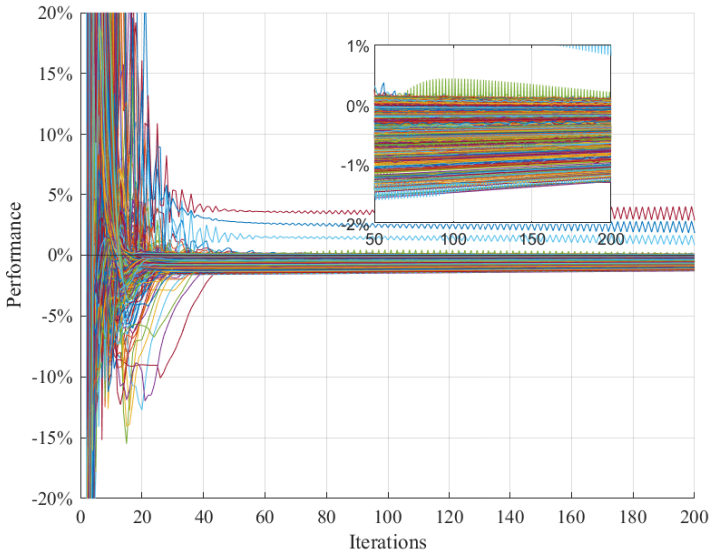


Figure 5.3. `Performance` as described in (5.4), for the first 1 000 hours.

5.1.4 Consensus

To evaluate whether or not the stakeholders have reached consensus, the primal residuals for Figure 5.3 are plotted in Figure 5.4. The values of the primal residual, considering the maximum pump station flow of $0.3\,\mathrm{m}^3\,\mathrm{h}^{-1}$, indicate that consensus is not present. To verify this, the prediction of the volume in the tower is plotted for the three stakeholders in Figure 5.5. Here, it is observed that the tower's lower volume constraint is broken by approximately 7 L by the pump stations, whereas the tower fulfils the constraints. Therefore, a lack of consensus is present. The negative `performance` in Figure 5.3 is assumed to be a result of a lack of consensus.

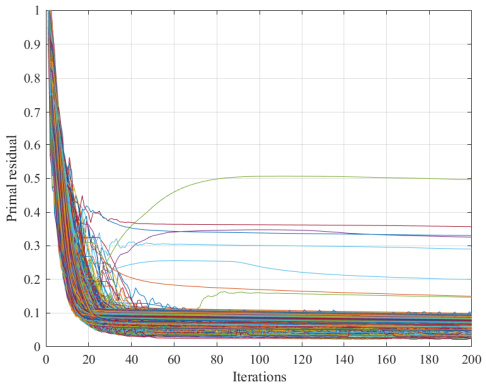


Figure 5.4. Primal residuals for the first 1 000 in Figure 5.3.

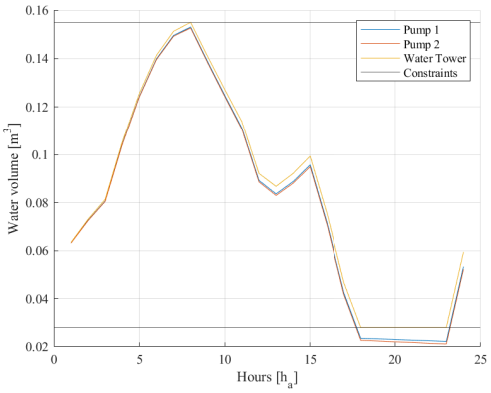


Figure 5.5. Predicted water volume for each stakeholder for hour one.

A number of different solutions to obtain consensus exist. One theoretical solution would be to lower μ in (5.5), thereby forcing the primal and dual residuals closer together, which should ensure consensus. However, results have been sub-optimal regarding convergence rate, consensus, and the number of iterations with varying penalty parameters required. Another computationally heavy solution would be to significantly increase the number of iterations. After 3 000 iterations, the tower's volume constraint is broken by 0.04 L; thereby fulfilling the constraints to a satisfying degree.

This approach results in consensus due to the Lagrange multipliers in (5.2), restated in (5.6). From (5.6c), it is observed that the value of $\boldsymbol{\lambda}_i$ will increase or decrease when consensus is not present. Since $\boldsymbol{\lambda}_i$, in (5.6a), is multiplied with the difference from consensus, it will result in consensus given enough iterations.

$$\mathbf{x}_i^{k+1} = \arg\min_{\mathbf{x}_i} \left(f_i(\mathbf{x}_i) + \boldsymbol{\lambda}_i^{k\top}(\mathbf{x}_i - \mathbf{z}^k) + \frac{\rho}{2}||\mathbf{x}_i - \mathbf{z}^k||_2^2 \right) \tag{5.6a}$$

$$\mathbf{z}^{k+1} = \frac{1}{N_\mathrm{s}} \sum_{i=1}^{N_\mathrm{s}} \left(\mathbf{x}_i^{k+1} + \frac{1}{\rho}\boldsymbol{\lambda}_i^k \right) \tag{5.6b}$$

$$\boldsymbol{\lambda}_i^{k+1} = \boldsymbol{\lambda}_i^k + \rho \left(\mathbf{x}_i^{k+1} - \mathbf{z}^{k+1} \right) \tag{5.6c}$$

Another method to obtain consensus can be found in (5.6a), where greater values of ρ result in greater punishment for not having consensus. Greater values of ρ also amplify the rate of change for $\boldsymbol{\lambda}_i$, as shown in (5.6c). Therefore, it is suggested to increase ρ after a predetermined number of iterations. From examining Figure 5.3, it is observed that there are minimal changes in `performance` beyond the 30th iteration. Therefore, ρ is increased by a factor of 500 after 30 iterations. The factor was found through hand-tuning until a satisfying convergence speed of the primal and dual residuals was achieved.

Figure 5.6 and Figure 5.7 show the primal residual and `performance` respectively with increased penalty parameter at iteration 30. Figure 5.6 shows that the primal residual drops after increasing the penalty parameter, while Figure 5.7 shows that the `performance` becomes positive and then converges to zero.

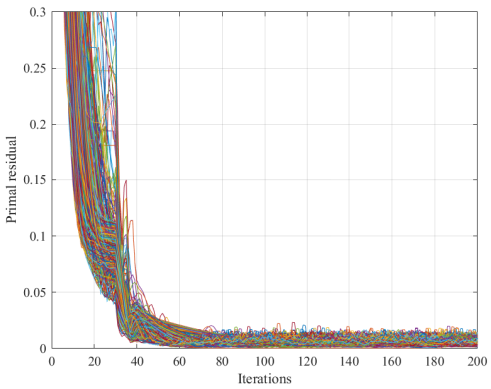


Figure 5.6. Primal residual.

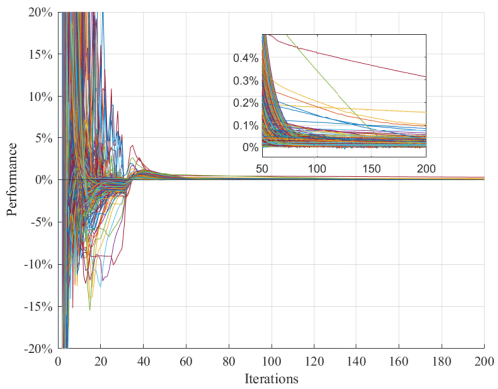


Figure 5.7. `Performance` for 1 000 hours.

Figure 5.8 shows the maximum disagreement in the predicted water volume in the tower among the three stakeholders after 200 iterations. From the figure, it is observed that this disagreement is less than 0.04 L. Since this is expected to be within the accuracy of the low-level controllers described in Section 2.1.2, consensus is considered achieved. Therefore, increasing the penalty parameter has the desired effect.

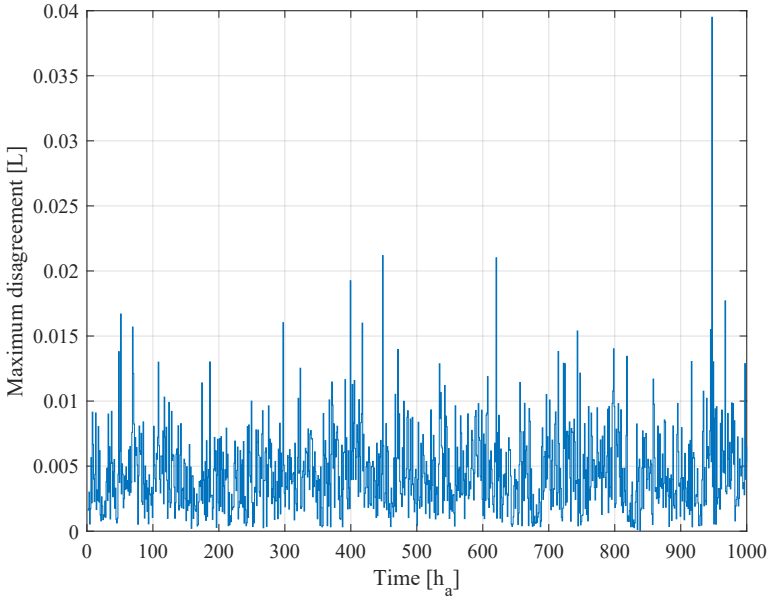


Figure 5.8. Maximum disagreement between stakeholders in the predicted volume of water in the tower.

5.1.5 Stopping Criteria

The purpose of the stopping criterion is to determine when enough iterations have been computed to reach a desired level of `performance` and consensus. One way to do this is to terminate the process after a predetermined number of iterations. From Figure 5.7, it is observed that for most cases, the `performance` is satisfyingly close to zero after 125 iterations, making this a potential stopping criterion. However, some hours require significantly fewer iterations, resulting in unnecessary use of computational power when continuing until the 125th iteration. In addition, there are instances that may require more iterations. Therefore, a different stopping criterion is used based on the solution's optimality.

Boyd, Parikh, Chu, *et al.* [14, Sec. 3.3.1] suggests using the stopping criterion in (5.7), using the primal and dual residuals. As previously mentioned, the primal residual r^k approaches zero as the difference between $\mathbf{x}_i$ and $\mathbf{z}$ decreases. Meanwhile, the dual residual s^k approaches zero as $\mathbf{x}_i$ approaches a constant value, indicating a minimum.

$$||r^k||_2 \leq \epsilon^{\text{pri}} \tag{5.7a}$$

$$||s^k||_2 \leq \epsilon^{\text{dual}} \tag{5.7b}$$

In (5.7), $\epsilon^{\text{pri}} > 0$ and $\epsilon^{\text{dual}} > 0$ is known as feasibility tolerances. Boyd, Parikh, Chu, *et al.* [14, Sec. 3.3.1] suggests determining these as shown in (5.8) for general ADMM. It is desired to rewrite this such that it can be applied to consensus ADMM. Therefore, sums are needed due to the presence of multiple $\mathbf{x}_i$ and $\boldsymbol{\lambda}_i$. Furthermore $\mathbf{A} = \mathbf{I} \in \mathbb{R}^{N_\text{q}N_\text{c} \times N_\text{q}N_\text{c}}$, $\mathbf{B} = -\mathbf{I} \in \mathbb{R}^{N_\text{q}N_\text{c} \times N_\text{q}N_\text{c}}$, $c = 0$, and $p = n = N_\text{q}N_\text{c}$. With these rewrites, (5.9) is obtained. Boyd, Parikh, Chu, *et al.* [14, Sec. 3.3.1] suggests using $\epsilon^{\text{rel}} = 10^{-3}$ or 10^{-4}. Whereas ϵ^{abs} depends on the optimisation problem.

$$\epsilon^{\text{pri}} = \epsilon^{\text{abs}}\sqrt{p} + \epsilon^{\text{rel}} \max(||\mathbf{A}\mathbf{x}^k||_2, ||\mathbf{B}\mathbf{z}^k||_2, ||c||_2) \tag{5.8a}$$

$$\epsilon^{\text{dual}} = \epsilon^{\text{abs}}\sqrt{n} + \epsilon^{\text{rel}}||\mathbf{A}^\top \boldsymbol{\lambda}||_2 \tag{5.8b}$$

Where:

p	Number of constraints
n	Number of optimisation variables

$$\epsilon^{\text{pri}} = \epsilon^{\text{abs}}\sqrt{N_\text{q}N_\text{c}} + \epsilon^{\text{rel}} \max\left(\sum_{i=1}^{N_\text{s}} ||\mathbf{x}_i^k||_2, N_\text{s}||\mathbf{z}^k||_2\right) \tag{5.9a}$$

$$\epsilon^{\text{dual}} = \epsilon^{\text{abs}}\sqrt{N_\text{q}N_\text{c}} + \epsilon^{\text{rel}} \sum_{i=1}^{N_\text{s}} ||\boldsymbol{\lambda}_i||_2 \tag{5.9b}$$

The suggested algorithm in (5.9), requires communication between stakeholders to determine the sum of $||\boldsymbol{\lambda}_i||_2$ and $||\mathbf{x}_i||_2$. In addition, this would require revealing more information, as all stakeholders would have to know the sum of $||\boldsymbol{\lambda}_i||_2$ and $||\mathbf{x}_i||_2$. To avoid potentially unnecessary communication and to keep the amount of information known by all stakeholders to a minimum, it is decided to investigate the effect of the stopping criterion using (5.9) compared to having a constant ϵ^{pri} and ϵ^{dual}. These stopping criteria will henceforth be referred to as varying and constant stopping criterion respectively.

In both implementations, $||r^k||_2$ and $||s^k||_2$ are determined as shown in (5.5c) and (5.5d) respectively, which requires communication between stakeholders. To reduce this communication, it is chosen only to determine whether the stopping criteria are met every fifth iteration. Additionally, the stopping criteria are not evaluated before the penalty parameter has been increased, i.e. the stopping criteria are evaluated from iteration 35 and onward. To avoid an unexpectedly long computation time, an upper limit of 500 iterations is chosen.

The values of the different variables for the varying and constant stopping criteria are presented in Table 5.2. Note that for the varying stopping criterion, ϵ^{pri} and ϵ^{dual} are created using ϵ^{rel} and ϵ^{abs}. ϵ^{rel} is chosen based on suggestion in Boyd, Parikh, Chu, *et al.* Meanwhile, ϵ^{abs} is hand-tuned by decreasing from 1 until satisfying `performance` for the first hour of simulation. ϵ^{pri} and ϵ^{dual} for the constant stopping criterion are chosen as the mean of the last values of ϵ^{pri} and ϵ^{dual} using the varying stopping criterion.

	ϵ^{rel}	ϵ^{abs}
Varying	10^{-3}	0.05

	ϵ^{pri}	ϵ^{dual}
Constant	0.07	0.06

Table 5.2. Values for the varying and constant stopping criteria.

Figure 5.9 shows the number of iterations used for both the constant and varying stopping criterion, while Figure 5.10 shows the `performance` of the controller using the two stopping criteria, as well as the `performance` of a stopping criterion of 125 iterations. Extended versions of the figures can be found in Appendix H. Figure 5.9 illustrates that the number of iterations for constant and varying stopping criteria are almost equal for the vast majority of hours. Furthermore, from Figure 5.10, it can be observed how the `performance` of the constant and varying stopping criteria only deviate from each other at a few instances.

The stopping criterion of 125 iterations also shows great `performance`; however, from Figure 5.9, it is observed that sometimes as low as 50 iterations are enough to achieve the desired `performance`, resulting in unnecessary additional computational power usage. Figure 5.9 shows that the constant stopping criterion reaches the maximum iteration number of 500 iterations at hour 357. Meanwhile, the varying stopping criterion stops right before the limit is reached. Nevertheless, allowing these extra iterations is reflected in their `performance`, as it can be seen how the `performance` of the stopping criterion of 125 iterations spikes at this point. From these results, it is decided to implement the constant stopping criterion, as the benefits from the varying stopping criterion are not deemed significant enough to justify the extra communication and revealed information.

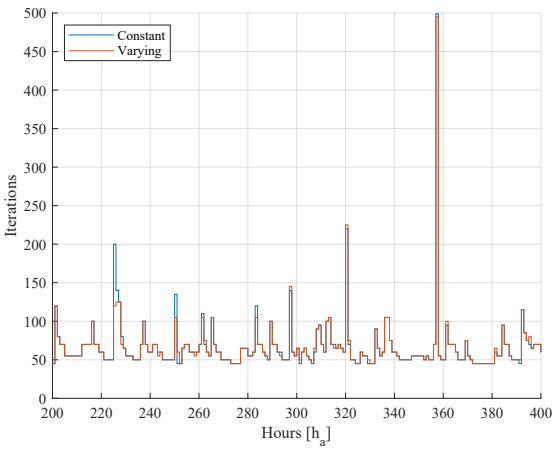


Figure 5.9. Number of iterations needed with constant and varying stopping criteria.

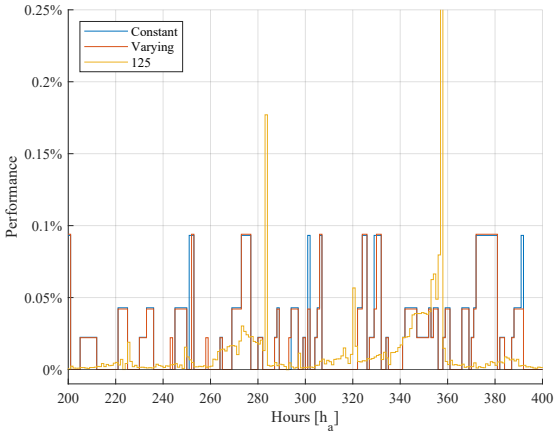


Figure 5.10. `Performance` after 125 iterations, constant stopping criterion, and varying stopping criterion.

5.2 Simulation of Performance

In this section, it is desired to evaluate the performance of the distributed controller by comparing it to the global controller from Chapter 4.

Both controllers are simulated for 10 days, with an exact plant model and the predicted flow into the consumption group described in Section 3.3. The results are shown in Figure 5.11, for the first 80 h, together with the prediction. The figure shows that despite having a non-convex cost function, both controllers achieved similar water volume and summed pump station flows, both in the past and in the prediction. Furthermore, both controllers demonstrated proactive behaviour by increasing the volume of water in the tower prior to increases in electricity prices.

From Figure 5.11, it is observed that the constraints are not always fulfilled. This is due to the difference between predicted and actual consumption. Rectifying this, e.g. by using stochastic MPC, is out of the scope of this project, as described in Section 4.3.

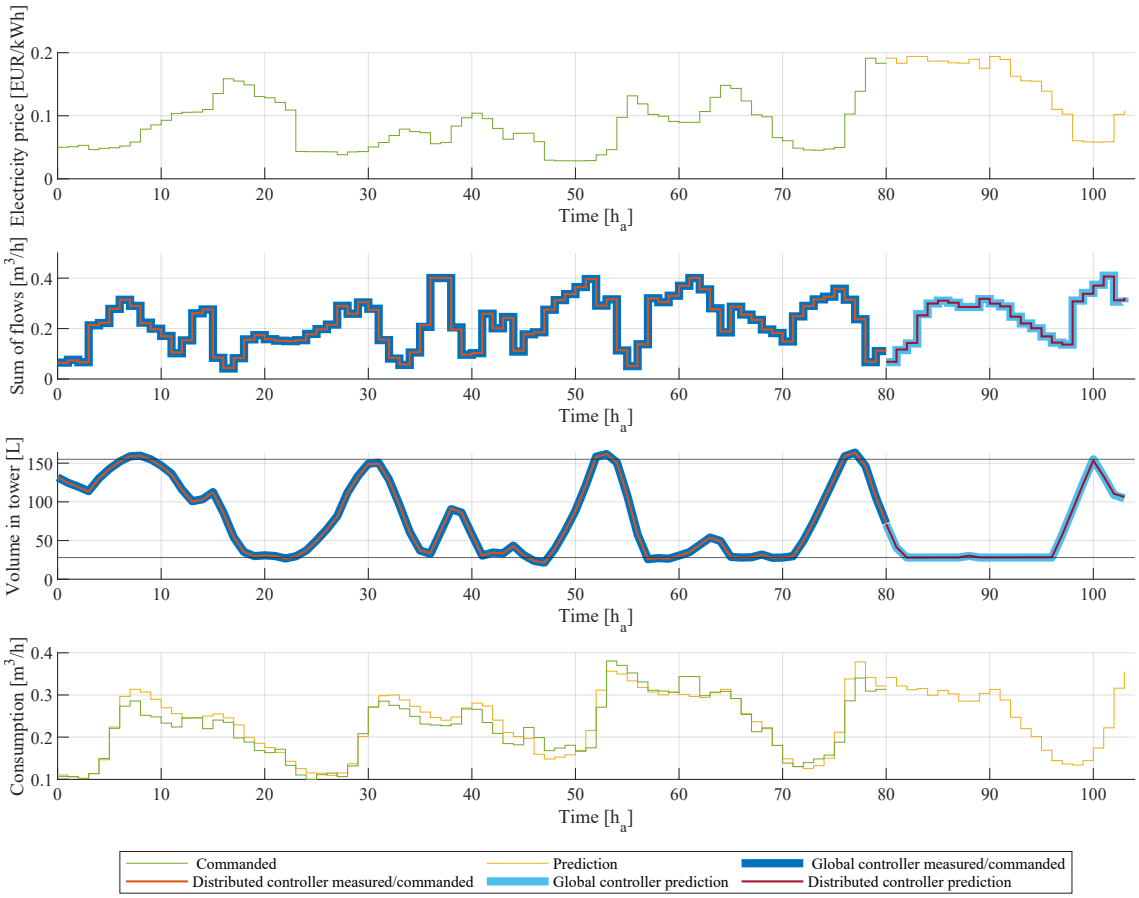


Figure 5.11. Simulated results for the global and distributed controllers.

The electricity bill for both the global and distributed controller is 0.89 Eurocent. Based on this and Figure 5.11, it is concluded that the distributed controller performance is identical to that of the global controller despite the non-convex cost function. In comparison with the ON/OFF controller simulated in Section 4.2, both optimising controllers achieve equal reductions in the electricity bill of 18.3 %.

In conclusion, the development of the distributed controller has been successful, showing equal performance to the global controller in reducing costs, fulfilling constraints, and proactive response to external factors. The code for simulating the distributed controller can be found in link 3 in Appendix A.

5.3 Alternative Distributed Controller

During the project, several attempts have been made to find a method for solving the optimisation problem without revealing one pump station's predicted flows to another pump station. This can be achieved through sharing ADMM. However, this cannot be applied to the cost function used in this project. The purpose of this section is to present this method as an inspiration. The method presented in this section will neither be simulated nor implemented.

The sharing ADMM algorithm can be used to solve sharing problems as the one given in (5.10). The cost function is a sum of N_s sub-cost functions $f_i(\mathbf{x}_i)$ plus a function g, which is a shared objective. The constraint defines that $\mathbf{x}_i$ and $\mathbf{z}_i$ must be equal, but unlike the consensus algorithm, the $\mathbf{x}_i$'s can be different.

$$\min_{\mathbf{x}_i,\mathbf{z}_i} \sum_{i=1}^{N_\mathrm{s}} f_i(\mathbf{x}_i) + g\left(\sum_{i=1}^{N_\mathrm{s}} \mathbf{z}_i\right) \tag{5.10a}$$

$$\text{s.t.} \quad \mathbf{x}_i - \mathbf{z}_i = 0, \quad i = 1, \ldots, N_\mathrm{s} \tag{5.10b}$$

The sharing ADMM algorithm is given in (5.11) [14, Sec. 7.3] and is a specific case of ADMM. In the algorithm, $\bar{\mathbf{x}}^k$ is defined as shown in (5.12), which is the only part of the algorithm that cannot be calculated individually by each stakeholder. The Lagrange multiplier in the sharing ADMM algorithm has been substituted for $\mathbf{u}$ which is possible by rewriting the problem. See Boyd, Parikh, Chu, *et al.* [14, Sec. 3.1.1 & 7.3] for the rewriting.

$$\mathbf{x}_i^{k+1} = \arg\min_{\mathbf{x}_i} \left(f_i(\mathbf{x}_i) + \frac{\rho}{2} ||\mathbf{x}_i - \mathbf{x}_i^k + \bar{\mathbf{x}}^k - \bar{\mathbf{z}}^k + \mathbf{u}^k||_2^2 \right) \tag{5.11a}$$

$$\bar{\mathbf{z}}^{k+1} = \arg\min_{\bar{\mathbf{z}}} \left(g(N_\mathrm{s}\bar{\mathbf{z}}) + \frac{N_\mathrm{s}\rho}{2} ||\bar{\mathbf{z}} - \mathbf{u}^k - \bar{\mathbf{x}}^{k+1}||_2^2 \right) \tag{5.11b}$$

$$\mathbf{u}^{k+1} = \mathbf{u}^k + \bar{\mathbf{x}}^{k+1} - \bar{\mathbf{z}}^{k+1} \tag{5.11c}$$

$$\bar{\mathbf{x}}^k = \frac{1}{N_\mathrm{s}} \sum_{i=1}^{N_\mathrm{s}} \mathbf{x}_i^k \tag{5.12}$$

To obtain the improved privacy-preserving features, $\mathbf{x}_i$ must only hold the predicted flows for pump station i. This is not possible in this project due to the terms in the cost function (4.2) regarding the water volume and combined pipe resistance since they have flows from different pump stations multiplied with each other. Thereby, the cost function in this project requires $\mathbf{x}_i$ to hold $\boldsymbol{\mathcal{U}}_k$. This would not be required if each pump station

was connected directly to the top of the tower instead of a shared pipe at the bottom. In that case, f_i could hold the electricity bill for each pump station, while g held J_{V}. This would result in a significantly more privacy-preserving algorithm.

5.4 Summary

This chapter described how consensus ADMM can be used to solve the optimisation problem distributed. The only communication between the stakeholders solving the problem is calculations of sums, which can be made privacy-preserving using SMPC as described in the following chapter. The plant and distributed controller have been simulated using an exact plant model, resulting in the same operation as the global controller. Lastly, a more privacy-preserving distributed algorithm, which can not solve this project's optimisation problem, has been suggested as inspiration.

Privacy-Preserving Summation

6

In the previous chapter, a distributed method for solving optimisation problems was presented. This method becomes privacy preserving if (5.2b), (5.5b), and (5.5c), restated in (6.1), can be calculated without disclosing all variables with subscript i. This chapter describes how this can be achieved using Secure Multi-Party Computation (SMPC).

The equations in (6.1), which include both summation and division, are split up into calculations before, during, and after privacy-preserving summation, as shown in Table 6.1. Henceforth, the results of the calculation before summations will be referred to as secrets, where subscript i denotes which stakeholder the secret belongs to. The computations before and after SMPC is applied will be performed by each stakeholder individually. Thus, only the sums in the second column, the result gained from SMPC, will be known by all stakeholders.

$$\mathbf{z}^{k+1} = \frac{1}{N_\mathrm{s}} \sum_{i=1}^{N_\mathrm{s}} \left(\mathbf{x}_i^{k+1} + \frac{1}{\rho} \boldsymbol{\lambda}_i^k \right) \in \mathbb{R}^{N_\mathrm{q} N_\mathrm{c}} \tag{6.1a}$$

$$\bar{\mathbf{x}}^k = \frac{1}{N_\mathrm{s}} \sum_{i=1}^{N_\mathrm{s}} \mathbf{x}_i^k \in \mathbb{R}^{N_\mathrm{q} N_\mathrm{c}} \tag{6.1b}$$

$$||r^k||_2^2 = \sum_{i=1}^{N_\mathrm{s}} ||\mathbf{x}_i^k - \bar{\mathbf{x}}^k||_2^2 \in \mathbb{R} \tag{6.1c}$$

	Before summation (Secret)	SMPC / Sum	After summation
(6.1a)	$s_{\mathbf{z}_i} = \mathbf{x}_i + \frac{1}{\rho}\boldsymbol{\lambda}_i$	$S_\mathbf{z} = \sum_{i=1}^{N_\mathrm{s}} s_{\mathbf{z}_i}$	$\mathbf{z} = \frac{1}{N_\mathrm{s}} S_\mathbf{z}$
(6.1b)	$s_{\bar{\mathbf{x}}_i} = \mathbf{x}_i$	$S_{\bar{\mathbf{x}}} = \sum_{i=1}^{N_\mathrm{s}} s_{\bar{\mathbf{x}}_i}$	$\bar{\mathbf{x}} = \frac{1}{N_\mathrm{s}} S_{\bar{\mathbf{x}}}$
(6.1c)	$s_{r_i} = \|\|\mathbf{x}_i - \bar{\mathbf{x}}\|\|_2^2$	$S_r = \sum_{i=1}^{N_\mathrm{s}} s_{r_i}$	$\|\|r\|\|_2^2 = S_r$

Table 6.1. Split of (6.1) to local computations before and after SMPC. The superscript, indicating the iteration number, is omitted for simplicity.

6.1 Secure Multi-Party Computation

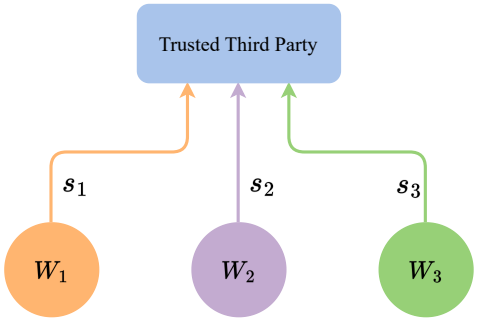
One approach to handling computation, in this case summation, without disclosing each stakeholder's secret, is to involve a trusted third party. Each stakeholder can disclose their

secret to the trusted party, which can perform the necessary computations and distribute the results. This principle is illustrated in Figure 6.1a, where s_i is the secret of stakeholder W_i. However, finding a third party, which indeed can be trusted, may not be plausible. One way to allow for such computations without the need for a third party is through the use of SMPC [19, p. 6].

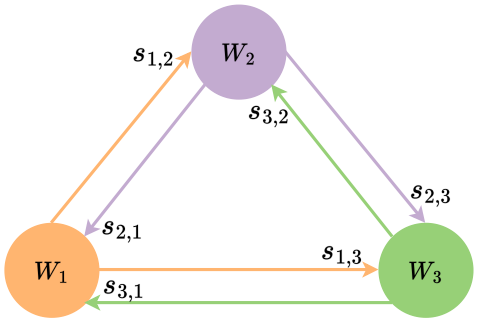
Several methods for performing SMPC exist; however, this report will focus solely on secret sharing. Furthermore, as only addition is required in this project, how other computations would be performed will not be described. For the interested reader, an extensive description of SMPC can be found in [19].

An illustration showing the principle of secret sharing with three stakeholders can be seen in Figure 6.1b. Here, each stakeholder W_i divides its secret s_i into three shares $s_{i,1}, s_{i,2}, s_{i,3}$, where all three shares are required to obtain the secret. The stakeholder then distributes one share to each of the other stakeholders, keeping one share $s_{i,i}$ to itself. Computations can then be performed on the shares, where the result is distributed to all stakeholders. From the result, the stakeholders can calculate the sum. The method for splitting the secrets into shares and how to reconstruct the sum depends on the secret sharing scheme.

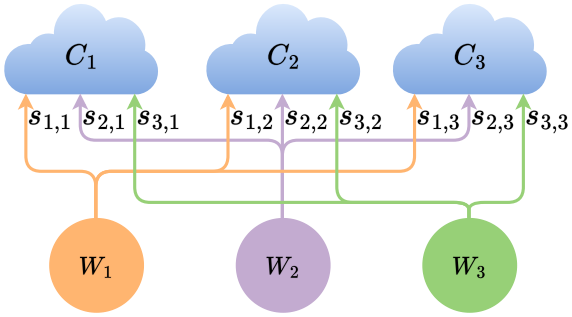
Alternatively, cloud-based stakeholders can be introduced, to which the stakeholders forward shares of their secrets. The clouds then perform the computations on the shares and distribute the results to the stakeholders. This principle is illustrated in Figure 6.1c with three cloud-servers.



(a) The principle of using a trusted third party.



(b) The principle of secret sharing.



(c) The principle of secret sharing using cloud servers.

Figure 6.1. Privacy-preserving computation methods.

A minimum of three stakeholders are required to ensure privacy in the summation. With only two stakeholders, a stakeholder's own secret and the sum is enough to learn the other stakeholder's secret, i.e. both stakeholders would know the other's secret.

This project defines privacy-preserving as protection against passive attacks, as well as eavesdropping on communication lines. For this reason, utilising a trusted party is undesired since eavesdropping on any communication line would leak a secret. Furthermore, all secrets are leaked in case of passive attacks on the trusted third party. This is the same vulnerability as for the global controller described in Chapter 4.

By distributing shares of the secrets among the stakeholders, passive attacks on one stakeholder would only leak that stakeholder's secret. In addition, eavesdropping on a single communication line would leak no information. Therefore, only eavesdropping on several communication lines or passive attacks on several stakeholders would potentially leak more than a single secret. Utilising cloud servers presents the same vulnerabilities regarding passive attacks and eavesdropping. For this reason, introducing cloud servers would not present any benefits, therefore, the method illustrated in Figure 6.1b will be utilised.

Using the principals illustrated in Figure 6.1b and 6.1c secrets can be obtained by an adversary by eavesdropping on a sufficient number of communication lines. For this reason, the communication between stakeholders should be encrypted to increase protection, in case the communication lines utilise the same network equipment. Encryption is not applied in this project; however, standard libraries exist.

Several secret sharing schemes for creating shares exist. Since only addition is required of the SMPC algorithm, a simple additive secret sharing scheme would be sufficient [20, Sec. 2.2]. However, it is chosen to focus solely on Shamir's Secret Sharing Scheme (Shamir's Scheme), as it allows for detecting deviations from protocol, which will be discussed in later chapters.

6.2 Shamir's Secret Sharing Scheme

To obtain perfect privacy-preservation, Shamir's Scheme utilises finite fields. Therefore, all calculations will be performed in the finite field $\mathbb{Z}_\beta$, which is a range of integers from 0 to $\beta - 1$, where β is a prime larger than the sum. To cast values into the finite field, mod β is used.

Shamir's Scheme is based on the principle of polynomial interpolation, i.e., given n points in a 2-dimensional plane, there is only one polynomial of degree $n - 1$ that goes through all n points [21]. The secret s is hidden at $x = 0$ of a polynomial $b(x)$:

$$b(x) = s + a_1 x + \cdots + a_{n-1} x^{n-1} \bmod \beta \tag{6.2}$$

Where the values of a_i are selected randomly from a uniform distribution across $\mathbb{Z}_\beta$. The uniform distribution guarantees privacy by ensuring that all values are equally likely, whereas, for a normal distribution, privacy is not guaranteed [5, p. 176]. The secret is split into N_s shares according to a set C, where $N_\mathrm{s} \geq n$, with each share being a point on

the polynomial in C, i.e., $b(1), b(2), \ldots, b(N_\mathrm{s})$ for $C = \{1, 2, \ldots N_\mathrm{s}\}$. The distribution of C is not confined to chronological order, nor is it required to start at 1.

Due to the principle of polynomial interpolation, anyone in possession of less than n of the shares cannot discern any information about the secret, given the potential for an infinite number of distinct polynomials. Meanwhile, n or more shares can be used to reconstruct the secret using Lagrange Interpolation.

6.2.1 Lagrange Interpolation

A short introduction to Lagrange interpolation is given here; for the interested reader, see Cramer, Damgaard, and Nielsen [19, p. 33-35]. Note that all divisions must be performed within the finite field $\mathbb{Z}_\beta$. An example of how this is done is presented in Appendix I.

Given a secret within a finite field $\mathbb{Z}_\beta$, located at $x = 0$ of a $t < n$ degree polynomial with n known data points, it is possible to determine the secret by utilising (6.3).

$$b(x) = \sum_{i \in C} \delta_i(x) b(i) \bmod \beta \tag{6.3}$$

Where $\delta_i(x)$ is known as the Lagrange basis polynomial and is given by:

$$\delta_i(x) = \prod_{j \in C,\, j \neq i} \frac{x - j}{i - j} \bmod \beta \tag{6.4}$$

Since the secret is present at $x = 0$, it is possible to write:

$$\delta_i(0) = \prod_{j \in C,\, j \neq i} \frac{-j}{i - j} \bmod \beta \tag{6.5}$$

As $\delta_i(0)$ is independent of $b(i)$, it can be calculated in advance. It is possible to write (6.3) on matrix form for $x = 0$ and $C = \{1,2,3\}$ as:

$$b(0) = \underbrace{\begin{bmatrix} \delta_1(0) & \delta_2(0) & \delta_3(0) \end{bmatrix}}_{\boldsymbol{\delta}} \begin{bmatrix} b(1) \\ b(2) \\ b(3) \end{bmatrix} \bmod \beta \tag{6.6}$$

Where $\boldsymbol{\delta}$ is known as the recombination vector.

6.2.2 Algorithm

Algorithm 1 goes through the steps of Shamir's Scheme where each stakeholder W_i has its own secret s_i, and it is desired to calculate the sum of these secrets S. For simplicity, $C = \{1,2,\ldots,N_\mathrm{s}\}$ in the algorithm below. An illustration of the principle with three stakeholders and three shares can be seen in Figure 6.2.

Algorithm 1 Additive Shamir's Scheme [19, p. 37-39]

Each stakeholder W_i for $i = 1, \ldots, N_s$ holds a secret $s_i \in \mathbb{Z}_\beta$

1. Each stakeholder W_i generates a polynomial $b_i(x)$ of degree $t < N_s$, where $b_i(0) = s_i$ and $a_{i,1}, \ldots, a_{i,t}$ are uniformly random distributed in $\mathbb{Z}_\beta$:

$$b_i(x) = s_i + a_{i,1}x + \cdots + a_{i,t}x^t \mod \beta$$

   where $\sum_{i=1}^{N_s} b_i(x) = b(x)$ and $\sum_{i=1}^{N_s} b_i(0) = b(0) = \sum_{i=1}^{N_s} s_i = S$

2. Each stakeholder W_i generates shares of their secret as $s_{i,1}, \ldots, s_{i,N_s} = b_i(1), \ldots, b_i(N_s)$ and distributes a single share to each other stakeholder, keeping $b_i(i)$ to itself.

3. Each stakeholder W_i computes the sum of shares $b(i) = \sum_{j=1}^{N_s} b_j(i) \mod \beta$ and broadcasts the result.

4. Each stakeholder computes the sum $S = b(0)$ using (6.6).

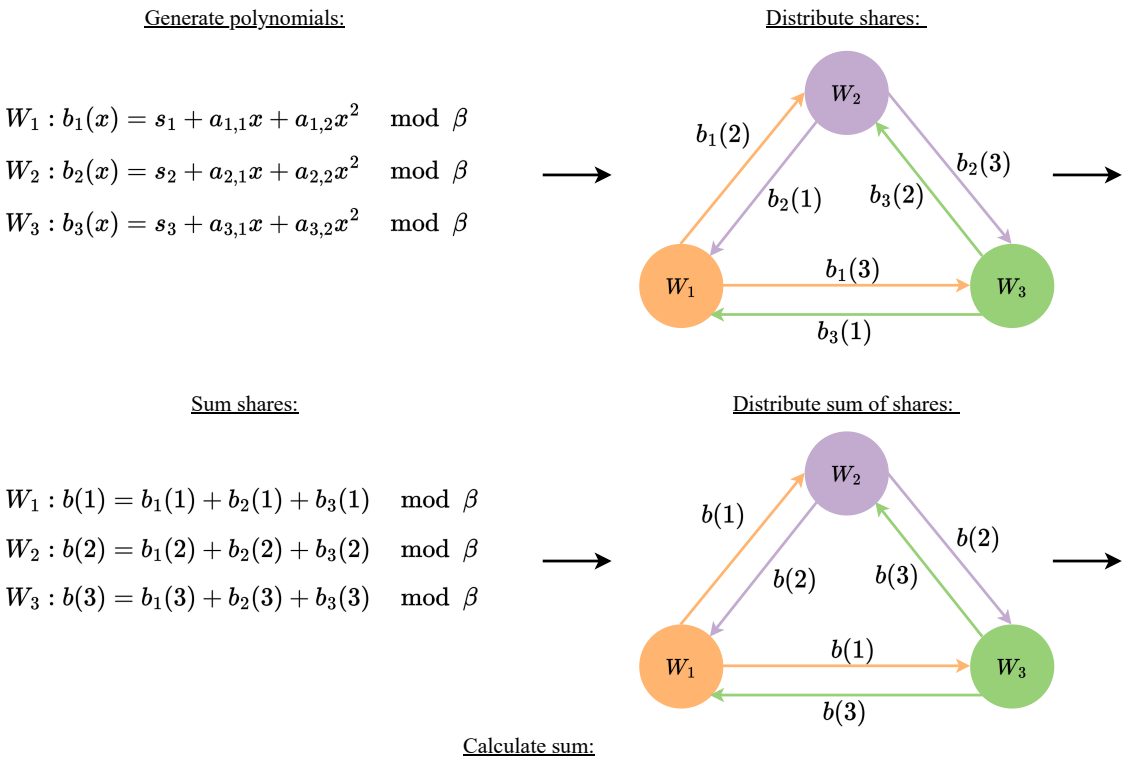


Figure 6.2. Illustration of how a sum of secrets can be computed in a privacy-preserving manner using Shamir's Scheme and Lagrange interpolation.

6.3 Implementation

Each of the three stakeholders: the two pump stations and the water tower holds their own secrets: $s_{\mathbf{z}_i}$, $s_{\bar{\mathbf{x}}_i}$, and s_{r_i} from Table 6.1. It is decided that the sum of shares from all three stakeholders are required to recreate the polynomial, meaning it must be of second degree. Furthermore, $C = \{1,2,3\}$ is utilised, where the water tower holds $b_j(1)$, pump station 1 holds $b_j(2)$, and pump station 2 holds $b_j(3)$. Before the algorithm can be implemented, the size of the finite field, as well as a method for casting the secrets into the finite field, have to be defined.

As previously mentioned, all secrets and sums must be within the field, i.e., they must be cast to integers. To avoid introducing rounding errors, scaling the secrets is necessary. Given the maximum flow of $0.3\,\mathrm{m}^3\,\mathrm{h}^{-1}$, present in $s_{\bar{\mathbf{x}}_i}$, it is evident that a significant number of decimals will be required for the calculations. A scaling factor of $\gamma = 10\,000$ is selected to ensure that the rounding error is negligible.

The size of the finite field β has to be larger than the sums, which means it is necessary to determine their largest possible value. In Section 5.1.3, a 1 000 hour simulation is performed, where the range for each sum is shown in Table 6.2. Here, it is observed that the largest value is 1.5. The table also shows the minimum and maximum values for the secrets. From this, it can be seen that the secret $s_{\mathbf{z}_i}$ can become negative. While this did not yield a negative sum, it is still required that each of the individual secrets are located within the finite field [19, p. 38]. A negative $s_{\mathbf{z}_i}$ can occur when λ_i^{k+1}, see (5.2c), is negative. To circumvent negative secrets, an offset is added to guarantee that all secrets are positive. From the table, it can be seen that the lowest value of $s_{\mathbf{z}_i}$ is -0.5. As a result, an offset $\varepsilon = 5$ is chosen.

Secret	$s_{\mathbf{z}_i}$	$s_{\bar{\mathbf{x}}_i}$	s_{r_i}	Sum	$S_{\mathbf{z}}$	$S_{\bar{\mathbf{x}}}$	S_r
Min value	-0.5	0	0	Min value	0	0	0
Max value	0.7	0.4	0.7	Max value	1.1	1.1	1.5

Table 6.2. Minimum and maximum values for the sums and secrets.

With the scaling and offset, the scaled secret is:

$$s_{i_{\text{scaled}}} = \gamma(s_i + \varepsilon) \tag{6.7}$$

Where the descaling to determine the sum can be performed as:

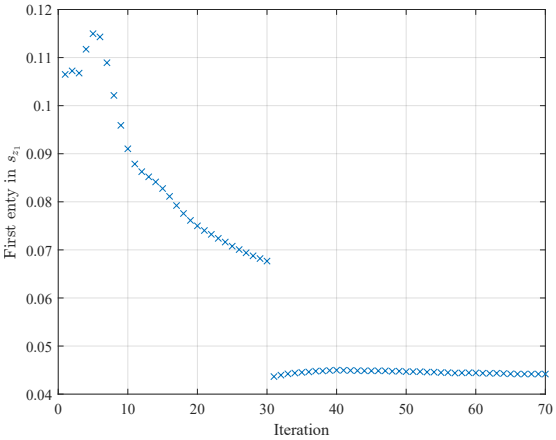
$$S = \frac{(S_{\text{scaled}} - N_{\text{s}}\varepsilon)}{\gamma} \tag{6.8}$$

With both offset and scaling implemented, the largest sum can be found to be: $(1.5 + 5)10\,000 = 65\,000$. However, due to unpredictable electricity prices and unknown future predictions of water consumption, there is no guarantee that the sum will not become larger. Since the number will be represented as an unsigned 32-bit integer, it is decided to choose the largest prime that the integer can hold, namely $\beta = 4\,294\,967\,029$.

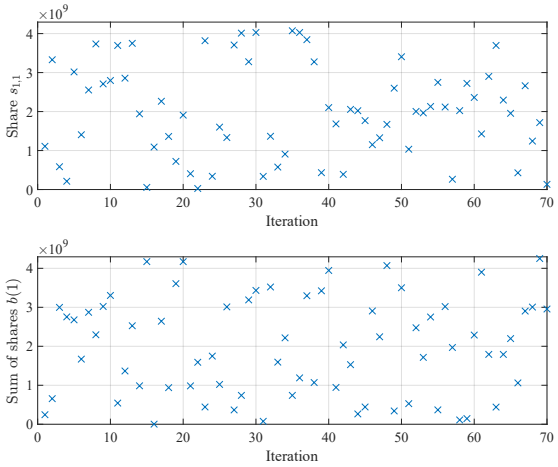
The code for the implementation of Shamir's Scheme can be found in link 4 in Appendix A.

6.4 Results

In this section, it is desired to show the communication between stakeholders with and without Shamir's Scheme. Figure 6.3 shows the messages sent from stakeholder 1 to stakeholder 2 to compute the first entry in $S_{\mathbf{z}}$. Figure 6.3a shows the first entry in the secret $s_{\mathbf{z}_1}$, which is sent to stakeholder 2 and 3 when Shamir's Scheme is not applied. In contrast, Figure 6.3b shows the first entry in the share $s_{1,2}$ and the first entry in the sum of shares $b(1)$ sent from stakeholder 1 to stakeholder 2. Figure 6.3b resembles uniformly distributed noise, while Figure 6.3a shows the secret. Thereby, the communication has been made privacy-preserving.



(a) Messages sent by stakeholder 1 to stakeholder 2 without Shamir's Scheme.



(b) Messages sent by stakeholder 1 to stakeholder 2 with Shamir's Scheme.

Figure 6.3. Effects of applying Shamir's Scheme.

6.5 Summary

In this chapter, the principle of SMPC was introduced, with the purpose of calculating a sum of contributions from each stakeholder without revealing individual contributions. This was achieved through the application of Shamir's Scheme. Since Shamir's Scheme operates within finite fields, the secrets are cast into integers before the sums are calculated, after which the sums are cast back into decimals. Finally, it has been shown that communication between stakeholders after Shamir's Scheme has been applied resembles uniformly distributed noise.

7 Implementation

This chapter describes the laboratory implementation of the optimising control algorithms, followed by testing and evaluating their performance.

Both controllers are implemented in Python, as it was desired to have the privacy-preserving controller distributed on the Raspberry Pis of the stakeholders, where a Matlab installation could prove troublesome. The global controller runs directly on the laboratory PC; however, it is implemented in Python to ensure fairer comparison.

For both controllers, it was desired to use the optimisation tool Casadi [22] to solve the optimisation problem. Unfortunately, the standard distribution of Casadi does not support solving nonlinear optimisation problems on the 32-bit Linux operating system in the laboratory. It is possible to upgrade the operating system to 64-bit. However, this would require knowledge of all software settings in the current laboratory setup and, therefore, deemed out of this project's scope.

An attempt was made to change the solver to SciPy.optimize [23], which does support 32-bit Linux systems. However, SciPy.optimize spent 12 s solving the stakeholder's local optimisation problem when implemented on a Raspberry Pi. This is deemed unacceptable, considering the required number of iterations in the consensus ADMM algorithm. Even on a laptop, the computation time was 4 s using SciPy.optimize, while only being 20 ms using Casadi. Therefore, the long computation time on the Raspberry Pi is assumed to be a result of the implementation of SciPy.optimize and not the complexity of the optimisation problem.

Instead of making a distributed implementation on the Raspberry Pis, the privacy-preserving controller is implemented on the laboratory PC using dedicated terminals for each stakeholder. The stakeholders communicate using TCP to emulate a distributed system as shown in Figure 7.1. The TCP communication between the stakeholders is set up as described in Appendix C. The low-level controllers described in Section 2.1.2 receive references from the optimising controllers and send actuation commands to the soft PLCs using ModbusTCP.

The implemented controllers can be found at link 5.

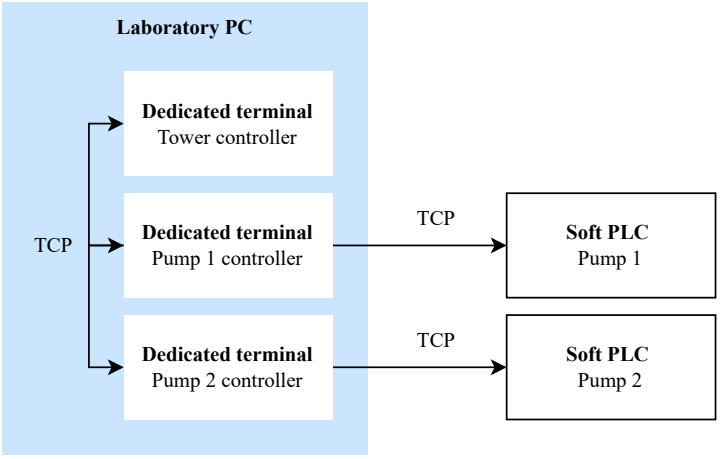


Figure 7.1. TCP communication between stakeholders and primary actuation commands from low-level controllers to soft PLCs at laboratory modules.

7.1 Results

Figure 7.2 and 7.3 show the global and privacy-preserving controllers, respectively, at a given time step. Two animations at link 6 in Appendix A show the same information with updating prediction as time progresses.

Figure 7.2, Figure 7.3, and the animations show that the flow measurements correspond to the actuation commands except close to hour 80. Meanwhile, the consumption measurements correspond to the actuation commands except for a few hours close to hours 48 and 68. These differences are caused by the safety level control described in Section 2.1.2, as the constraints of the water tower are broken. This issue was noted in simulation as well, as described in Section 4.3, where it was chosen not to implement possible solutions.

The volume of water in the tower is increased before an increase in electricity price, which is the desired performance. The animations show that the predicted actuation is close to the applied actuation, even at the end of the horizon.

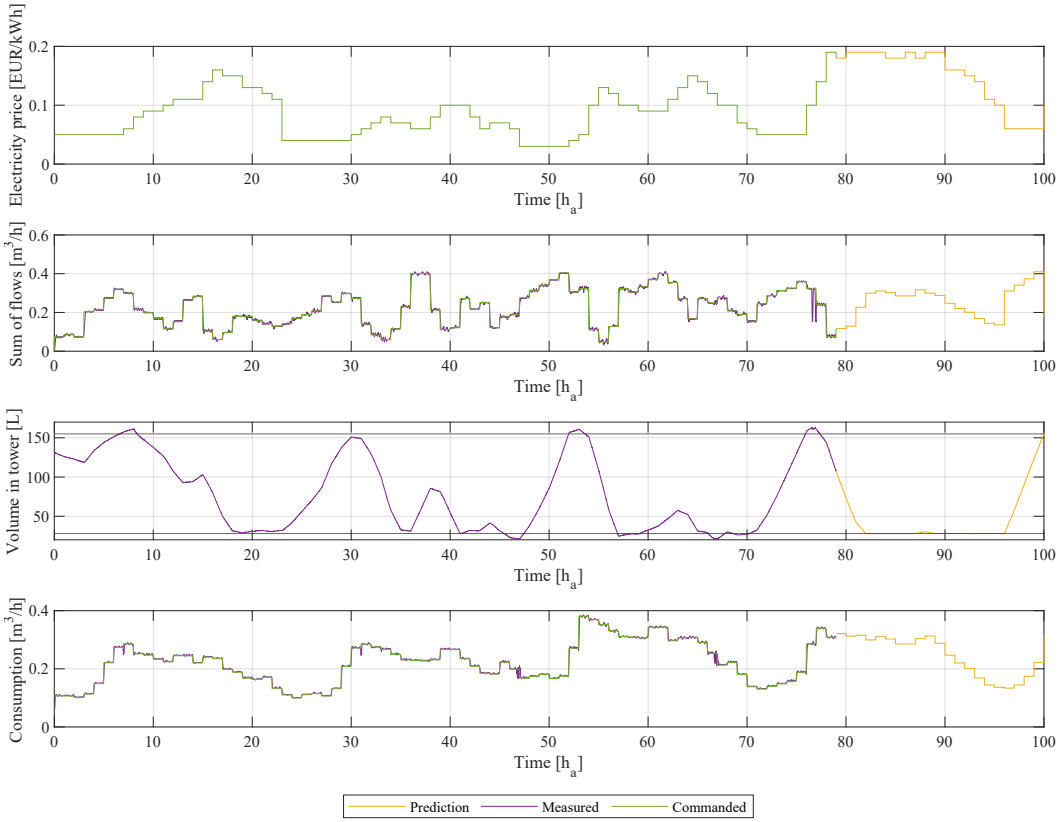


Figure 7.2. Operation of the global controller.

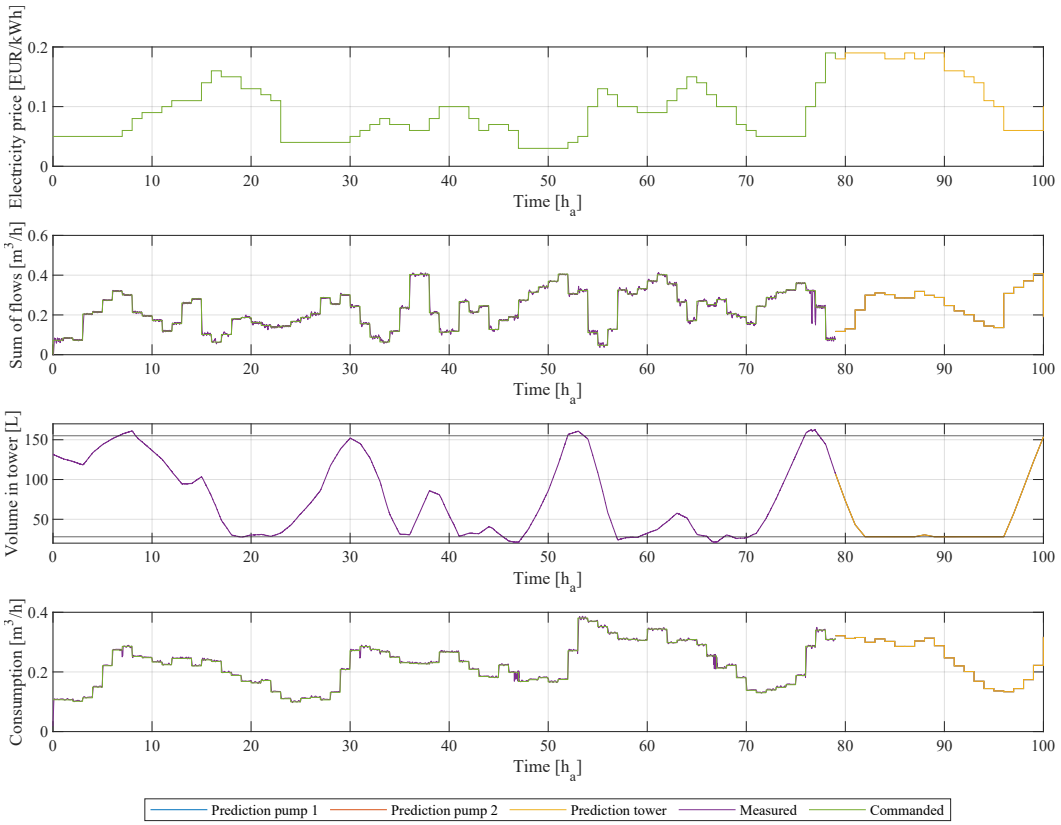


Figure 7.3. Operation of the privacy-preserving controller.

In Figure 7.4, the commanded flows and volume of water in the tower for the two implemented controllers are compared to those of the simulated. As determined in Section 5.2, the simulation of the distributed and global controller are practically identical; thus, only the simulated global controller is used for this comparison. The figure shows that the implementation is nearly identical to the simulation, i.e., the controllers work as desired.

The electricity bills of the implementation of the global and privacy-preserving controllers have been estimated to 0.894 Eurocent and 0.892 Eurocent, respectively. The difference in cost between the two is deemed insignificant. For the simulated controller, the estimated electricity bill is 0.888 Eurocent, which means the implementation has resulted in negligible change in cost. Therefore, implementing the privacy-preserving controller is deemed a success, as it performs exactly as the global controller while being privacy-preserving. Remark that this is the best possible outcome.

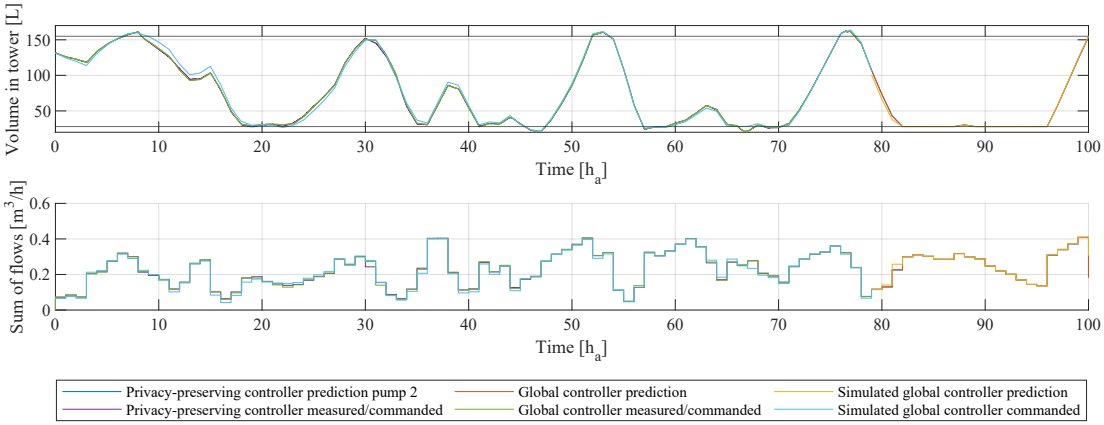


Figure 7.4. Comparison of simulated and implemented controllers.

7.2 Summary

The global and distributed privacy-preserving controllers have been implemented in the laboratory. The privacy-preserving controller are implemented using dedicated terminals for the stakeholders such that a distributed controller is emulated. The implemented controllers perform the same as the simulation and results in almost identical electricity billing. Thereby, the implementation of a privacy-preserving controller is deemed a success, as it does not result in loss of performance.

Further Considerations 8

This chapter discusses practical considerations regarding the implementation of the privacy-preserving controller in real-world settings. Moreover, the chapter will investigate the vulnerability to passive, eavesdropping, and active attacks.

8.1 Implementation Considerations

Implementing a privacy-preserving control algorithm does not come without drawbacks, such as increased communication, complexity, and computation time.

Using the global controller, sensor data is collected, and actuation is computed, after which actuation commands are sent to the actuators. In the privacy-preserving controller, sensor data is collected by the stakeholders, and actuation is computed using consensus Alternating Direction Method of Multipliers (ADMM), after which actuation commands are sent to actuators.

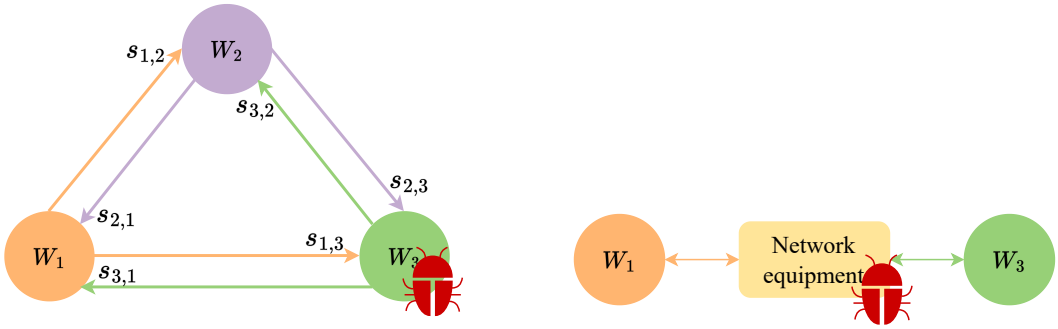
In the simulation, the ADMM-algorithm requires 67 iterations and 101 privacy-preserving summations on average. In the worst case, 500 iterations and 705 privacy-preserving summations are required. Note that 500 iterations is a hard limit set by the stopping criterion in Section 5.1.5. Thereby, communication is significantly increased, and even small communication delays result in significant computation time in the ADMM-algorithm. While this has not caused any issues in the laboratory, due to the high bandwidth and low delay of the network, it might pose a problem in real-world applications.

Another drawback is that operational staff will not be able to get a full understanding of the control algorithm at one interface. Troubleshooting on an interface where cost functions and constraints of individual stakeholders are inaccessible could prove quite challenging. Therefore, careful considerations must be taken to determine whether a plant can benefit from the suggested control algorithm. The algorithm might be most beneficial for plants where operational staff does not rely on a full understanding to maintain the plant while still being subject to a significant cyber security risk.

8.2 Cyber Attacks

The purpose of the distributed privacy-preserving algorithm is to protect against passive and eavesdropping attacks. In this section, it is desired to analyse what information can be gained should those attacks occur. In addition, the section investigates the possibility of protecting against active attacks.

The section will focus solely on two distinct points of attack, illustrated in Figure 8.1. In Figure 8.1a, the adversary targets one of the stakeholders directly. In contrast, Figure 8.1b illustrates an attack where a communication line between stakeholders is infected.



(a) Attack point (a): Attack directly on one of the stakeholders.

(b) Attack point (b): Attack on communication between two stakeholders.

Figure 8.1. Different points of attack, where the adversary is illustrated as a red bug.

8.2.1 Passive and Eavesdropping Attacks

In passive attacks, represented as attack point (a), an adversary infects a stakeholder directly. In contrast, in an eavesdropping attack, the adversary attacks communication between stakeholders, as shown in attack point (b). In both instances, the intention of the adversary is to gain information. In this section, a brief introduction to methods to gain information from the privacy-preserving control algorithm is given.

The constraints and cost functions are kept private using Secure Multi-Party Computation (SMPC) and ADMM. However, information can be inferred from the consensus variable $\mathbf{z}$ and the solution, which all stakeholders know. For instance, information about the maximum pump capacity and Total Yearly Extraction Limit (TYEL) can be inferred from a large number of solutions. One way to avoid this issue would be to implement the alternative privacy-preserving controller described in Section 5.3 based on ADMM sharing. Using the ADMM sharing algorithm, very little information can be inferred since the stakeholders only know their own flow and the sum of flows. However, ADMM sharing cannot be used to optimise the cost function in this project, as described in Section 5.3.

Should two of the three stakeholders be subject to passive attacks, the third stakeholder's secrets in the SMPC algorithm can be calculated from the sum and the infected stakeholders' secrets. Based on the solution to the local optimisation problem $\mathbf{x}_i^{k+1}$ and consensus variable $\mathbf{z}_i^{k+1}$, the gradient of the cost function $\nabla f_i(\mathbf{x}_i^k)$ can be inferred, which, given

enough data points, leaks the cost function [24]. It is worth noting that an adversary can obtain a stakeholder's secrets by eavesdropping on all the stakeholder's communication lines and applying similar analysis techniques.

It is possible to lessen these vulnerabilities to the above attacks by implementing either more stakeholders or cloud servers while increasing the degree of the polynomial. With additional stakeholders, the adversary would have to infect more stakeholders in order to gain the same information. The same would be the case when introducing several cloud servers for which the stakeholders communicate with, i.e. similar to the principle illustrated in Figure 6.1c, however, with more clouds.

From the above, it can be noted that some information can be gained, should an adversary choose to perform a passive or eavesdropping attack. However, the amount of information leaked is significantly less than that of the global controller or the distributed controller without SMPC. Due to the cost function and constraints being distributed, an adversary would have to perform passive attacks on all but one stakeholder to gain the same information that a passive attack on the global controller would achieve. Therefore, the controller using consensus ADMM and SMPC improves privacy-preservation.

8.2.2 Active Attacks

In active attacks, the adversary makes an infected component deviate from protocol. For the attack points presented in Figure 8.1, it is assumed that an adversary can both see and manipulate data. The investigation will focus solely on possibilities for protection, detection, and correction that can be implemented using the methods already described in this project. Therefore, methods may exist to counter these active attacks that are not presented in this section. The following lists the potential intentions of the adversary, attack methods, and possible detection and protection methods.

Physical Damage

Here, the adversary intends to cause physical damage. For instance, this could be causing overflow or underflow in the tower, where the latter could lead to depriving the consumption group of water.

Using attack point (a) on a pump station, the adversary changes the cost function of the infected pump station, such that it drives the volume of water in the tower towards the maximum or minimum. At this point, the adversary makes the pump station deviate from its actuation command, resulting in overflow or underflow of the tower. Remark that by changing the cost function, it is possible to have other pump stations contributing to this. This is hard to detect prior to the underflow or overflow since the cost function is private.

Using attack point (b) an adversary can manipulate the data, however, unknowing of what they are manipulating. Thus, manipulating the data with the intention of causing physical damage is not possible. However, it is possible to cause denial of service.

Manipulating Solutions

Here, the adversary attempts to increase the Water Distribution Network (WDN)'s electricity bill.

Using attack point (a), the adversary modifies the cost function and constraints to increase the electricity bill. Once again, such attacks would be difficult to detect due to the privacy of the cost functions and constraints.

As for attacks causing physical damage, an adversary using attack point (b) would not be able to know what data to manipulate to affect the electricity bill.

Denial of Service

Here, an adversary intends to disrupt the optimisation problem algorithm.

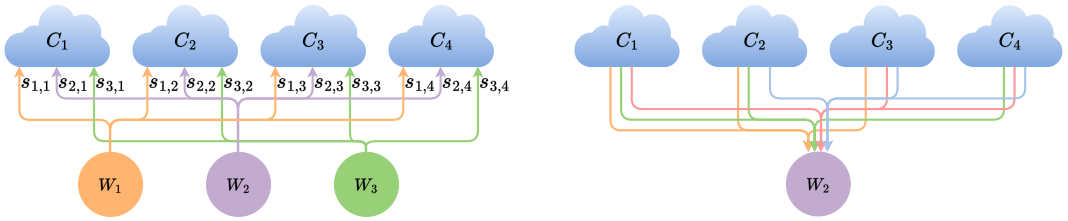
Method 1: Using attack points (a) or (b), the adversary causes a stakeholder to cease communication or disrupts the communication line. This method is easy to detect; however, hard to distinguish from other faults, such as network errors.

Method 2: Using attack point (a), the adversary applies constraints to the stakeholder's optimisation problem, such that finding a solution with consensus ADMM becomes infeasible. This is easy to detect, as the primal residual would not decrease. However, it would prove difficult to distinguish from altered constraints caused by other factors, such as maintenance, as the constraints are private.

Method 3: Using attack point (a) or (b), the adversary adds noise to the secret or shares, such that the optimisation problem does not converge. If attack point (a) is used, detection would be difficult, as described in method 2. However, if attack point (b) is used, it is possible to detect. This can be achieved by adding at least one cloud server while the degree of the polynomial created using Algorithm 1 in Section 6.2 remains unchanged. Alternatively, a setup where all shares are distributed to four or more cloud servers can be used. The latter, which is illustrated in Figure 8.2a, will be used to describe the principle, as it is simpler to illustrate and convey.

Since the degree of the polynomial remains unchanged, only three sum of shares (see step 3 in Algorithm 1) are required to compute the sum. As each stakeholder will be in possession of four different sum of shares, they can compute the sum using different recombination vectors, i.e using $C = \{1, 2, 3\}$ or $C = \{2, 3, 4\}$. This principle is illustrated in Figure 8.2b for a single stakeholder. If the sums found from the recombination vectors are not equal, either a cloud or a communication line must be under attack.

If a fifth cloud server were introduced, determining which communication line is under attack would be possible. By having five cloud servers, the recombination vector can be constructed in 10 different ways. Each sum using the infected communication line would provide wrong yet different results, while all others would be equal. Thereby, the only cloud not contributing to the winning vote must be under attack.



(a) Distribution of shares from stakeholders to clouds.

(b) Distribution of sum of shares from clouds to stakeholders.

Figure 8.2. Using SMPC to detect attacks. The coloured lines in Figure 8.2b indicate the different sets of C a stakeholder can use for the recombination vector.

From the above, it can be concluded that while the privacy-preserving control algorithm does not protect against all attacks, it does achieve a great level of protection. Passive and eavesdropping attacks on the privacy-preserving controller leak significantly less information than such attacks on the global controller. Additionally, active attacks aimed at attack point (b) are, in several cases, difficult, as the adversary has no knowledge of what information they are manipulating. Should an active adversary at attack point (b) choose to manipulate the shares or sum of shares, the above presents a method for detecting and correcting such attacks by utilising cloud servers.

Conclusion 9

The tendency of internet connectivity in critical infrastructure, combined with modern society's heavy reliance on said infrastructure, has resulted in a significant cyber security threat. This report presents a method for implementing distributed optimising control algorithms for critical infrastructure, which keeps local cost functions and constraints private in case of eavesdropping and passive cyber attacks.

The report's use case is a Water Distribution Network (WDN), with two pump stations, a consumption group, and an elevated water tower. The control algorithm was tested using the Smart Water Infrastructures Laboratory (SWIL) at Aalborg University (AAU), emulating the WDN. Low-level PI controllers were developed such that a model for the WDN could be derived based on flows and volumes, abstracting from pump, valve and pipe dynamics.

The electricity bill of the WDN is minimised through the use of Model Predictive Control (MPC). Despite a non-convex cost function, the optimising controller achieved 18.3 % decrease in the electricity bill, compared to an ON/OFF controller. The control problem was solved distributed using consensus Alternating Direction Method of Multipliers (ADMM) and made privacy-preserving using Secure Multi-Party Computation (SMPC), specifically Shamir's Secret Sharing Scheme. The performance of the privacy-preserving controller is compared to that of a global controller to prove that privacy-preservation can be achieved without performance loss.

The privacy-preserving controller has been shown to have the same operation and electricity bill as the global controller in simulation and implementation. Remark that this is the best possible outcome, as neither distributing the controller nor making it privacy-preserving increases the electricity bill.

The information which can be gained through passive or eavesdropping attacks has been significantly reduced for the distributed privacy-preserving controller compared to the global controller. It has been described how this reduction could be further improved through the use of a sharing ADMM algorithm, which was not possible to apply to the cost function in this project.

Furthermore, it has been illustrated that the privacy-preserving controller is able to protect against some instances of active attacks on the communication lines between stakeholders. By introducing cloud servers, the privacy-preserving controller can be modified to detect active attacks on communication lines and determine which communication line is under attack. It is then possible to exclude the infected communication line, while still maintaining normal operation.

In conclusion, it has been shown how distributed privacy-preserving controllers can be implemented without performance loss. Using this state-of-the-art method, critical infrastructure can be protected against passive, eavesdropping, and specific active attacks while communicating over open networks to optimise its operation.

Bibliography

[1] K. Tjell, *Distributed control in water distribution network*, Non published slides.

[2] C. S. Kallesøe, J. B. Deleuran, and K. M. Balla, "Safe and robust cloud-based predictive control for water distribution networks," *IFAC-PapersOnLine*, vol. 56, no. 2, pp. 749–754, 2023, 22nd IFAC World Congress, ISSN: 2405-8963. DOI: https://doi.org/10.1016/j.ifacol.2023.10.1656. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2405896323020657.

[3] M. Barrett, "Framework for improving critical infrastructure cybersecurity version 1.1," en, National Institute of Standards and Technology, Tech. Rep., Mar. 2018. DOI: https://doi.org/10.6028/NIST.CSWP.04162018.

[4] N. A. M. D. Mar, "The nis2 directive: A high common level of cybersecurity in the eu," European Parliment, Tech. Rep., Feb. 2023, https://www.europarl.europa.eu/RegData/etudes/BRIE/2021/689333/EPRS_BRI(2021)689333_EN.pdf, (Accessed on 16/02/2024).

[5] K. Tjell, "Privacy in optimization algorithms based on secure multiparty computation," English, PhD supervisor: Prof. Rafael Wisniewski, Aalborg University, Ph.D. dissertation, 2021. DOI: 10.54337/aau466211893.

[6] Q. Li, R. Heusdens, and M. G. Christensen, "Privacy-preserving distributed optimization via subspace perturbation: A general framework," *IEEE Transactions on Signal Processing*, vol. 68, pp. 5983–5996, 2020. DOI: 10.1109/TSP.2020.3029887.

[7] A. Løvemærke, "Secure control in the cloud using multiparty computation," https://kbdk-aub.primo.exlibrisgroup.com/permalink/45KBDK_AUB/a7me0f/alma9921565577605762, M.S. thesis, Aalborg University, 2019.

[8] J. Val Ledesma, R. Wisniewski, and C. S. Kallesøe, "Smart water infrastructures laboratory: Reconfigurable test-beds for research in water infrastructures management," *Water*, vol. 13, no. 13, 2021, ISSN: 2073-4441. DOI: 10.3390/w13131875. [Online]. Available: https://www.mdpi.com/2073-4441/13/13/1875.

[9] *Entso-e transparency platform*, https://transparency.entsoe.eu/dashboard/show?fbclid=IwAR0OPA5aZiCUBFyqSkh_lYuY8VdHnUDOp1Rjr4EG0MdxKHJkmCNBmGclyhY, (Accessed on 26/02/2024).

[10] N1, *Priser og vilkår*, https://n1.dk/priser-og-vilkaar, (Accessed on 12/09/2023).

[11] J. Maciejowski, *Predictive Control: With Constraints*. Prentice Hall, 2002, ISBN: 9780201398236. [Online]. Available: https://books.google.dk/books?id=HV_Y58c7KiwC.

[12] Nordpool, *Day-ahead market*, https://www.nordpoolgroup.com/en/the-power-market/Day-ahead-market/, (Accessed on 19/04/2024).

[13] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004. DOI: https://doi.org/10.1017/CBO9780511804441.

[14] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2011, ISSN: 1935-8237. DOI: 10.1561/2200000016. [Online]. Available: http://dx.doi.org/10.1561/2200000016.

[15] C. Chen, B. He, Y. Ye, and X. Yuan, "The direct extension of admm for multi-block convex minimization problems is not necessarily convergent," *Mathematical Programming*, vol. 155, pp. 57–79, 2014. [Online]. Available: https://doi.org/10.1007/s10107-014-0826-5.

[16] K. Tjell and R. Wisniewski, "Privacy preservation in distributed optimization via dual decomposition and admm," English, in *2019 IEEE 58th Conference on Decision and Control (CDC)*, ser. IEEE Conference on Decision and Control. Proceedings, 2019 IEEE 58th Conference on Decision and Control (CDC), CDC ; Conference date: 11-12-2019 Through 13-12-2019, United States: IEEE, Mar. 2020, pp. 7203–7208, ISBN: 978-1-7281-1399-9. DOI: 10.1109/CDC40024.2019.9028969.

[17] B. He, L. Hou, and X. Yuan, "On full jacobian decomposition of the augmented lagrangian method for separable convex programming," *SIAM Journal on Optimization*, vol. 25, no. 4, pp. 2274–2312, 2015. DOI: 10.1137/130922793. eprint: https://doi.org/10.1137/130922793. [Online]. Available: https://doi.org/10.1137/130922793.

[18] E. Ghadimi, A. Teixeira, I. Shames, and M. Johansson, "Optimal parameter selection for the alternating direction method of multipliers (admm): Quadratic problems," *IEEE Transactions on Automatic Control*, vol. 60, no. 3, pp. 644–658, 2015. DOI: 10.1109/TAC.2014.2354892.

[19] R. Cramer, I. Damgaard, and J. B. Nielsen, *Secure multiparty computation and secret sharing*, eng. New York, NY: Cambridge University Press, 2015, ISBN: 9781107337756. DOI: https://doi.org/10.1017/CBO9781107337756.

[20] K. S. Tjell, "Privacy preserving control using multiparty computation," https://kbdk-aub.primo.exlibrisgroup.com/permalink/45KBDK_AUB/n4l1aj/alma9921566335905762, M.S. thesis, Aalborg University, 2018.

[21] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, no. 11, pp. 612–613, Nov. 1979, ISSN: 0001-0782. DOI: 10.1145/359168.359176. [Online]. Available: https://doi.org/10.1145/359168.359176.

[22] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi – A software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, vol. 11, no. 1, pp. 1–36, 2019. DOI: 10.1007/s12532-018-0139-4.

[23] P. Virtanen, R. Gommers, T. E. Oliphant, *et al.*, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020. DOI: 10.1038/s41592-019-0686-2.

[24] Y. Ye, H. Chen, M. Xiao, M. Skoglund, and H. Vincent Poor, "Privacy-preserving incremental admm for decentralized consensus optimization," *IEEE Transactions on Signal Processing*, vol. 68, pp. 5842–5854, 2020. DOI: 10.1109/TSP.2020.3027917.

Developed code A

1. Check for convexity of the cost function:

   https://github.com/ES10-1024/P10_Handin/tree/main/Check%20for%20convexity

2. Global controller:

   https://github.com/ES10-1024/P10_Handin/tree/main/Global%20controller/Simple%20Simulink%20implemtation

3. Code used for simulated results:

   https://github.com/ES10-1024/P10_Handin/tree/main/Simulated_results

4. Shamir's Secret Sharing Scheme (Shamir's Scheme):

   https://github.com/ES10-1024/P10_Handin/blob/main/Labratory_implementation/SMPC.py

5. Laboratory implementation:

   https://github.com/ES10-1024/P10_Handin/tree/main/Labratory_implementation

6. Animation of global and privacy-preserving controller:

   https://github.com/ES10-1024/P10_Handin/tree/main/Animations_implemented_controller

Piping and Instrumentation Diagrams

B

Figure B.2 shows a simplified piping and instrumentation diagram of the lab setup. The primary simplification is that components in parallel are drawn as a single component, while some sensors and actuators are omitted. Figures B.3 to B.7 show piping and instrumentation diagrams of the modules, where green lines and green letters indicate pipes outside the modules or nodes connected by pipes. Grey components are not used. All legends are shown in Figure B.1. The pipe resistance $r_{f\Sigma}$ in Figure B.2 is not found in the piping and instrumentation diagrams of the modules since it is a result of the pipe dimensions on the tower module, which is not meant to result in significant pipe resistance.

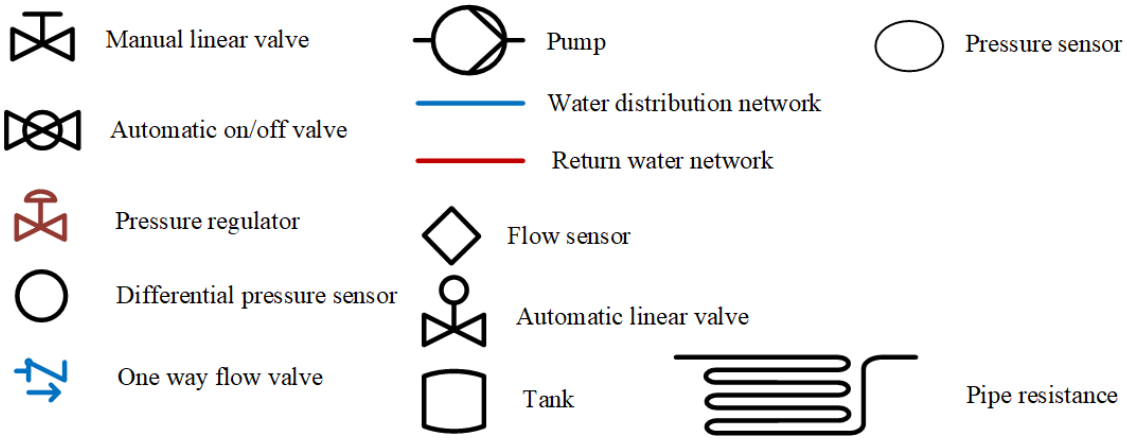


Figure B.1. Legend for the piping and instrumentation diagrams.

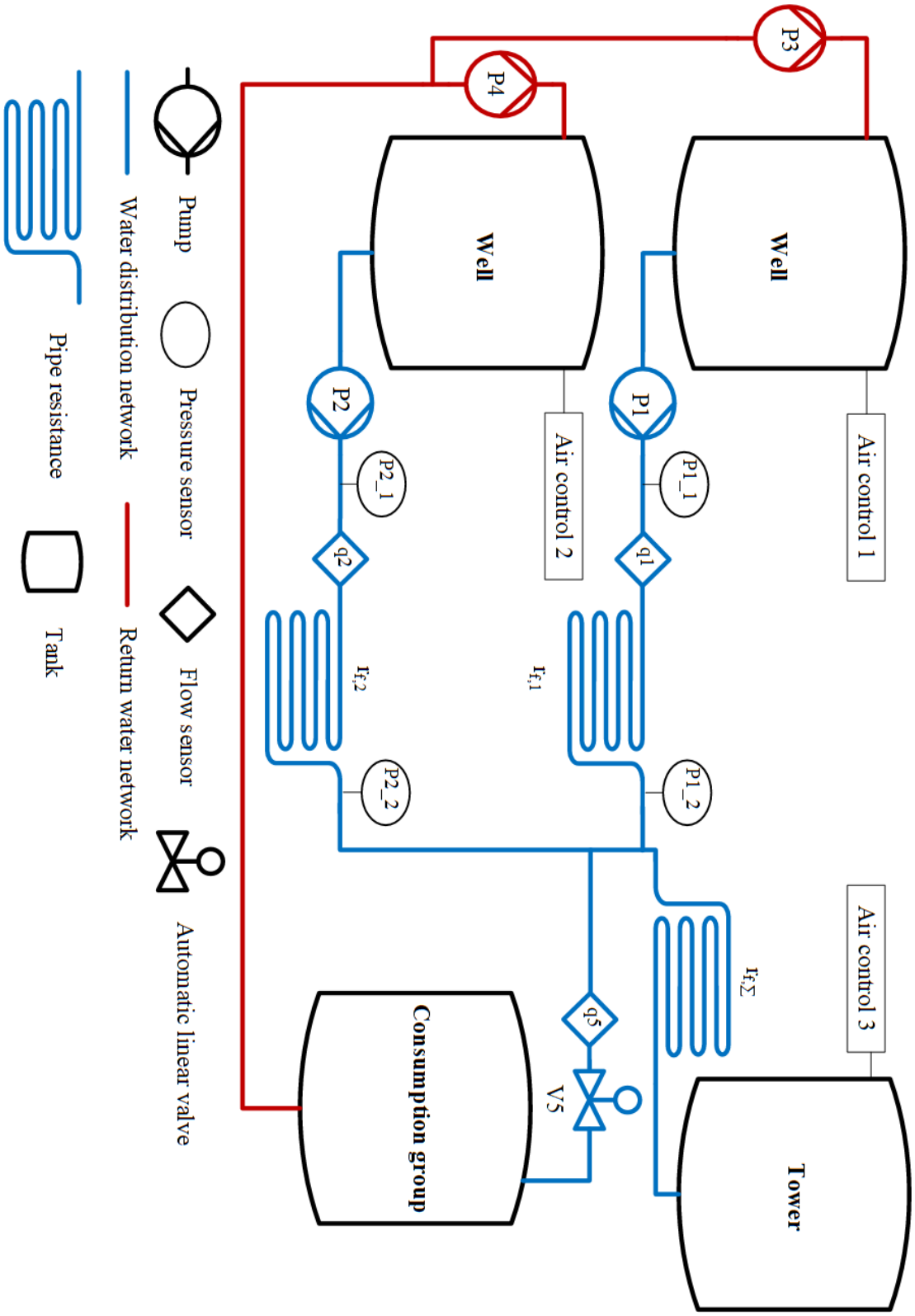


Figure B.2. Simplified piping and instrumentation diagram of the laboratory setup used.

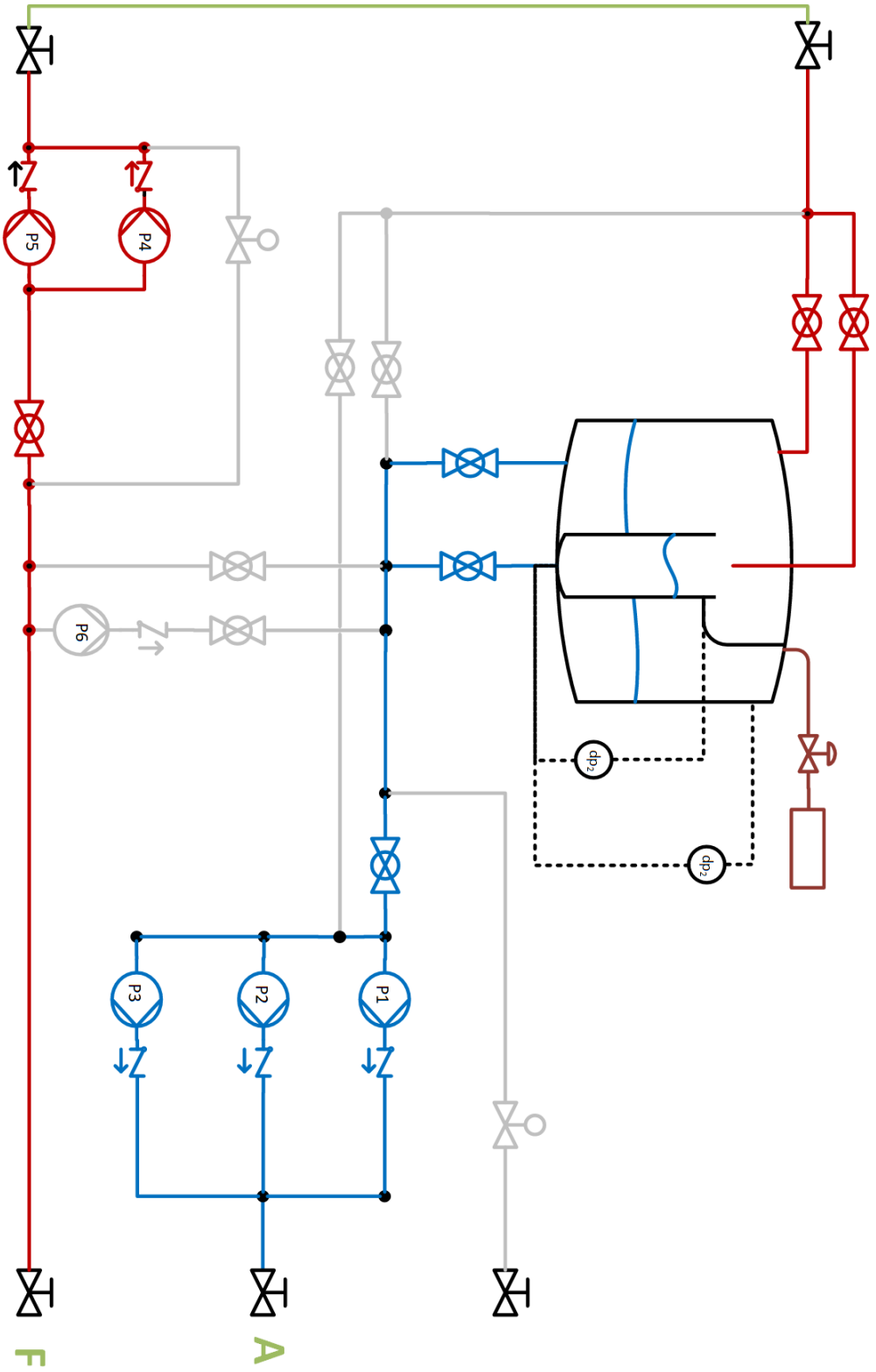


Figure B.3. Piping and instrumentation diagram for pump station 1.

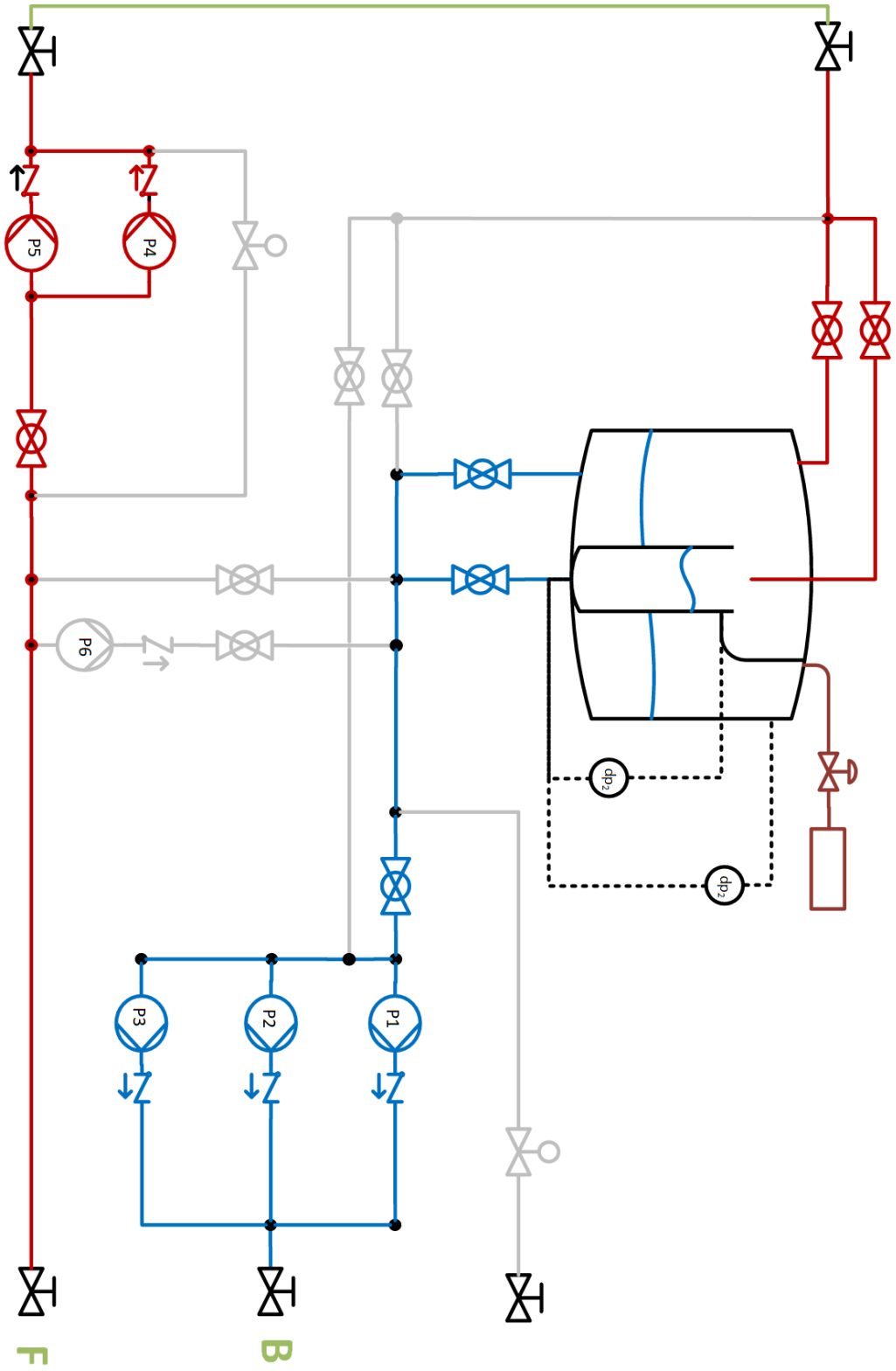


Figure B.4. Piping and instrumentation diagram for pump station 2.

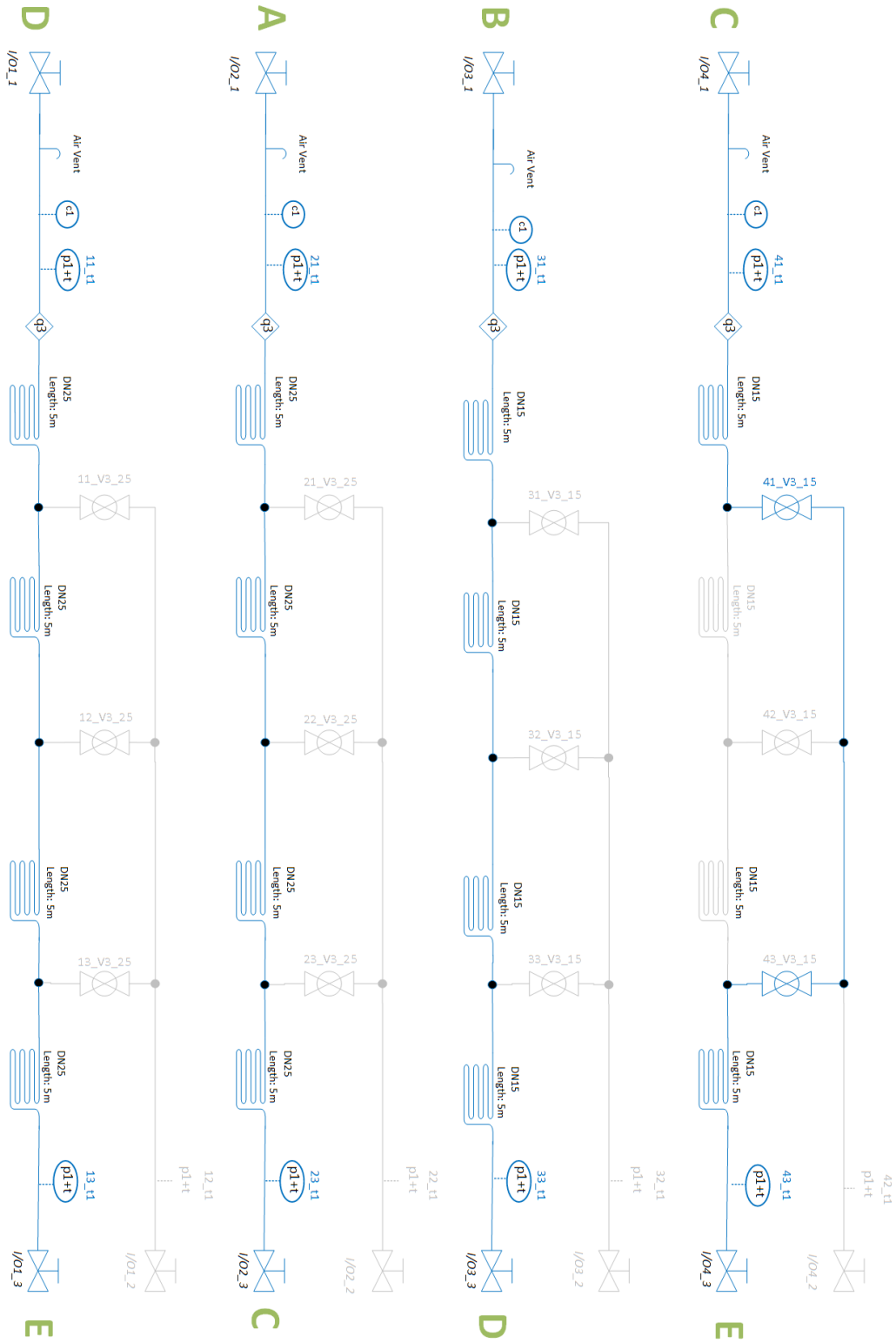


Figure B.5. Piping and instrumentation diagram for the pipe module.

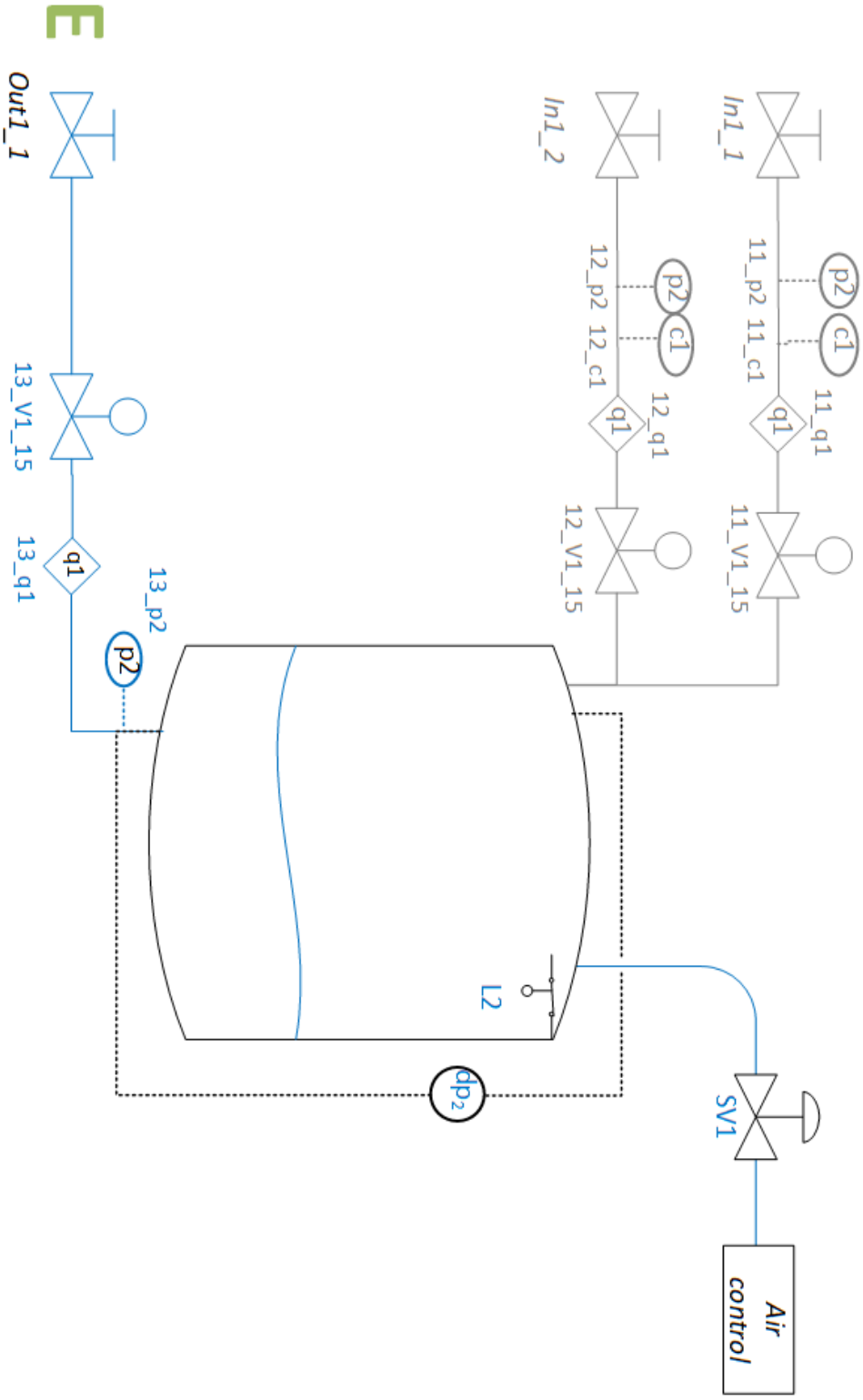


Figure B.6. Piping and instrumentation diagram for the tower module.

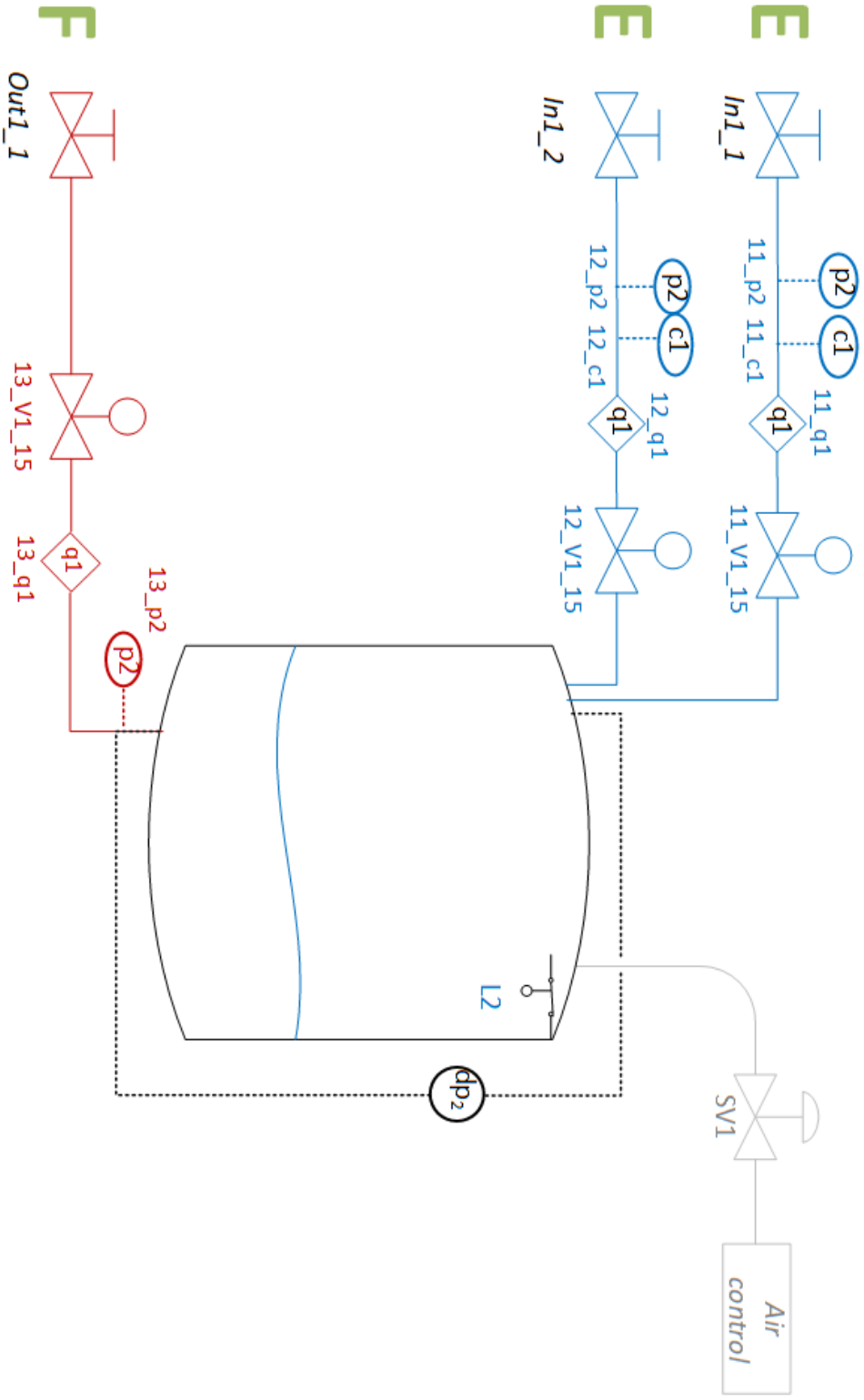


Figure B.7. Piping and instrumentation diagram for the consumer module.

Code Examples C

In this appendix, it is desired to provide code a example of how it is possible to communicate with each module using Modbus over TCP. Furthermore, a code example is given of how it is possible to communicate between the modules using TCP.

C.1 Communication With Modules Using Modbus Over TCP

All sensor data and actuator settings on the modules can be read by a Python script using the pyModbusTCP library as shown in Code snippet C.1. Equally, all actuator settings can be changed as shown for a single pump station in Code snippet C.2. The method works on the PC in lab and the Raspberry Pis on the modules. A firewall blocks other computers. Remark that multiple scripts can read and write to the same register. The latest write command is applied. Which registers the different sensors and actuators belong to can be found in Simulink in the Modbus write block.

```
1  from pyModbusTCP.client import ModbusClient
2
3  ip_addr = '192.168.100.20' #Module IP address
4
5  MB = ModbusClient(host=ip_addr, port=502, auto_open=True)
6  #Set up connection to module, 502 is standard Modbus port
7
8  if MB.open(): #Check if connection has been correctly set up
9      print("Modbus open")
10 else:
11     print("Modbus not open") #Some bug, e.g.  ethernet not connected
12     quit()  #End script
13
14 data = MB.read_input_registers(0,25) #Read all registers on pump station
15 print(data)
```

Code snippet C.1. Read registers over Modbus TCP.

```
1  MB.write_single_register(6,50*100)
2  #Setting register value to 50, multiplication is required per the PLC code
```

Code snippet C.2. Write registers over Modbus TCP.

C.2 Communication Between Modules Using TCP

TCP communication is set up as shown in Code snippet C.3 and Code snippet C.4. The server needs to be started before the client. In the Code snippets, the IP address is set to `127.0.0.1`, which is an internal IP on a Windows PC. Thereby, the example can run on two different terminals on one PC. To run the example in the laboratory on the Raspberry Pis, use the local IP `192.168.100.module_number`.

Line 9 in Code snippet C.3 specifies that the port should be stolen from another program if the port is already in use. This is practical, as programs usually do not close down connections if they crash, but it is generally considered an unsafe method.

In the implementation, the water tower is the server for both pump stations while pump station 1 is the server for pump station 2.

```
1  #Server
2  import socket
3  import time
4
5  HOST = "127.0.0.1"    #Own IP
6  PORT = 6644           #Port to listen on (non-privileged ports are > 1023)
7
8  s= socket.socket(socket.AF_INET, socket.SOCK_STREAM)      #Setup TCP socket
9  s.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)   #If port is in use, steal the port
10 s.bind((HOST, PORT))         #Bind to port
11 s.listen()                   #Start listening for incoming connections
12 conn, addr = s.accept()      #Accept connection
13 print("Connected by", addr)
14
15 while True:
16     data = conn.recv(1024) #Receive up to the specified amount of bytes
17     print(data)
18     conn.sendall(data)     #Send the message back
```

Code snippet C.3. TCP server example.

```
1  #Client
2  import socket
3  import time
4  import numpy as np
5
6  HOST = "127.0.0.1"    #Servers IP
7  PORT = 6644           #Servers port
8
9  s=socket.socket(socket.AF_INET, socket.SOCK_STREAM)       #Setup TCP socket
10 s.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)   #If port is in use, steal the port
11 s.connect((HOST, PORT))  #Connect to server
12
13 while True:
14     a= np.array([5,7,9])     #Data to send
15     s.sendall(a.tobytes())   #Pack data to bytes
16     print("Message sent")
17     retuned_data = s.recv(1024) #Recive up to the specified amount of bytes
18     print(retuned_data)
19     returned_message = np.frombuffer(retuned_data, dtype=a.dtype) #Unpack data
20     print("Received",returned_message)
21     time.sleep(60)
```

Code snippet C.4. TCP client example.

Pump Station Flow Controller

D

This appendix describes the design of the low-level controllers, regulating the flow from the pump stations to the tower. The input to the pumps goes from 0 to 100 %, while the flow can be measured using sensors on the modules. To achieve reference following, a Proportional Integral (PI) controller, shown in (D.1), is chosen, as this is generally utilised in real Water Distribution Network (WDN).

$$C(s) = K_{\mathrm{P}} + \frac{K_{\mathrm{I}}}{s} \tag{D.1}$$

$C(s)$	Pump station controller	$[\cdot]$
s	Laplace domain variable	$[\cdot]$
K_{P}	Proportional gain	$[\cdot]$
K_{I}	Integral gain	$[\cdot]$

The controller is implemented in Python. The output is limited to the range of 0 to 100, and anti-windup is implemented. During testing, it was found that the pumps required a frequency of approximately 40-50 % to produce enough pressure for the water to flow. Therefore, the integral for each pump station is initiated as a value slightly below.

Two different flow sensors are available to measure the flow: one located at the pump station and another at the input to the pipe module. Through testing, it was discovered that the flow sensor at the pump station had poor measurements at low flows; therefore, the sensor on the pipe module was used.

It was found that the controllers for the two pump stations must be tuned individually. The values for each controller are found through hand-tuning until a rise time within a minute and low flow oscillations are presented. The constants for the two controllers can be seen in Table D.1.

	K_{P}	K_{I}
Pump station 1	6.5	3.25
Pump station 2	16	2

Table D.1. Controller constants for the two pump stations.

The performance of the controller for pump station 1 can be seen in Figure D.1, while the performance for pump station 2 can be seen in Figure D.2. From the figures, it can be seen how the flow for both pump stations oscillate. However, it can also be observed how the changes in actuation at these oscillations are rather small, meaning that the flows are sensitive to small changes in their actuation.

It was desired to investigate whether these oscillations may be caused by a cross-coupling between the two pump stations, which may be present as the pump stations are pumping water into the same entrance of the tower. Figure D.3 shows a cross-coupling is present, as it can be seen how the flows oscillate with 180° phase difference. Nevertheless, while these oscillations are not ideal, the mean flow of both controllers follows their given reference. Therefore, the performance is deemed acceptable.

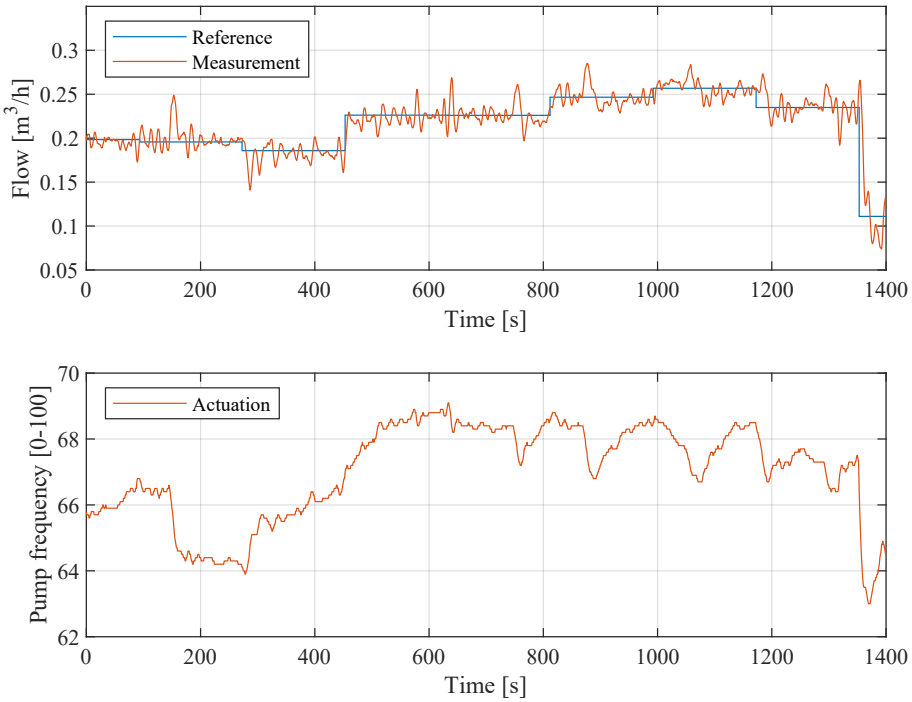


Figure D.1. Performance of the PI controller for pump station 1 with the controller values presented in Table D.1.

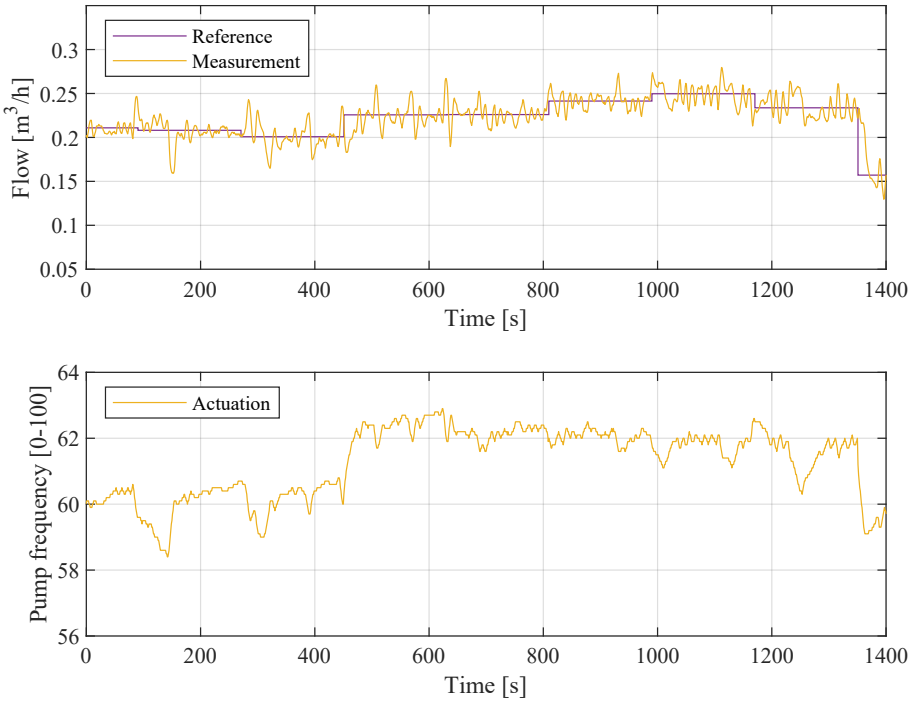


Figure D.2. Performance of the PI controller for pump station 2 with the controller values presented in Table D.1.

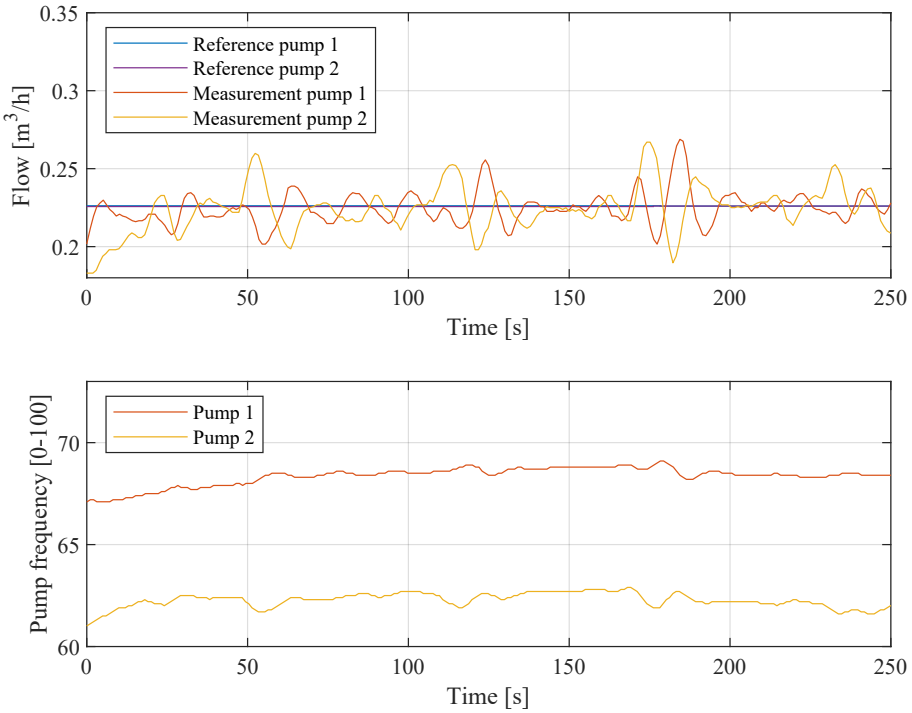


Figure D.3. Cross-coupling between the two pump stations

Valve Controller E

In this appendix, it is desired to design a controller for the valve in the consumer module such that a desired consumption is obtained. This is achieved using a Proportional Integral (PI) controller:

$$C(s) = K_{\mathrm{P}} + \frac{K_{\mathrm{I}}}{s} \tag{E.1}$$

s	Laplace domain variable	$[\cdot]$
$C(s)$	Valve controller	$[\cdot]$
K_{P}	Proportional gain	$[\cdot]$
K_{I}	Integral gain	$[\cdot]$

The consumer module has two flow sensors in parallel, and two valves, with opening degrees 0-100, where 100 is fully open. At first, an individual controller for a single valve is designed, followed by the design of a combined controller, setting the reference for each individual controller. The controllers are implemented in Python with anti-windup applied.

The individual controllers have been hand-tuned to have low oscillations and a rise time within a couple of minutes. The performance of the controller for a single valve with $K_{\mathrm{P}} = 40$ and $K_{\mathrm{I}} = 20$ over several reference changes is shown in Figure E.1. The graph shows that the flow, in some instances, leaves steady state without change in reference. This is caused by changes in pump station flow, in order to avoid overfilling the tower. The figure shows a rise time of approximately 30 s. An upper limit for flow was observed to be around $0.3\,\mathrm{m^3\,h^{-1}}$.

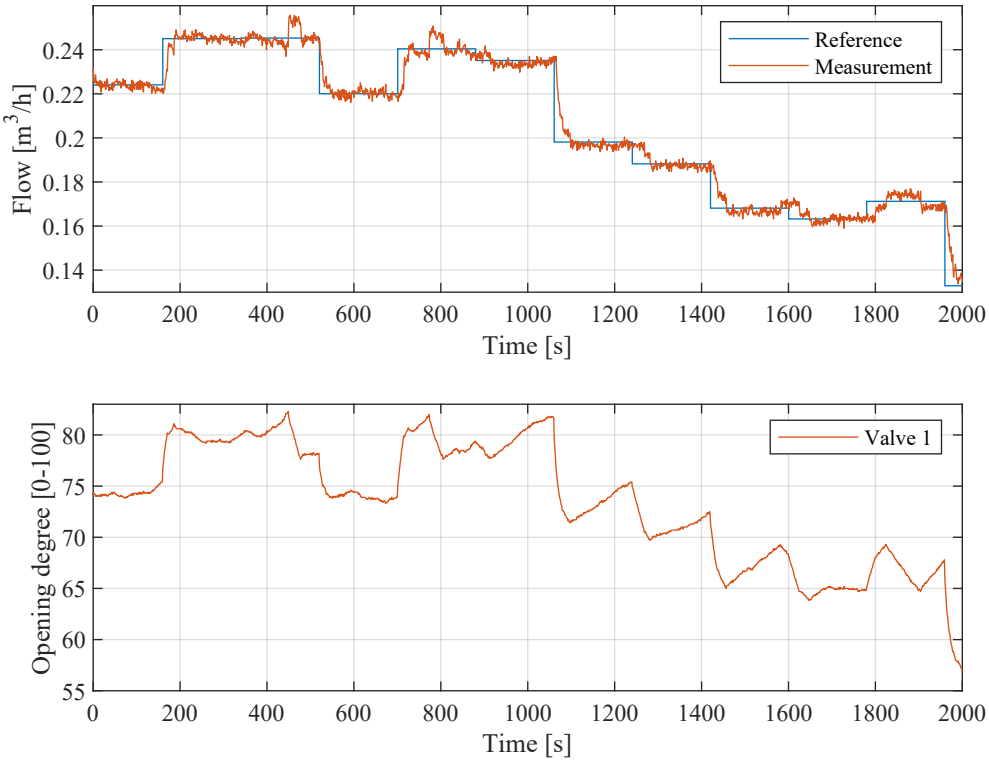


Figure E.1. Test of the controller for a single valve.

The second valve is used to allow for larger flows. Through testing, it was found that the controllers performed poorly at low flows. For this reason, it is chosen to use the second valve at high flows. The second valve has its own controller and sensor, but utilises the same controller values. The reference for the controllers is set as shown in Figure E.2.

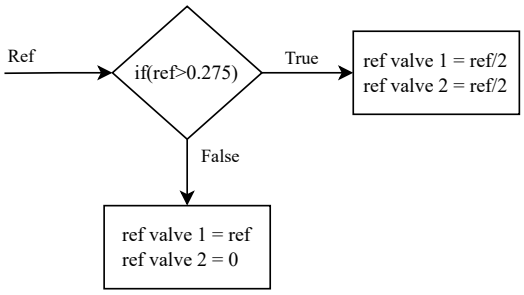


Figure E.2. Flowchart describing how the reference is set for the two valve controllers.

The combined controller has been tested for several consumption references, resulting in the flows shown in Figure E.3. The figure shows great reference tracking and a rise time of approximately 30 s.

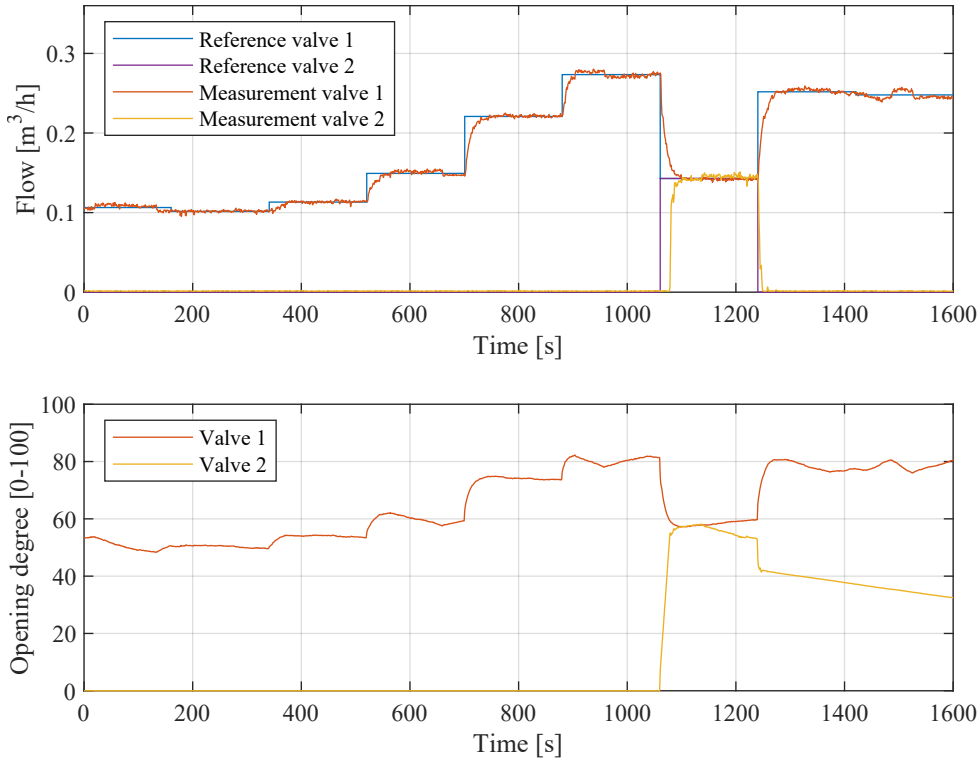


Figure E.3. Test of controller for the two consumer valves.

Estimation of Pipe Resistance

F

In this appendix, it is desired to estimate the resistance of the pipes shown in Figure 2.1.

The flow through the pipes and the pressure drop over the pipes were measured at different flows. During measurements, the flow has been changed from the tower to the consumer to avoid overflow of the tower. The tower has been pressurised to the elevation indicated in Section 3.4.

F.1 Estimation of $r_{f,1}$ and $r_{f,2}$

Figure F.1 shows measurements and chosen data points for estimation of the pipe resistance $r_{f,1}$. A similar method has been used for the estimation of $r_{f,2}$.

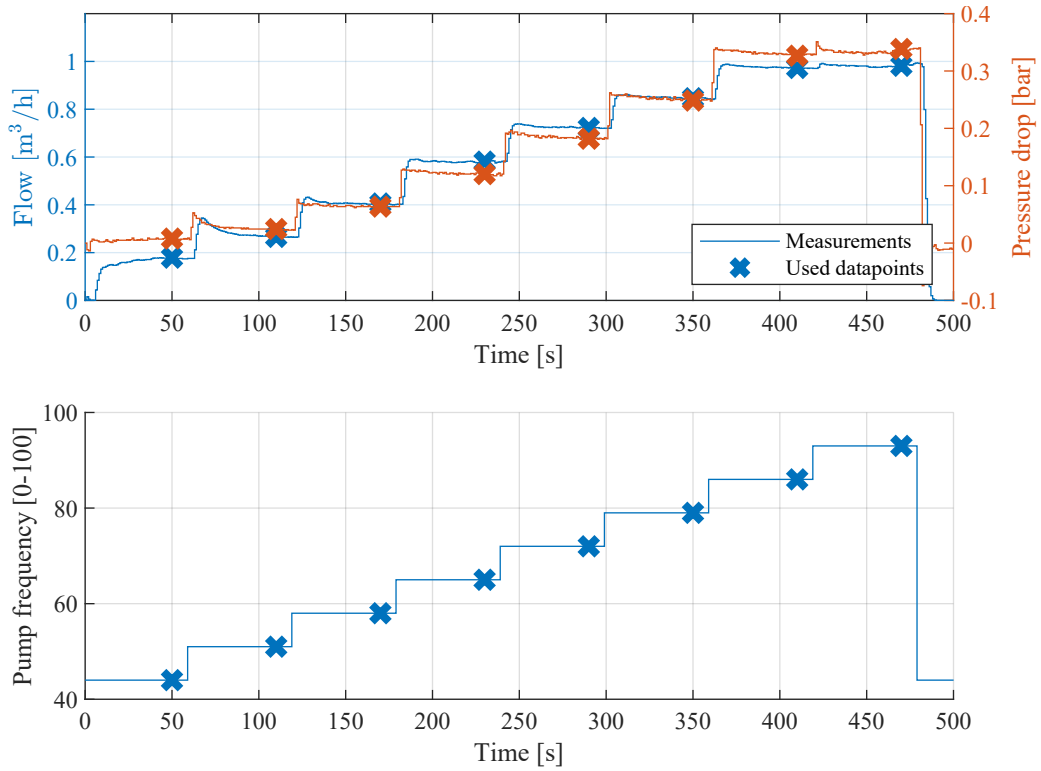


Figure F.1. Measurements of pressure drop over the pipe, flow through the pipe, and pump actuation.

The data points are fitted, using least squares, to the model shown in (F.1), which is an ordinary pipe resistance model plus a constant. The constant is added such that DC offsets in pressure measurements do not influence the estimate of the pipe resistance. Figure F.2 shows the measurements and the fitted model for $r_{f,1}$ and $r_{f,2}$. The figure shows that the model is a great fit. The estimated pipe resistances are shown in Table F.1.

$$\Delta p = r_{f,i} u_i^2 + k \tag{F.1}$$

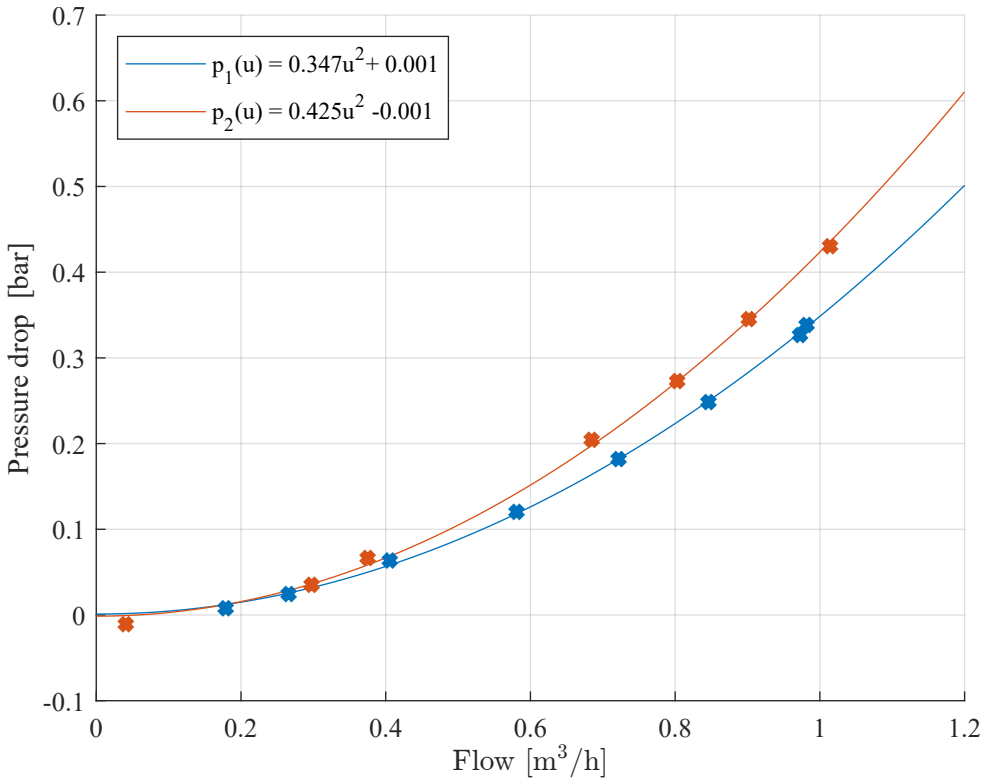


Figure F.2. Measurements of pressure drop and flow, and model for the measurements.

$r_{f,1} = 0.35$ bar/(m^3/h)^2
$r_{f,2} = 0.42$ bar/(m^3/h)^2

Table F.1. Estimated pipe resistances.

F.2 Estimation of $r_{f,\Sigma}$

The same procedure as above has been used, with zero flow into the consumption group. Since there is no pressure sensor at the bottom of the tank, the air pressure emulating elevation is removed, and the pressure is estimated as one atmosphere plus the water column in the tank. The used data is shown in Figure F.3. The fitted model is shown in Figure F.4, resulting in a resistance of $r_{f,\Sigma}$ =0.30 bar/(m^3/h)^2. Thereby, the resistance is significant compared to the resistance in the pipe modules.

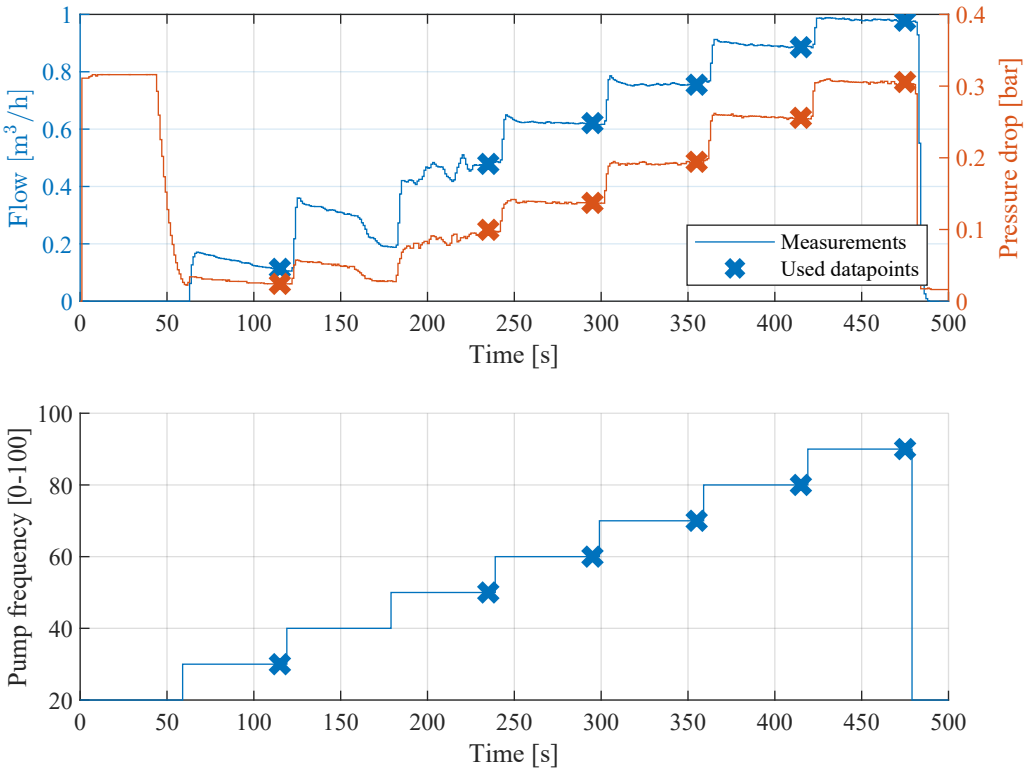


Figure F.3. Measurements of pressure drop over the pipe, flow through the pipe, and pump actuation.

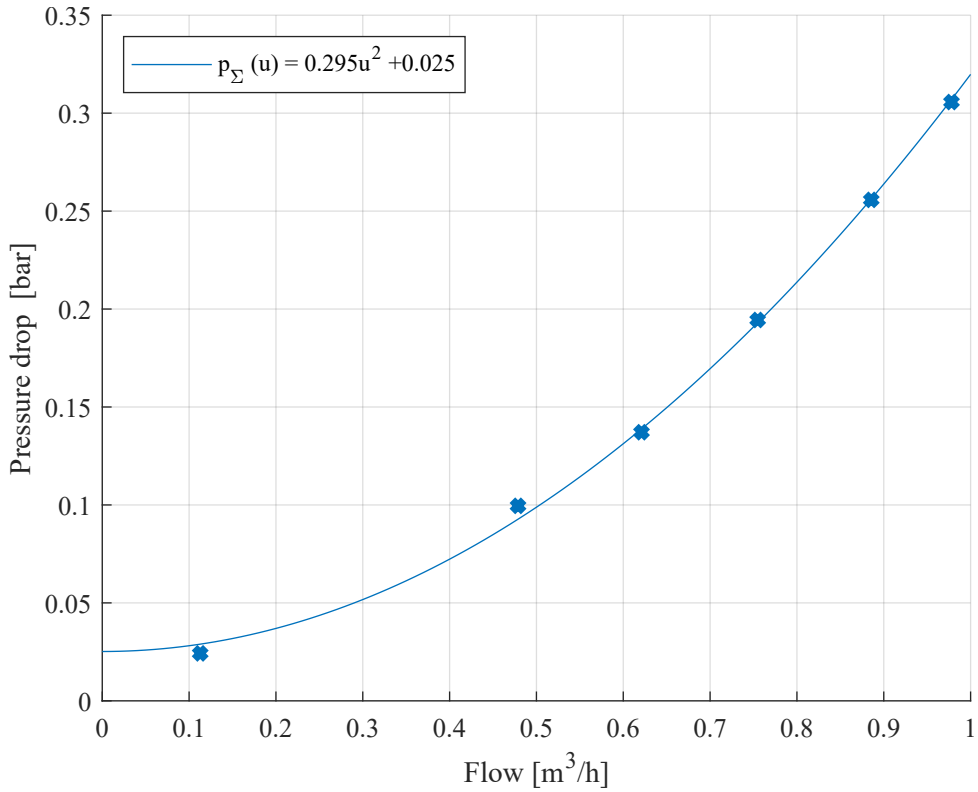


Figure F.4. Measurements of pressure drop and flow, and model for the measurements.

Cost Function and Constraints

G

In this appendix, the cost function and constraints are rewritten in terms of $\boldsymbol{\mathcal{U}}_k$ instead of $q_i(t)$. $\boldsymbol{\mathcal{U}}_k$ is defined as shown in (G.1). For simplicity, it is chosen to rewrite the cost function for $N_\mathrm{q} = 2$ pump stations and $N_\mathrm{d} = 1$ consumption group, matching the numbers in the laboratory setup.

$$\boldsymbol{\mathcal{U}}_k = \begin{bmatrix} \boldsymbol{u}_k \\ \boldsymbol{u}_{k+1} \\ \vdots \\ \boldsymbol{u}_{k+N_\mathrm{c}-1} \end{bmatrix} \in \mathbb{R}^{N_\mathrm{c}N_\mathrm{q}} \qquad \boldsymbol{u}_k = \begin{bmatrix} q_1(t_0 + kt_\mathrm{s}) \\ q_2(t_0 + kt_\mathrm{s}) \\ \vdots \\ q_{N_\mathrm{q}}(t_0 + kt_\mathrm{s}) \end{bmatrix} \in \mathbb{R}^{N_\mathrm{q}} \tag{G.1}$$

where:

t_s	Sampling time
t_0	Initial time
N_c	Control horizon

The flow into the water tower $q_\Sigma(\alpha)$ is defined as shown in (G.2), where α is a time index. This can be written as a vector of the flow for all samples within the control horizon N_c as shown in (G.3), where $\mathbf{d} \in \mathbb{R}^{N_\mathrm{c}}$ and $\mathbf{q}_\Sigma \in \mathbb{R}^{N_\mathrm{c}}$ are the predicted demand and flow into the water tower over the control horizon, respectively.

$$q_\Sigma(\alpha) = \sum_{i=1}^{N_\mathrm{q}} q_i(\alpha) - d(\alpha) \tag{G.2}$$

$$\mathbf{q}_\Sigma = \mathbf{A}_1 \boldsymbol{\mathcal{U}}_k - \mathbf{d} \in \mathbb{R}^{N_\mathrm{c}} \qquad \mathbf{A}_1 = \mathrm{diag}(\mathbb{1}_{N_\mathrm{q}}^\top) \in \mathbb{R}^{N_\mathrm{c} \times N_\mathrm{q}N_\mathrm{c}} \tag{G.3}$$

The water height h_v written as a function of time in (G.4), can be written as a vector $\mathbf{h}_\mathrm{V} \in \mathbb{R}^{N_\mathrm{c}}$ of the water heights within the control horizon as shown in (G.5). Here, $\mathbf{A}_2 \in \mathbb{R}^{N_\mathrm{c} \times N_\mathrm{c}}$ is a lower triangular matrix of ones, or in other words, an integrator. V_0 is the volume of water at the beginning of the control horizon.

$$h_\mathrm{V}(\alpha + 1) = \frac{V(\alpha + 1)}{A_\mathrm{t}} \tag{G.4a}$$

$$V(\alpha + 1) = t_\mathrm{s} q_\Sigma(\alpha) + V(\alpha) \tag{G.4b}$$

$$\mathbf{h}_{\mathrm{V}} = (V_0 + t_{\mathrm{s}}\mathbf{A}_2\mathbf{q}_\Sigma)\frac{1}{A_{\mathrm{t}}} \in \mathbb{R}^{N_{\mathrm{c}}} \tag{G.5}$$

A vector of flows for a single pump station $\mathbf{q}_i$ for the entire control horizon can be written as shown in (G.6).

$$\mathbf{q}_i = \mathbf{A}_{3,i}\boldsymbol{\mathcal{U}}_k \in \mathbb{R}^{N_{\mathrm{c}}} \tag{G.6a}$$

$$\mathbf{v}_1 = \begin{bmatrix} 1 & 0 \end{bmatrix}^\top \in \mathbb{R}^{N_{\mathrm{q}}} \qquad \mathbf{v}_2 = \begin{bmatrix} 0 & 1 \end{bmatrix}^\top \in \mathbb{R}^{N_{\mathrm{q}}} \tag{G.6b}$$

$$\mathbf{A}_{3,i} = \mathrm{diag}(\mathbf{v}_i^\top) \in \mathbb{R}^{N_{\mathrm{c}} \times N_{\mathrm{c}} N_{\mathrm{q}}} \tag{G.6c}$$

Given the above, it is possible to construct a vector of power consumption for the entire control horizon for one pump station. The power consumption for a single hour is shown in (G.7). For simplicity, each term is handled individually in (G.8). Here, $\odot$ denotes element-wise multiplication.

$$\begin{aligned} P_i(\alpha) = \frac{1}{\eta_i}\Big(&\underbrace{r_{f,i}|q_i(\alpha)|q_i^2(\alpha)}_{\text{Pipe resistance}} + \underbrace{r_{\mathrm{f},\Sigma}|q_\Sigma(\alpha)|q_\Sigma(\alpha)q_i(\alpha)}_{\text{Combined pipe resistance}} + \\ &+ \underbrace{\rho_w g_0 h_{\mathrm{V}}(\alpha)q_i(\alpha)}_{\text{Water height}} + \underbrace{\rho_{\mathrm{w}} g_0 h_i q_i(\alpha)\Big)}_{\text{Elevation}} \end{aligned} \tag{G.7}$$

$$\mathbf{P}_i = \text{Pipe resistance}_i + \text{Combined pipe resistance}_i + \text{Water height}_i + \text{Elevation}_i \tag{G.8a}$$

$$\text{Pipe resistance}_i = \frac{1}{\eta_i} r_{\mathrm{f},i} \left(\mathbf{A}_{3,i}\boldsymbol{\mathcal{U}}_k\right)^{\odot 3} \in \mathbb{R}^{N_{\mathrm{c}}} \tag{G.8b}$$

$$\text{Combined pipe resistance}_i = \frac{1}{\eta_i} r_{\mathrm{f},\Sigma}|\mathbf{A}_1\boldsymbol{\mathcal{U}}_k - \mathbf{d}| \odot (\mathbf{A}_1\boldsymbol{\mathcal{U}}_k - \mathbf{d}) \odot (\mathbf{A}_{3,i}\boldsymbol{\mathcal{U}}_k) \in \mathbb{R}^{N_{\mathrm{c}}} \tag{G.8c}$$

$$\text{Water height}_i = \frac{1}{\eta_i} \rho_{\mathrm{w}} g_0 \left(V_0 + t_{\mathrm{s}}\mathbf{A}_2 \left(\mathbf{A}_1\boldsymbol{\mathcal{U}}_k - \mathbf{d}\right)\right) \odot (\mathbf{A}_{3,i}\boldsymbol{\mathcal{U}}_k) \in \mathbb{R}^{N_{\mathrm{c}}} \tag{G.8d}$$

$$\text{Elevation}_i = \frac{1}{\eta_i} \rho_{\mathrm{w}} g_0 h_i \mathbf{A}_{3,i}\boldsymbol{\mathcal{U}}_k \in \mathbb{R}^{N_{\mathrm{c}}} \tag{G.8e}$$

The electricity bill can be written using a sum of the number of pump stations and a sum of the hours within the control horizon as shown in equation (G.9). Alternatively, it can be written as a matrix-vector product and a sum of each pump station's electricity bill as shown in (G.10). Here, $\mathbf{J}_{\mathrm{e}} \in \mathbb{R}^{N_{\mathrm{c}}}$ is a vector of the electricity prices within the control horizon.

$$J_{\mathrm{p}} = \sum_{i=1}^{N_{\mathrm{q}}} \sum_{\alpha=k}^{k+N_{\mathrm{c}}-1} t_{\mathrm{s}} J_{\mathrm{e}}(\alpha) P_i(\alpha) \tag{G.9}$$

$$J_{\mathrm{p}} = \sum_{i=1}^{N_{\mathrm{q}}} t_{\mathrm{s}} \mathbf{J}_{\mathrm{e}}^\top \mathbf{P}_i \in \mathbb{R} \tag{G.10}$$

The term in the cost function punishing different water volumes in the tower at the beginning and end of the control horizon shown in (G.11), can be written as a function of $\boldsymbol{\mathcal{U}}_k$ as shown in (G.12).

$$J_{\mathrm{V}} = \kappa \left(V(k) - V(k+N_{\mathrm{c}})\right)^2 \tag{G.11}$$

$$J_{\mathrm{V}} = \kappa \left(\mathbb{1}_{N_{\mathrm{c}}}^{\top} \left(\mathbf{A}_1 \boldsymbol{\mathcal{U}}_k - \mathbf{d}\right)\right)^2 \in \mathbb{R} \tag{G.12}$$

Thereby, the cost function has been written in terms of $\boldsymbol{\mathcal{U}}_k$.

G.1 Constraints

In this section, the constraints are written in terms of $\boldsymbol{\mathcal{U}}_k$. The maximum flow through each pump station can be written as shown in (G.13). The constraint can be written in terms of $\boldsymbol{\mathcal{U}}_k$ for each pump station individually as shown in (G.14). Here, $\leq_{\mathrm{e}}$ denotes element-wise less than or equal to, and $\overline{q}_i$ is the maximum pump station flow.

$$0 \leq q_i(t) \leq \overline{q}_i \tag{G.13}$$

$$0 \leq_{\mathrm{e}} \mathbf{A}_{3,i} \boldsymbol{\mathcal{U}}_k \leq_{\mathrm{e}} \overline{q}_i \tag{G.14}$$

The constraint ensuring the water volume in the tower is above a minimum and below a maximum can be written as shown in (G.15), and in terms of $\boldsymbol{\mathcal{U}}_k$ as shown in (G.16). Here, $\underline{V}$, and $\overline{V}$ are the minimum and maximum volume of water in the tower, respectively.

$$\underline{V} \leq V(t) \leq \overline{V} \tag{G.15}$$

$$\underline{V} \leq_{\mathrm{e}} V_0 + t_{\mathrm{s}} \mathbf{A}_2 \mathbf{q}_{\Sigma} \leq_{\mathrm{e}} \overline{V} \tag{G.16}$$

The Total Yearly Extraction Limit (TYEL) constraint shown in (G.17), can be written in terms of $\boldsymbol{\mathcal{U}}_k$ as shown in (G.18). Here $\overline{Q}_i$ denotes the TYEL for pump station i.

$$\sum_{\alpha=k}^{k+N_{\mathrm{c}}} q_i(k) t_{\mathrm{s}} \leq \overline{Q}_i \tag{G.17}$$

$$t_{\mathrm{s}} \mathbf{A}_2 \left(\mathbf{A}_{3,i} \boldsymbol{\mathcal{U}}_k\right) \leq_{\mathrm{e}} \overline{Q}_i \tag{G.18}$$

G.2 Scaled Cost Functions

Here, it is desired to write up the scaled cost function. Be aware that it is only the cost function and not the constraints which is scaled. The scaled cost function is shown below; remark that it is only J_p which is scaled. Thereby, requiring a new value of κ for J_V.

$$J_\mathrm{p} = \sum_{i=1}^{N_\mathrm{q}} \mathbf{J}_\mathrm{e}^\top \mathbf{P}_i \in \mathbb{R} \tag{G.19a}$$

$$\mathbf{P}_i = \text{Pipe resistance}_i + \text{Combined pipe resistance}_i + \text{Water height}_i + \text{Elevation}_i \tag{G.19b}$$

$$\text{Pipe resistance}_i = \frac{1}{\eta_i} \frac{r_{\mathrm{f},i}}{10\,000} \left(\mathbf{A}_{3,i} \boldsymbol{\mathcal{U}}_k\right)^{\odot 3} \in \mathbb{R}^{N_\mathrm{c}} \tag{G.19c}$$

$$\text{Combined pipe resistance}_i = \frac{1}{\eta_i} \frac{r_{\mathrm{f},\Sigma}}{10\,000} |\mathbf{A}_1 \boldsymbol{\mathcal{U}}_k - \mathbf{d}| \odot (\mathbf{A}_1 \boldsymbol{\mathcal{U}}_k - \mathbf{d}) \odot (\mathbf{A}_{3,i} \boldsymbol{\mathcal{U}}_k) \in \mathbb{R}^{N_\mathrm{c}} \tag{G.19d}$$

$$\text{Water height}_i = \frac{1}{\eta_i} \frac{\rho_\mathrm{w} g_0}{10\,000} \left(V_0 + \mathbf{t}_\mathrm{s} A_2 \left(\mathbf{A}_1 \boldsymbol{\mathcal{U}}_k - \mathbf{d}\right)\right) \odot \left(\mathbf{A}_{3,i} \boldsymbol{\mathcal{U}}_k\right) \in \mathbb{R}^{N_\mathrm{c}} \tag{G.19e}$$

$$\text{Elevation}_i = \frac{1}{\eta_i} \frac{\rho_\mathrm{w} g_0}{10\,000} h_i \mathbf{A}_{3,i} \boldsymbol{\mathcal{U}}_k \in \mathbb{R}^{N_\mathrm{c}} \tag{G.19f}$$

$$J_\mathrm{V} = \kappa \left(\mathbb{1}_{N_\mathrm{c}}^\top \left(\mathbf{A}_1 \boldsymbol{\mathcal{U}}_k - \mathbf{d}\right)\right)^2 \in \mathbb{R} \tag{G.19g}$$

Stopping Criteria H

This appendix shows extended versions of the figures showing the difference between the different stopping criteria regarding the number of iterations used and performance.

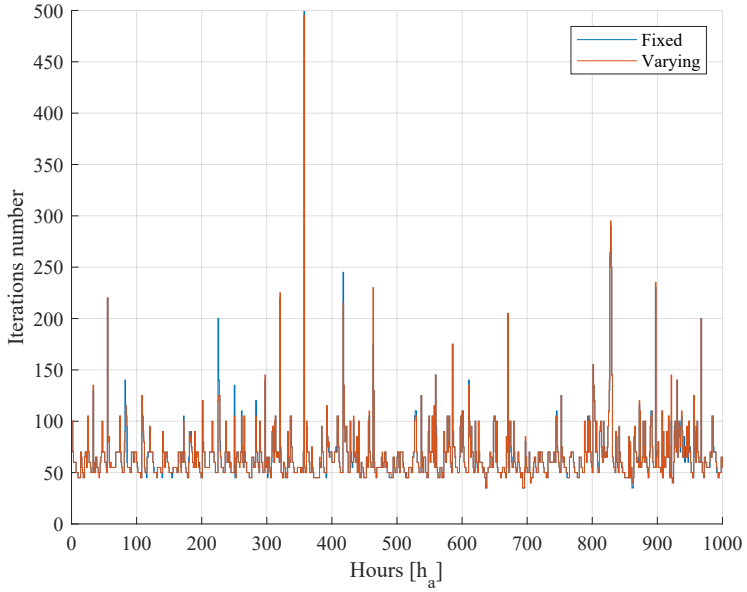


Figure H.1. Number of iterations used with fixed and varying stopping criteria.

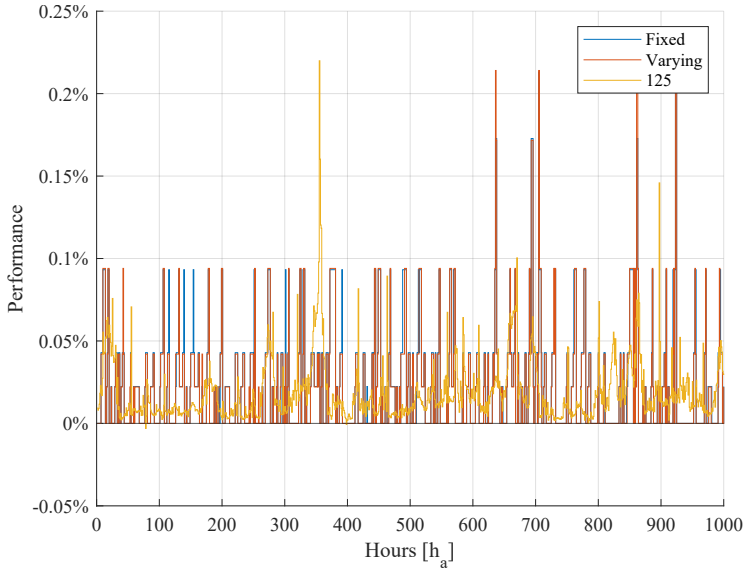


Figure H.2. `Performance` with 125 iterations, fixed, and varying stopping criteria.

Example: Finite Field Division

I

This appendix provides an example of how division is performed within a finite field, as this is required to compute the recombination vector. The example is heavily inspired by [7, p. 58-60].

The equation for the recombination vector is as follows:

$$\delta_i(0) = \prod_{j \in C, j \neq i} \frac{-j}{i-j} \mod \beta \tag{I.1}$$

This example will calculate $\delta_1(0)$ for $C = \{1, 2, 3\}$ and $\beta = 7$.

$$\delta_1(0) = \left(\frac{-2}{1-2} \cdot \frac{-3}{1-3}\right) \mod 7 = \left(\frac{2}{1} \cdot \frac{3}{2}\right) \mod 7 \tag{I.2}$$

The first fraction can easily be computed as:

$$\frac{2}{1} \mod 7 = 2 \mod 7 = 2 \tag{I.3}$$

However, as the second fraction does not yield an integer, it cannot be computed as easily. Instead, the fraction can be rewritten as:

$$\frac{3}{2} \mod 7 = 3 \cdot \frac{1}{2} \mod 7 \tag{I.4}$$

To determine the value of $\frac{1}{2} \mod 7$, the following is used:

$$2 \cdot x \mod 7 = 1 \tag{I.5}$$

Values for which modulo 7 yields 1 are: $1, 8, 15, 22, \ldots$; however, the number must return an integer when divided by 2. Thereby, the solution can be found as: $8/2 \mod 7, 22/2 \mod 7, \ldots$, which is 4. Which means:

$$\frac{1}{2} \mod 7 = 4 \tag{I.6}$$

The total fraction then yields:

$$3 \cdot \frac{1}{2} \mod 7 = 3 \cdot 4 \mod 7 = 12 \mod 7 = 5 \tag{I.7}$$

This results in:

$$\delta_1(0) = 2 \cdot 5 \mod 710 \mod 7 = 3 \tag{I.8}$$