Iterative Learning Pressure Control and Leakage Localization in Water Network Distribution

Master's thesis

Group CA10-931

Aalborg University
Department of Electronic systems
Fredrik Bajers Vej 7c
DK-9220 Aalborg

AALBORG UNIVERSITY
STUDENT REPORT

MSc in Control and Automation
Department of Electronic systems
Fredrik Bajers Vej 7C
DK-9220 Aalborg
http://es.aau.dk/

Title:
Iterative Learning Pressure Control and Leakage Localization in Water Network Distribution

Theme:
Master's Thesis

Project Period:
Autumn 2021 - Summer 2022

Project Group:
CA10-931

Participant(s):
Danas Sirvidas
Artis Revelis
Ibrahim Abdiweli

Supervisor(s):
Carsten S. Kallesøe
Saruch S. Rathore

Copies: 1

Page Numbers: 91

Date of Completion:
June 2, 2022

Abstract:

The purpose of this project is to test if it is possible to implement an Iterative Learning-based Control (ILC) in the AAU Smart Water Infrastructure Laboratory and what effect it has on such a system. By taking this into account, the plan was to try and implement ILC and include a leakage detection method in the water network. This specific implementation was plausible due to consumers using the water repeatedly, thus allowing for ILC to be used. In this report, specific requirements were made to allow the testing of such control. Later in the report, modeling for the water distribution network was made, which includes static and dynamic models. Then, after introducing ILC, multiple tests were conducted for both simulations and in an actual lab. To detect the leakages in the network, pressure residuals are generated from the difference between the dynamic model measurements and estimated values from the static model to compare it with a threshold that identifies the leak. In conclusion, it is plausible for ILC based controller to be implemented in the water lab and can be expanded even further in the future.

CONTENTS

PREFACE

This project was created by the third/fourth-semester student group 931 from the Engineering Masters' Degree of Automation and Control at Aalborg University during the period from autumn of 2021 to summer 2022. The supervisors for this project were Carsten S. Kallesøe and Saruch S. Rathore. The literature is referenced in the bibliography. The reference to the literature is a number in the square brackets. Figures and tables are numbered based on sections they are in.

Many thanks to the supervisors, Carsten S. Kallesøe and Saruch S. Rathore, who have provided guidance and support throughout the entire project. A special thank to Saruch S. Rathore for his support in the smart water lab with the network setup.

Aalborg University, June 2, 2022

Signatures

[illegible]

Danas Sirvidas
dsirvi16@student.aau.dk

Revelis

Artis Revelis
arevel17@student.aau.dk

Ibrahim

Ibrahim Abdiweli Jama
ijama20@student.aau.dk

Chapter 1

Introduction

Water is a vital resource no matter where in the world. Therefore, access to good quality water is a basic right for all human beings. It plays an essential role in life and the conservation of the health of the population.[1]

Water resources and infrastructure are facing challenges caused by the rapid global changes such as population growth, economic development, migration, and urbanisation, with over half of the world population now living in cities, which introduces new strains on the systems that supply water to the citizens, businesses, industries, and institutions.[2] Therefore, the function of a water network distribution is to deliver water to all the customers, of the network, in sufficient quantity for potable drinking water at the appropriate pressure, with minimal loss, of safe and acceptable quality.[3]

The project aims to implement a self-adjusting control method that becomes more accurate over time. The controller will be implemented in AAU Smart Water Infrastructure Laboratory setup, which contains a simulated water network. The simulated water network has a set of pumps, consumers and pipes.
The goal is to have an iterative learning control to reduce error over time. Iterative learning control will be implemented due to the repetitive flow pattern in the water network. Combined with leakage detection, it would reduce water waste.

1.1 Chapter overview

This section gives an overview of chapters contained in this project, followed by a brief introduction of each chapter.
Chapter 2: Problem analysis investigates if it is possible to use iterative learning control. It is done by first investigating what water distribution networks consist of then what types of leakages occur in them. The chapter also describes if there have been changes to household water consumption patterns. Since iterative learning control is used on periodic systems. Problem formulation is formulated after the conclusion of problem analysis.
Chapter 3: Requirements describes system requirements that later will be tested, to see if the implementation of iterative learning control is a success.
Chapter 4: Modelling describes mathematical models for the water distribution network. In the chapter, graph theory is used to derive static and dynamic models for the simulation

of water networks.
Chapter 5: Lab setup describes the AAU Smart Infrastructure Laboratory water network layout used in the test chapter. Descriptions of the pumping unit, consumer unit and pipe unit, that are used in the water network, are given in this chapter.
Chapter 6: Iterative learning pressure control introduces the control method used for the project, by first giving an overview. Then the input update law is described, and the stability analysis of the system controlled by iterative learning control is presented.
Chapter 7: Tests describes simulation tests of iterative learning controller across several versions of the models together with graph representations. The different model versions are separated into static models and dynamic models.
Chapter 8: Leakage detection describes a method that is based on residuals of the pressure in the water distribution network. The static model is used as the prediction model to generate the pressure residuals for the dynamic model. And the general structure of the leakage detection method is given.
Chapter 9: Discussion evaluates the simulation results, lab results and leakage detection. A comparison between test results and Chapter 3 is made. Lastly, there is a section with possible future improvements.
Chapter 10: Conclusion assesses the overall state of the report. And answers a question If the problem formulated in Section 2.4.1 has been solved.

Chapter 2

Problem analysis

2.1 Water distribution network

A water distribution network(WDN) is a part of a water supply network(WSS). WDN is a system to deliver water from safe water sources to residential, industrial and commercial consumers. Depending on the consumer there are different consumption patterns. However, the purpose is to deliver water at a pressure that satisfies the water consumption, without impeding the quality or safety of the water. Figure 2.1 is an example of a WDN.

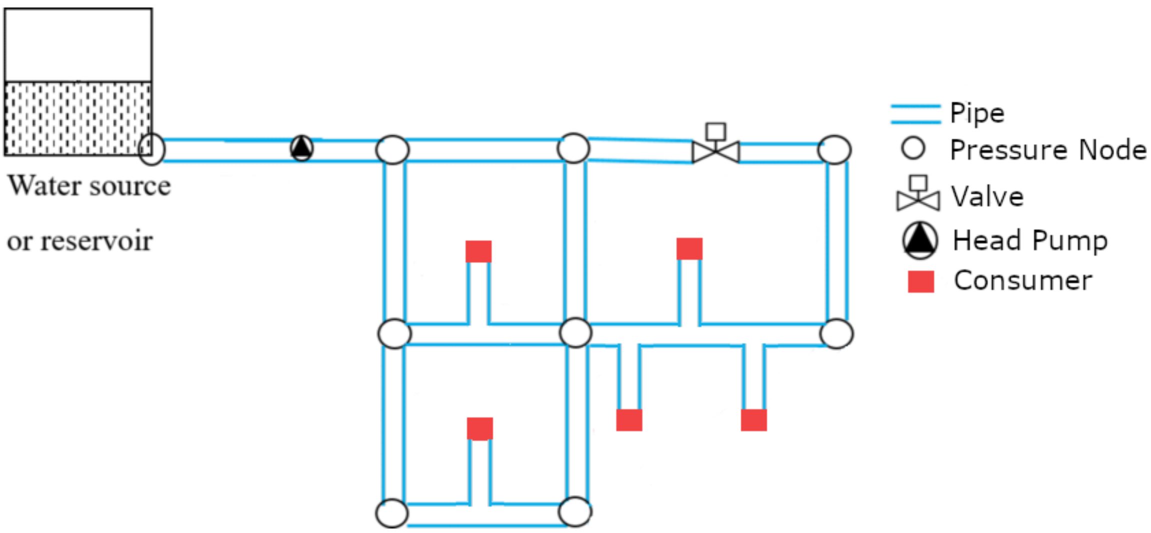

Figure 2.1: Schematics of a water distribution network based on [4]. Pressure nodes contain pressure sensors

WDN consist of pipes, valves, pumps and water sources to supply water to consumers or designated areas. The water source can be a water reservoir or a water tower. WDN can be controlled to keep a designated pressure across the network. Or it can be divided into different operational zones, that can be controlled individually. A network that is separated into different operational zones is called District Metered Area (DMA) or Pressure Management Area (PMA). In the current literature pressure and flow sensors are placed in intersection nodes to localize leakages [5],[6],[7].

2.2 Leakage types

The pressure control is quite cost-effective for reducing leakages since it is easier to detect pressure drops in certain areas of the water network. There are three considerations regarding what gets classified as leakage faults [8]:

- Reported leaks and breaks- have a high flow rate and short run time, which is easier to detect than other leaks.
- Unreported leaks or breaks- have average flow rates and a long run time. Usually, these leaks are detected by leakage detection programs.
- Background leaks- have a small flow rate and can take a very long time to be detected due to the small size of the break or leak. Usually, flow rates are too low to be detected by leakage detection systems, so another type of detection method needs to be used.

The most common ways of pressure management are establishing different small zone boundaries or DMAs, adding fixed outlet valves, controlling the level and pump outputs, different modulated flow and time valves, etc.

In this reference, [8] there are some examples of showing multiple DMAs for Madraset Arama in Alexandria, Egypt. The first example Fig. 2.2 shows how the water network looks with multiple DMAs in the city, while the second one Fig. 2.3 shows only one isolated DMA from the rest of the network, such that the reader could visualize the difference between multiple DMAs or a single one.

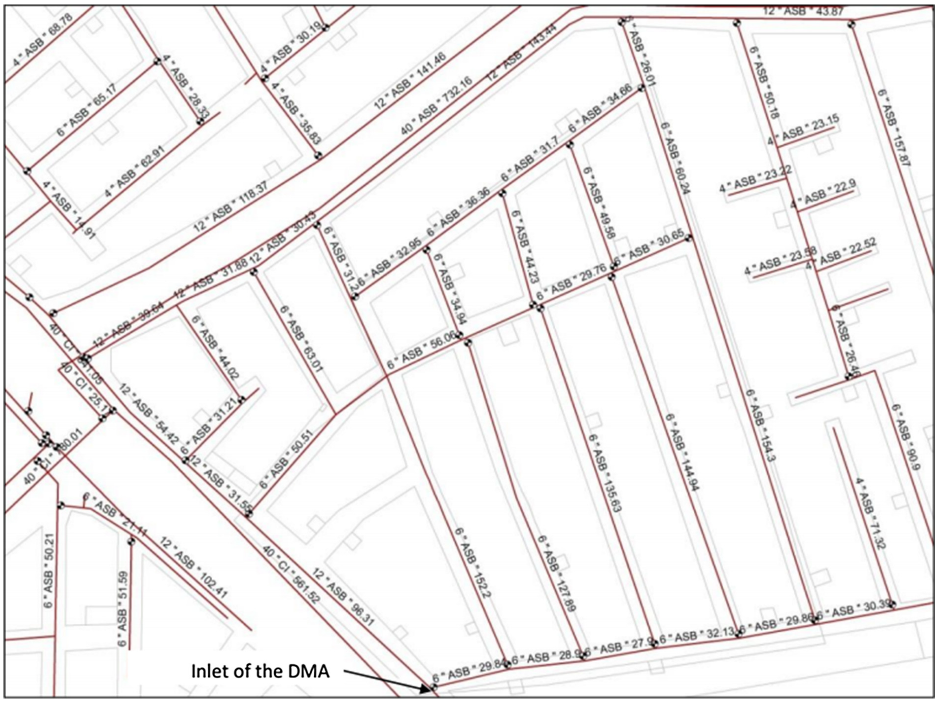

Figure 2.2: Madraset Arama DMAs network with one of the inlets for DMA [8]

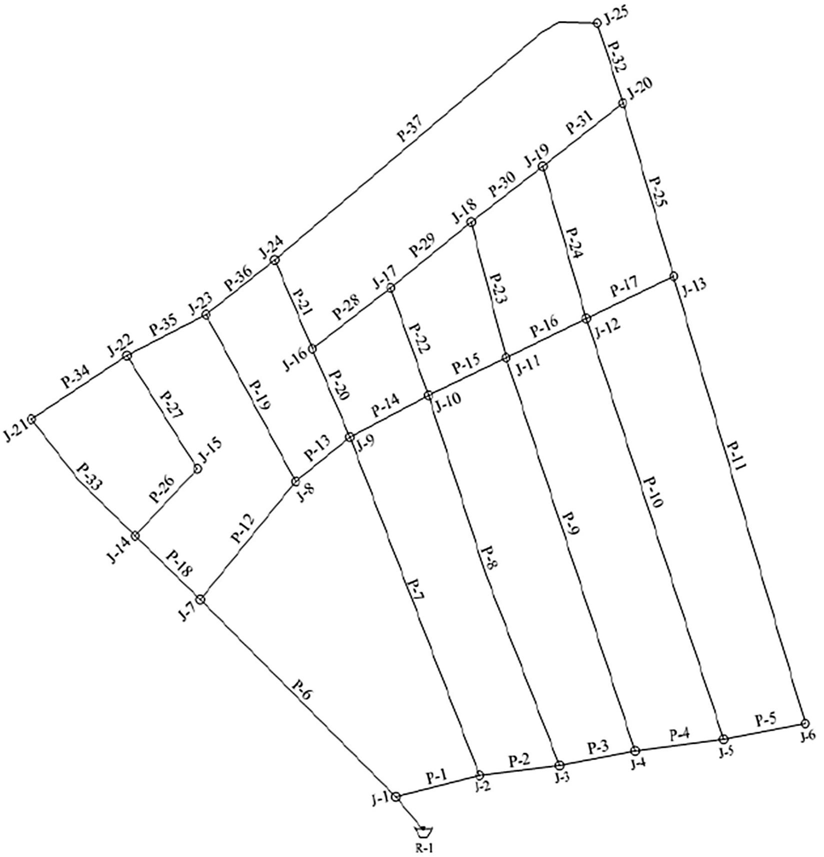

Figure 2.3: Simplified singular DMA from Madraset Arama DMAs network [8]

Pressure Reducing Valves (PRVs) seems to be the most used and effective method of controlling the pressure in the water network. PRVs are instruments installed at strategic places in the network to reduce the pressure to a set level. PRVs are usually placed within DMA, next to the flow meter such that the meters' accuracy does not get affected [9]. The placement in the network can be seen in Fig. 2.4.

Figure 2.4: PRV (blue colored) installed on the pipe which is part of DMA network [9]

It can be complicated to see the actual placement of PRV from previous figure, so the model version in the pipe is also made for readers to understand. This is represented in Fig. 2.5.

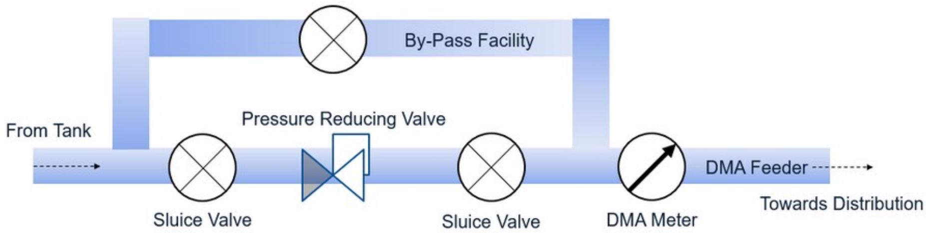

Figure 2.5: PRV inside the pipe network model [10]

2.3 Household water consumption patterns

It is important to understand if COVID-19 had an impact on household water consumption patterns. Since water consumption patterns need to be repetitive for iterative learning control to function.

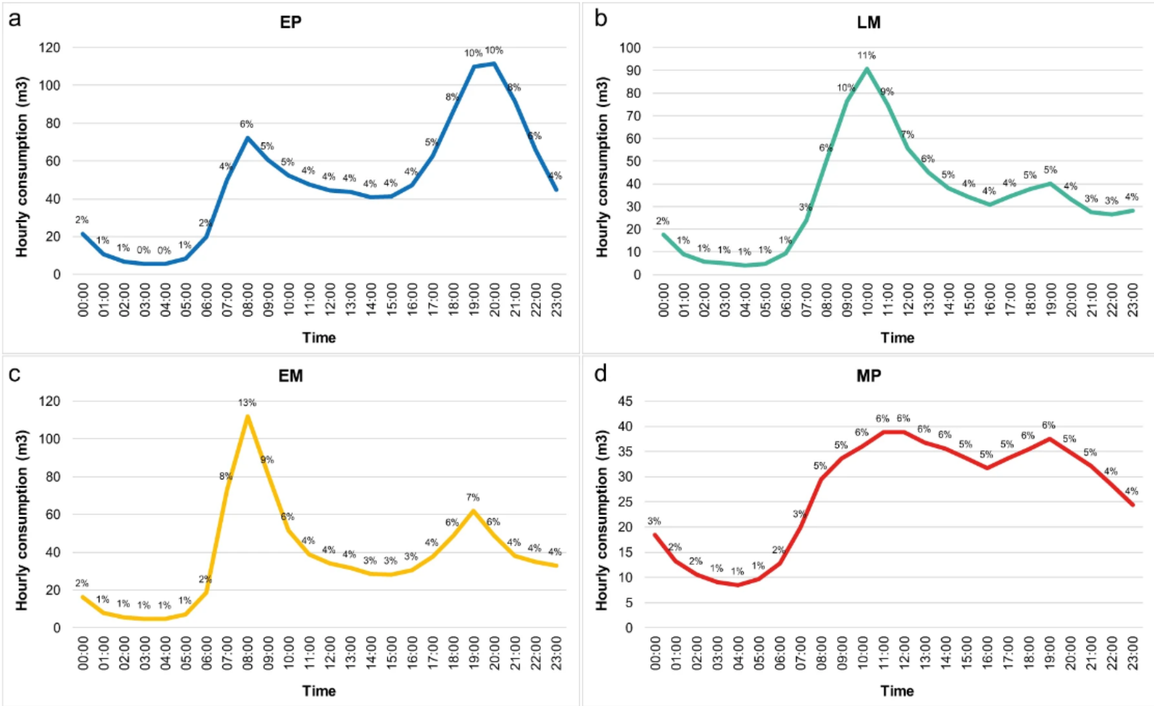

Figure 2.6: Cumulative patterns and percentages of hourly consumption for UK households during COVID-19 pandemic in the "Evening Peak (EP)", "Late Morning Peak Peak (LM)", "Early Morning Peak (EM)" and "Multiple Peak (MP)" clusters [11]

As seen in figure 2.6 the water consumption in households is not constant over the 24 hours. Due to it, the water pressure, in water networks, has to be adjusted to minimize water losses. Increased water flow increases the pressure, and the increased pressure on pipes and holes leads to higher leakage.

In figure 2.7e compared to figure 2.7f it can be noted that water consumption has increased during the COVID-19 pandemic. Water consumption during COVID-19 has higher maximum values in all of the patterns, except the early morning pattern.

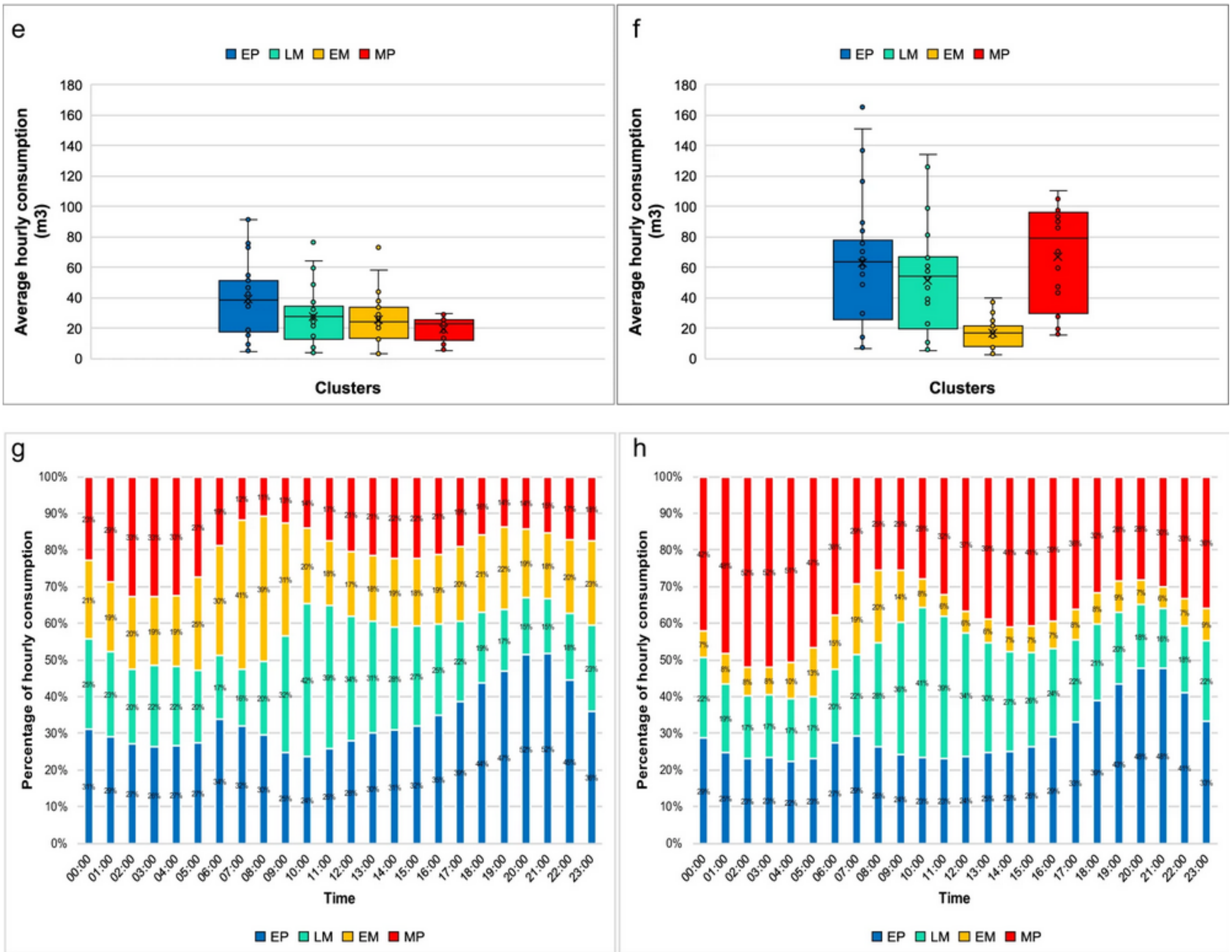

Figure 2.7: Charts for peak hours of water consumption in the UK during COVID-19. Chart e is the average hourly water consumption before COVID-19. Chart f is the average hourly water consumption during COVID-19 in the UK. Chart g is the percentage of hourly water consumption before COVID-19 in the UK. Chart h is the percentage of hourly water consumption during COVID-19 in UK [11]

2.4 Problem analysis conclusion

Iterative learning control will be used in this project since the household water consumption patterns are periodic. However, there have been changes in daily water consumption patterns, due to Covid-19. The average household water consumption, during COVID-19, has increased. The most common consumption pattern is the multiple peak pattern. It is possible to simulate WDN on the AAU Smart Water Infrastructure Laboratory setup. Specifically, the household water consumption pattern will affect the pressure of the water delivered to the consumer. Therefore, the iterative learning control will be used to keep the pressure at the desired reference for the consumers.

2.4.1 Problem formulation

We will investigate:

How an iterative learning based controller at the inlet, in addition to a leakage detection method at the outlets, can accommodate the pressure requirements in the water network, which has a periodic disturbance?.

Chapter 3

Requirements

From Chapter 2, it was decided to implement an iterative learning controller in the water lab setup and design a leakage detection method in the dynamic model. Requirements were formulated to later be compared to test results.

3.1 Simulation requirements

1. Iterative learning control should stabilize the system at the reference point after five iterations, if the consumer consumption is considered to be a sinusoidal wave.
2. Iterative learning control should stabilize the system at the reference point of 0.2 Bar even with more consumers that act as disturbances to the network.

3.2 Lab requirements

1. Can the iterative learning controller control the pressure to reach the desired reference of 0.5 Bar when a constant valve opening degree is used in the network?.
2. Using the iterative learning control, water pressure has to reach the desired reference of 0.5 Bar when the valve dynamics has a periodic opening degree.
3. Iterative learning control should never go above 100% of the maximum propeller speed.

3.3 Leakage detection requirements

1. The generated pressure residuals should indicate whether or not there is a leakage in the network model.

After the requirements are finalized, it is time to describe the modelling for static and dynamic systems in the next chapter.

Chapter 4

Modelling

In this chapter a mathematical models for the water distribution network is formulated using models of the different components that form the WDNs and derived with graph theory principles. In Section 4.1 the models for the components of the WDNs are derived and in Section 4.2 an introduction to the graph theory is presented as well as the derivation of the mathematical model for the WDN using graph theory principles.

4.1 Water distribution network components

A water distribution network is an integrated system that consist of different components such as pipes, pumps, and valves. carrying demanded water amounts at preferred pressures and water qualities to consumers, therefore in this section the model for each component is presented and the derivation of these component models will follow[12], [13] and [14].

4.1.1 Pipe

The change in pressure in all components can be expressed as

$$\Delta p_k = p_{in,k} - p_{out,k} \tag{4.1}$$

Where,
Δp_k is the pressure drop across the k^{th} component $[Pa]$
$p_{in,k}$ is the pressure at the k^{th} component input node $[Pa]$
$p_{out,k}$ is the pressure at the k^{th} component output node $[Pa]$

The pipes in the network are assumed to have the same diameter that is constant along the pipe's length. The flow is assumed to be uniform along the pipe's cross-section. Therefore the dynamic model for pipes in the water network distribution, as presented in [14], can be described as

$$\Delta p_k = \mathcal{J}_k \dot{q}_k + \lambda_k(q_k) - \zeta_k \tag{4.2}$$

where,
Δp_k is the drop in pressure across the k^{th} pipe $[Pa]$
$\mathcal{J}_k$ is the mass inertia of water in the k^{th} pipe $\left[\frac{kg}{m^4}\right]$

q_k is the flow of water through the k^{th} pipe $\left[\frac{m^3}{s}\right]$
$\lambda_k(q_k)$ is the drop in pressure due to surface and form resistances in the k^{th} pipe $[Pa]$
ζ_k is the drop in pressure due to elevation $[Pa]$
The mass inertia of water in the pipe can be expressed as

$$\mathcal{J}_k = \frac{L\rho}{A} \tag{4.3}$$

Where,
L is the pipe length $[m]$
ρ is the density of the water $\left[\frac{kg}{m^3}\right]$
A is the cross sectional area to the pipe $[m^2]$

The pressure drop in the pipe due to surface resistance h_f is given by Darcy-Weisbach equation[15]:

$$h_f = \frac{8fLq^2}{\pi^2 g D^5} \tag{4.4}$$

where,
h_f is the head loss due to surface resistance $[m]$
f is the coefficient of surface resistance known as friction factor
g is the gravitational constant $\left[\frac{m}{s^2}\right]$
D is the diameter of the pipe $[mm]$
The friction factor f is dependent on the Reynolds number Re, which describes the flow type and the average height of roughness ϵ of the pipe wall. By assuming turbulent flow of the water, the friction factor of the pipe can be given by

$$f = 1.325\left[log\left(\frac{\epsilon}{3.7D} + \frac{5.74}{Re^{0.9}}\right)\right]^{-2} \tag{4.5}$$

where $Re \geq 4000$ for turbulent flow.
The pressure drop due form resistance can be expressed as

$$h_m = k_f\frac{8q^2}{\pi^2 g D^4} \tag{4.6}$$

where,
h_m is the head loss due to form resistance $[m]$
k_f is the form-loss coefficient $[m]$
the term k_f can take different values in different situations. Such as a change in an area where the pipe bends or change in direction, where elbows can be used to provide a sharp turn, the form-loss coefficient k_f in these situations can be expressed as

$$k_f = \left[0.0733 + 0.923\left(\frac{D}{R}\right)^{3.5}\right]\alpha^{0.5} \tag{4.7}$$

where,
R is the bend radius $[mm]$
α is the bend angle $[rad]$

$$k_f = 0.442\alpha^{2.17} \tag{4.8}$$

where,
α is the elbow angle $[rad]$
The drop in pressure due elevation can be given by

$$\Delta z = \rho g \Delta h \tag{4.9}$$

4.1.2 Valve

The valves in this project for the water distribution network are valves with variable opening degrees (OD). The changes in the (OD) can alter the flow rate of the water and the pressure drop across the valve. Modelling of the valves was approached similarly to the pipe model in Eq. 4.2. Where length L and elevation change Δz are assumed to be zero since the valve is relatively small compared to the pipe.

$$\Delta p_k = \mu_k(q_k, OD_k) = \frac{1}{k_{vs}(OD_k)^2}|q_k|q_k \tag{4.10}$$

Where,
$\mu_k(q_k, OD_k)$ is a function expressing the pressure drop over the k^{th} valve $[Pa]$
$k_{vs}(OD_k)$ is the conductivity function of the k^{th} valve
OD_k is the opening degree of the k^{th} valve

4.1.3 Pump

The pumps in the water distribution network are used to deliver pressure and flow into the network and in this project the centrifugal pumps was chosen. the pressure difference of the pump is depend on two main variables that is: the flow through the pumps and the rotational speed of the centrifugal pumps. The model was derived based on these variables and can be presented such as,[13]

$$\Delta p_k = \alpha_k(q_k, \omega_k) = -a_{h2,k}|q_k|q_k + a_{h0,k}\omega_k^2 \tag{4.11}$$

Where,
$\alpha_k(q_k, \omega_k)$ is a function representing pressure delivered by the k^{th} pump $[Pa]$
$a_{h2,k}$,$a_{h0,k}$ are the pump constants of the k^{th} pump
ω_k is the rotational speed of the k^{th} pump

4.1.4 General component model

The general model is formulated by combining the component models namely the pipe model in Eq. 4.2, the valve model in Eq. 4.10 and the pump model in Eq. 4.11 as shown in Eq. 4.12.

$$\Delta p_k = \mathcal{J}_k \dot{q}_k + \lambda_k(q_k) + \mu_k(q_k, OD_k) - \alpha_k(q_k, \omega_k) - \Delta z_k \tag{4.12}$$

Where,

$$\mathcal{J}_k = \frac{L_k \rho}{A_k} \tag{4.13}$$

$$\lambda_k(q_k) = \rho g (f \frac{8L_k}{\pi^2 g D^5} + k_f \frac{8}{\pi^2 g D^4}) |q_k| q_k \tag{4.14}$$

$$\mu_k(q_k, OD_k) = \frac{1}{k_{vs}(OD_k)^2} |q_k| q_k \tag{4.15}$$

$$\alpha_k(q_k, \omega_k) = -a_{h2,k} |q_k| q_k + a_{h0,k} \omega_k^2 \tag{4.16}$$

$$\Delta z_k = \rho g \Delta h_k \tag{4.17}$$

Component	$\mathcal{J}_k$	λ_k	μ_k	α_k	Δz_k
k^{th} pipe	$\mathcal{J}_k$	λ_k	0	0	Δz_k
k^{th} valve	0	0	μ_k	0	0
k^{th} pump	0	0	0	α_k	0

Table 4.1: The general component model parametrization

Eq. 4.12 is used to describe the pressure drop across each component. It combines the different component models. Pressure drop across a specific component, for example the pump, can be calculated by setting other components to be zero. This can also be done by combining Eq. 4.12 and Table 4.1.

4.2 Graph theory based network model

In this section the fundamentals of graph theory are presented as well as the static model and dynamic model that are derived using the graph theory principles.

4.2.1 Basics of graph theory

Graph theory can be used to represent water networks as sets of vertices and edges. An example of a graph is shown in Fig. 4.1 and the following definitions and matrices can be introduced with the aid of this figure.

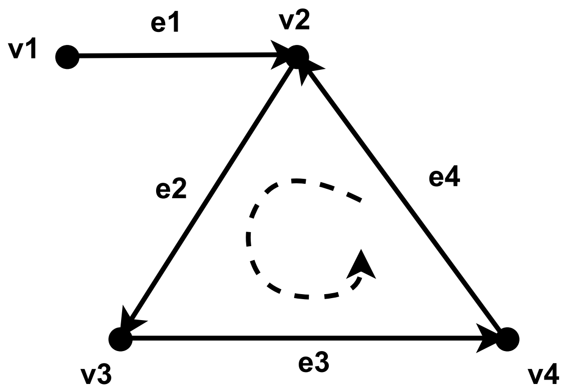

Figure 4.1: A graph example

Graph

A graph $G = (V, E)$ consist of a set of objects $V = \{v_1, v_2, \cdots, v_n\}$ called vertices, and another set $E = \{e_1, e_2, \cdots, e_m\}$, whose elements are called edges, such that each edge e_k is identified with pair (v_i, v_j) of vertices[16].

Loop or cycle

A loop or cycle of a graph is a closed walk where the initial and terminal vertices are identical [17], such closed walk could be $\{v_2, e_2, v_3, e_3, v_4, e_4, v_2\}$.

Tree

A tree T Given a connected graph G a tree T is a graph which contains no cycles by removing any of its edges[18].

Spanning tree

A spanning tree is a tree T that connects all the vertices of the graph G[18], such as $\{v_1, e_1, v_2, e_2, v_3, e_3, v_4\}$.

Chord

A chord is an edge of G that is not in the given spanning tree T[16], edge $\{e_4\}$ can be considered as a chord.

In water distribution networks, components such as pipes, valves, and pumps are represented as edges, the connections between these components are represented as vertices. The following two matrices are also used in the derivation of the model for the WDN.

Incidence matrix

An incidence matrix H of a digraph with n vertices and m edges is an n by m matrix, such that[16]

$$H_{ij} = \begin{cases} 1 & \text{edge } e_j \text{ is leaving vertex } v_i \\ -1 & \text{edge } e_j \text{ is entering vertex } v_i \\ 0 & \text{otherwise} \end{cases} \tag{4.18}$$

Incidence matrix Eq. 4.19 is created by following rules of Eq. 4.18.

$$H = \begin{array}{c} \begin{array}{cccc} e_1 & e_2 & e_3 & e_4 \end{array} \\ \begin{bmatrix} 1 & 0 & 0 & 0 \\ -1 & 1 & 0 & -1 \\ 0 & -1 & 1 & 0 \\ 0 & 0 & -1 & 1 \end{bmatrix} \end{array} \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \end{array} \tag{4.19}$$

A reduced incidence matrix can be obtained by choosing one of the nodes in the graph a reference node, node $\{v_4\}$ is chosen to be the reference node and edge $\{e_4\}$ as a chord edge, the reduced incidence matrix can be presented as,

$$\overline{H} = \left[\begin{array}{c|c} \overline{H}_C & \overline{H}_T \end{array} \right] \tag{4.20}$$

Where,
$\overline{H}_C$ is the reduced matrix for the chord edges in the graph
$\overline{H}_T$ is the reduced matrix for the tree edges in the graph

$$\overline{H} = \begin{array}{c} \begin{array}{c|ccc} e_4 & e_1 & e_2 & e_3 \end{array} \\ \left[\begin{array}{c|ccc} 0 & 1 & 0 & 0 \\ -1 & -1 & 1 & 0 \\ 0 & 0 & -1 & 1 \end{array}\right] \end{array} \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \end{array} \tag{4.21}$$

Cycle matrix

for a given connected graph G that has a spanning tree T, the loops that are formed by adding the chord edges to the graph are called cycles, therefore the cycle matrix B of a digraph is a matrix defined as,

$$B_{ij} = \begin{cases} 1 & \text{edge } e_j \text{ is in the loop direction} \\ -1 & \text{edge } e_j \text{ is opposite the loop direction} \\ 0 & \text{otherwise} \end{cases} \tag{4.22}$$

The cycle matrix can be partitioned into chord matrix and tree matrix as presented below,[13]

$$B = \begin{bmatrix} B_C & B_T \end{bmatrix} \tag{4.23}$$

The cycle matrix for the graph in the figure above is given as,

$$
B = \begin{array}{c} \begin{matrix} e_4 & e_1 & e_2 & e_3 \end{matrix} \\ \left[\begin{array}{c|ccc} 1 & 0 & 1 & 1 \end{array}\right] \end{array} \tag{4.24}
$$

The matrix does not contain negative entries that is because all the edges that forms the cycle in the graph follows the same direction as the loop.
The fundamental cycle matrix can be represented in terms of reduced incidence matrix as follows,[13]

$$
B = \begin{bmatrix} I & -\overline{H}_C^T \overline{H}_T^{-T} \end{bmatrix} \tag{4.25}
$$

4.2.2 Water network model

A two-terminal hydraulic element is represented by edge e_k. Associated with each edge are two edge variables, p_k and q_k. The variable p_k is called the edge pressure and may be regarded as a cross variable because it exists across the two end nodes of the edge. The other variable q_k is called the edge flow and flows through the edge. A hydraulic network can be described the same way as an electrical network. The current is equivalent to the flow in the edges and the voltage is equivalent to the pressure. The edge variables must also obey the two laws of Kirchhoff's [16]:
Kirchhoff´s Current Law (KCL) states that the net sum of all the current flows leaving and entering a node is zero.

$$
Ai = 0 \tag{4.26}
$$

Where,
A is the incidence matrix of the graph
i is the vector of current flows
The voltage difference across the edge can be described as

$$
\Delta v = A^T v \tag{4.27}
$$

where Δv is the voltage difference
Regarding Kirchhoff´s Voltage Law (KVL) it is stated that at any time the net sum of the voltage drops in a cycle is zero.

$$
B\Delta v = 0 \tag{4.28}
$$

where B is the cycle matrix of the graph.
Similar to the electrical network the KCL Eq. 4.26 can be formulated in terms of hydraulic network as follows:

$$
Hq = d \tag{4.29}
$$

where,
H is the incidence matrix

q is the vector of flows through the edges
d is the vector of nodal demands which are the nodes open to atmospheric pressure
The nodal demands take a negative sign if the water flows out of them, such as nodes connected to a consumer. And positive sign if the water flows in, such as nodes connected to a pump. The remaining nodes have no disturbance meaning no water is removed nor added to the system therefore they take values of 0.

$$d = d_{p_1} + d_{p_2} + d_{c_1} + d_{c_2} = 0 \tag{4.30}$$

where,
d_{p_1} and d_{p_2} are the pump flow rate
d_{c_1} and d_{c_2} are the consumer flow rate

$$d = Fd_f \tag{4.31}$$

Eq. 4.31 is formulating Eq. 4.30 in matrix form, where F is a matrix extracting the non-zero nodal demands d_f from the vector of nodes. Moreover, the reduced F matrix can be represented by removing the reference node from d_f vector, and can be written as follows:

$$\overline{d} = \overline{Fd_f} \tag{4.32}$$

Lemma 1

Let q be the vector of flows through the edges in a flow network with underlying graph G and let n be the number of vertices in G. With T denote a particular spanning tree of G and q_C the vector of flows through the chords of T with respect to G. Lastly, let $\overline{H}_T$ be the reduced incidence matrix of T and $\overline{d}$ be the vector of node flows for the non-reference nodes. Then the following is true [13],

$$q = B^T q_c + \begin{bmatrix} 0 \\ \overline{H}_T^{-1} \end{bmatrix} \overline{d} \tag{4.33}$$

From Eq. 4.33, B^T matrix is as follows:

$$B^T = \begin{bmatrix} I \\ -\overline{H}_T^{-1}\overline{H}_c \end{bmatrix} \tag{4.34}$$

The flow vector can be split into chord edge flow and tree edge flow, and substitute B^T matrix and $\overline{d}$ with their corresponding values as follows:

$$q = \begin{bmatrix} q_c \\ q_T \end{bmatrix} = \begin{bmatrix} I \\ -\overline{H}_T^{-1}\overline{H}_c \end{bmatrix} q_c + \begin{bmatrix} 0 \\ \overline{H}_T^{-1} \end{bmatrix} \overline{Fd_f} \tag{4.35}$$

From Eq. 4.35 q_T can be expressed as:

$$q_T = -\overline{H}_T^{-1}\overline{H}_c q_c + \overline{H}_T^{-1}\overline{Fd_f} \tag{4.36}$$

Differentiating Eq. 4.33 can be given as:

$$\dot{q} = B^T \dot{q}_c + \begin{bmatrix} 0 \\ \overline{H}_T^{-1} \end{bmatrix} \overline{F}\dot{\overline{d_f}} \tag{4.37}$$

From Eq. 4.37 $\dot{q_T}$ can be expressed as:

$$\dot{q_T} = -\overline{H}_T^{-1}\overline{H}_c\dot{q}_c + \overline{H}_T^{-1}\overline{F}\dot{\overline{d_f}} \tag{4.38}$$

4.2.3 Static model

In this section, the network is assumed to have a single pump which is also considered as a reference node for the network and takes the following assumptions. This model acts as a steady state model which discards the dynamics of the pipes in the network.

Assumption 1.

The distribution between the $n-1$ non-inlet demands $\overline{d}$, is fixed in time, that is, there exist $\nu \in \mathbb{R}_+^{n-1}$ with the property $\sum_{i=1}^{n-1} \nu_i = 1$, and d_n is the total non-inlet demand such that[19]

$$\overline{d}(t) = -\nu \cdot d_n(t) \tag{4.39}$$

Assumption 2.

The hydraulic resistance takes this form:[19]

$$\lambda_i(q_i) = f_i|q_i|q_i \qquad with \quad f_i > 0 \tag{4.40}$$

The nodal demand of the network in the static model is

$$d = \mathcal{F}_p d_p + \mathcal{F}_c d_c \tag{4.41}$$

Where,
$\mathcal{F}_p$ is a matrix that draws inlet demand vector d_p from the nodal demand vector d
$\mathcal{F}_c$ is a matrix that draws outlet demand vector d_c from the nodal demand vector d
by removing the reference node the nodal demand is given by

$$\overline{d} = \overline{\mathcal{F}}_p \overline{d}_p + \overline{\mathcal{F}}_c \overline{d}_c \tag{4.42}$$

Where,
$\overline{\mathcal{F}}_p$ is the $\mathcal{F}_p$ after the removal of the reference node, $\overline{\mathcal{F}}_c$ is the $\mathcal{F}_c$ after the removal of the reference node

In this static model the tree edge flows of the network is calculated from Eq. 4.33 where $\overline{d}$ is equal to $\overline{F}_c\overline{d}_c$

$$q_T = -\overline{H}_T^{-1}\overline{H}_c q_c + \overline{H}_T^{-1}\overline{F}_c\overline{d}_c \tag{4.43}$$

From assumption 1 and 2, there exist a vector B, which is partitioned into chord and tree parts, as shown in the following equation:

$$q(t) = \begin{bmatrix} q_C(t) \\ q_T(t) \end{bmatrix} = \begin{bmatrix} a_C \\ a_T \end{bmatrix} d_n(t) = a \cdot d_n(t) \tag{4.44}$$

This means that for a given ν, in the consumer demand, there exist a vector a in all edge flows of the network.[19]

$$q_C = a_C d_n \tag{4.45}$$

$$\overline{d} = -\nu d_n \tag{4.46}$$

$$\Delta p = \lambda(q) - \Delta z \tag{4.47}$$

Similar to electrical network, the KVL Eq. 4.28 can be formulated in terms of hydraulic network as follows:

$$B\Delta p = 0 \tag{4.48}$$

$$B\Delta z = 0 \tag{4.49}$$

$$B\lambda(q) = 0 \tag{4.50}$$

Using Eq. 4.48 and Eq. 4.49 in Eq. 4.47

$$\lambda_C(q_C) - \overline{H}_c^T\overline{H}_T^{-T}\lambda_T(q_T) = 0 \tag{4.51}$$

$$\lambda_C(q_C) - \overline{H}_c^T\overline{H}_T^{-T}\lambda_T(-\overline{H}_T^{-1}\overline{H}_c q_c + \overline{H}_T^{-1}\overline{F}_c\overline{d}_c) = 0 \tag{4.52}$$

Using Eq. 4.45 and Eq. 4.46 gives

$$(\lambda_C(a_C) - \overline{H}_c^T\overline{H}_T^{-T}\lambda_T(-\overline{H}_T^{-1}\overline{H}_c a_C - \overline{H}_T^{-1}\nu))d_n^2 = 0 \tag{4.53}$$

The pressure difference across edges is similar to voltage difference in electrical network given in Eq. 4.27:

$$\Delta p = H^T p \tag{4.54}$$

The elevation difference across edges can also be given as:

$$\Delta z = H^T z \tag{4.55}$$

The vector of pressure difference, elevation difference and incidence matrix can be split into chord edges and tree edges.

$$\begin{bmatrix} \Delta p_C \\ \Delta p_T \end{bmatrix} = \begin{bmatrix} H_C^T \\ H_T^T \end{bmatrix} p \tag{4.56}$$

$$\begin{bmatrix} \Delta z_C \\ \Delta z_T \end{bmatrix} = \begin{bmatrix} H_C^T \\ H_T^T \end{bmatrix} z \tag{4.57}$$

$$\Delta p_T = H_T^T p = \lambda_T(q_T) - H_T^T z \tag{4.58}$$

Lemma 2

Let T be a directed tree with the incidence matrix H_T and reduced incidence matrix $\overline{H}_T$ (without loss of generality, assuming that the last row of H_T has been deleted to obtain $\overline{H}_T$). The reduced incidence matrix is invertible, since a tree is a connected graph with n-1 edges. Then the following holds [20],

$$H_T\overline{H}_T^{-1} = \begin{bmatrix} I_{n-1} \\ -\mathbb{1}^T \end{bmatrix} \tag{4.59}$$

where $\mathbb{1}$ is a vector of ones, I_{n-1} is $n-1$ identity matrix and Transposing Eq. 4.59 gives,

$$(H_T\overline{H}_T^{-1})^T = \overline{H}_T^{-T} H_T^T = \begin{bmatrix} I & -\mathbb{1}^T \end{bmatrix} \tag{4.60}$$

The pressure vector can be split into non-reference node pressures $\overline{p}$ and reference node pressure p_o.

$$p = \begin{bmatrix} \overline{p} \\ p_o \end{bmatrix} \tag{4.61}$$

Similarly, vector of pressure, due to elevation, is divided into non-reference node pressures due to elevation $\overline{z}$ and reference node pressure due to elevation z_o.

$$z = \begin{bmatrix} \overline{z} \\ z_o \end{bmatrix} \tag{4.62}$$

$$\overline{H}_T^{-T} H_T^T p = \overline{p} - \mathbb{1}p_o = \overline{H}_T^{-T}\lambda_T(q_T) - (\overline{z} - \mathbb{1}z_o) \tag{4.63}$$

$$\overline{p} = \overline{H}_T^{-T}\lambda_T(q_T) - (\overline{z} - \mathbb{1}z_o) + \mathbb{1}p_o \tag{4.64}$$

$$\overline{p} = \overline{H}_T^{-T}\lambda_T(-\overline{H}_T^{-1}\overline{H}_c q_c + \overline{H}_T^{-1}\overline{F}_c\overline{d}_c) - (\overline{z} - \mathbb{1}z_o) + \mathbb{1}p_o \tag{4.65}$$

Using Eq. 4.45 and Eq. 4.46 gives

$$\overline{p}(t) = \overline{H}_T^{-T}\lambda_T(-\overline{H}_T^{-1}\overline{H}_c a_c - \overline{H}_T^{-1}\nu)d_n^2 - (\overline{z} - \mathbb{1}z_o) + \mathbb{1}p_o(t) \tag{4.66}$$

where
$\overline{p}(t)$ is the vector of pressure at the non-inlet nodes
$p_o(t)$ is the inlet node pressure
The Eq. 4.66 can be written with set of constants α_i, $\kappa(t)$ and γ_i in the following form[21]

$$\overline{p}_i(t) = \alpha_i d_n^2 - \gamma_i + u(t) \tag{4.67}$$

Where α_i = $(\overline{H}_T^{-T})_i\lambda_T(-\overline{H}_T^{-1}\overline{H}_c a_c - \overline{H}_T^{-1}\nu)$
γ_i = $(\overline{z}_i - \mathbb{1}z_o)$
$u(t) = \mathbb{1}p_0(t)$

4.2.4 Dynamic model

In this section the dynamics of the pipes, valves and pumps are taken into consideration to build the model. The vector representing pressure drop across all the edges of the graph can be described using general component model Eq. 4.12

$$\Delta p = \mathcal{J}\dot{q} + \lambda(q) + \mu(q, OD) - \alpha(q, \omega) - \Delta z \tag{4.68}$$

Using Eq. 4.48 and Eq. 4.49 in equation. Eq. 4.68

$$B\Delta p = B\mathcal{J}\dot{q} + B(\lambda(q) + \mu(q, OD) - \alpha(q, \omega)) - B\Delta z \tag{4.69}$$

$$0 = B\mathcal{J}\dot{q} + B(\lambda(q) + \mu(q, OD) - \alpha(q, \omega)) \tag{4.70}$$

$$B\mathcal{J}\dot{q} = -B(\lambda(q) + \mu(q, OD) - \alpha(q, \omega)) \tag{4.71}$$

Substituting Eq. 4.37 in Eq. 4.71 gives:

$$B\mathcal{J}B^T\dot{q}_c + B\mathcal{J}\begin{bmatrix} 0 \\ \overline{H}_T^{-1} \end{bmatrix}\dot{\overline{Fd_f}} = -B(\lambda(q) + \mu(q, OD) - \alpha(q, \omega)) \tag{4.72}$$

For simplifying Eq. 4.72 $\mathcal{J}$ is split into chord edges and tree edges $\begin{bmatrix} \mathcal{J}_C & 0 \\ 0 & \mathcal{J}_T \end{bmatrix}$

$$B\mathcal{J}B^T\dot{q}_c + \begin{bmatrix} I & -\overline{H}_C^T\overline{H}_T^{-T} \end{bmatrix}\begin{bmatrix} \mathcal{J}_C & 0 \\ 0 & \mathcal{J}_T \end{bmatrix}\begin{bmatrix} 0 \\ \overline{H}_T^{-1} \end{bmatrix}\dot{\overline{Fd_f}} = -B(\lambda_k(q) + \mu(q, OD) - \alpha(q, \omega)) \tag{4.73}$$

$$B\mathcal{J}B^T\dot{q}_c - \overline{H}_C^T\overline{H}_T^{-T}\mathcal{J}_T\overline{H}_T^{-1}\dot{\overline{Fd_f}} = -B(\lambda(q) + \mu(q, OD) - \alpha(q, \omega)) \tag{4.74}$$

Using Eq. 4.36 in Eq. 4.74 gives:

$$\begin{aligned} B\mathcal{J}B^T\dot{q}_c - \overline{H}_C^T\overline{H}_T^{-T}\mathcal{J}_T\overline{H}_T^{-1}\dot{\overline{Fd_f}} &= -B(\lambda(-\overline{H}_T^{-1}\overline{H}_c q_c + \overline{H}_T^{-1}\overline{Fd_f}) + \\ \mu(-\overline{H}_T^{-1}\overline{H}_c q_c + \overline{H}_T^{-1}\overline{Fd_f}, OD) &- \alpha(-\overline{H}_T^{-1}\overline{H}_c q_c + \overline{H}_T^{-1}\overline{Fd_f}, \omega)) \end{aligned} \tag{4.75}$$

Taking the tree part of the general component model Eq. 4.68 gives:

$$\Delta p_T = H_T^T p = \mathcal{J}_T\dot{q}_T + \lambda_T(q_T) + \mu_T(q_T, OD_T) - \alpha_T(q_T, \omega_T) - H_T^T z \tag{4.76}$$

Using Lemma 2 on Eq. 4.76 gives the following equations:

$$\overline{H}_T^{-T}H_T^T p = \overline{H}_T^{-T}\mathcal{J}_T\dot{q}_T + \overline{H}_T^{-T}(\lambda_T(q_T) + \mu_T(q_T, OD_T) - \alpha_T(q_T, \omega_T)) - \overline{H}_T^{-T}H_T^T z \tag{4.77}$$

$$\overline{p} - \mathbb{1}p_o = \overline{H}_T^{-T}\mathcal{J}_T\dot{q}_T + \overline{H}_T^{-T}(\lambda_T(q_T) + \mu_T(q_T, OD_T) - \alpha_T(q_T, \omega_T)) - \overline{z} - \mathbb{1}z_o \tag{4.78}$$

Pre-multiplying $\overline{F}^T$ with Eq. 4.78 gives:

$$\overline{F}^T(\overline{p} - \mathbb{1}p_o) = \overline{F}^T\overline{H}_T^{-T}\mathcal{J}_T\dot{q}_T + \overline{F}^T\overline{H}_T^{-T}(\lambda_T(q_T) + \mu_T(q_T, OD_T) - \alpha_T(q_T, \omega_T)) - \overline{F}^T(\overline{z} - \mathbb{1}z_o) \tag{4.79}$$

$$0 = \overline{F}^T \overline{H}_T^{-T} \mathcal{J}_T \dot{q}_T + \overline{F}^T \overline{H}_T^{-T} (\lambda_T(q_T) + \mu_T(q_T, OD_T) - \alpha_T(q_T, \omega_T)) - \overline{F}^T (\overline{z} - \mathbb{1} z_o) \tag{4.80}$$

$$\overline{F}^T \overline{H}_T^{-T} \mathcal{J}_T \dot{q}_T = -\overline{F}^T \overline{H}_T^{-T} (\lambda_T(q_T) + \mu_T(q_T, OD_T) - \alpha_T(q_T, \omega_T)) + \overline{F}^T (\overline{z} - \mathbb{1} z_o) \tag{4.81}$$

substituting $\dot{q}_T$ and q_T expressions from Eq. 4.38 and Eq. 4.36 into Eq. 4.81

$$\begin{aligned} -\overline{F}^T \overline{H}_T^{-T} \mathcal{J}_T \overline{H}_T^{-1} \overline{H}_c \dot{q}_c + \overline{F}^T \overline{H}_T^{-T} \mathcal{J}_T \overline{H}_T^{-1} \dot{\overline{F d_f}} = -\overline{F}^T \overline{H}_T^{-T} (\lambda_T(-\overline{H}_T^{-1} \overline{H}_c q_c + \overline{H}_T^{-1} \overline{F d_f}) + \\ \mu_T(-\overline{H}_T^{-1} \overline{H}_c q_c + \overline{H}_T^{-1} \overline{F d_f}, OD_T) - \alpha_T(-\overline{H}_T^{-1} \overline{H}_c q_c + \overline{H}_T^{-1} \overline{F d_f}, \omega_T)) + \overline{F}^T (\overline{z} - \mathbb{1} z_o) \end{aligned} \tag{4.82}$$

with Eq. 4.75 and Eq. 4.82, the model for the water distribution network is given and can be written in matrix form as follows:

$$\begin{aligned} \begin{bmatrix} B \mathcal{J} B^T & -\overline{H}_C^T \overline{H}_T^{-T} \mathcal{J}_T \overline{H}_T^{-1} \overline{F} \\ -\overline{F}^T \overline{H}_T^{-T} \mathcal{J}_T \overline{H}_T^{-1} \overline{H}_c & \overline{F}^T \overline{H}_T^{-T} \mathcal{J}_T \overline{H}_T^{-1} \overline{F} \end{bmatrix} \begin{bmatrix} \dot{q}_C \\ \dot{\overline{d}}_f \end{bmatrix} = - \begin{bmatrix} I & -\overline{H}_C^T \overline{H}_T^{-T} \\ 0 & -\overline{F}^T \overline{H}_T^{-T} \end{bmatrix} \\ \left(\begin{bmatrix} \lambda_C(q_C) \\ \lambda_T(q_C, \overline{d}_f) \end{bmatrix} + \begin{bmatrix} \mu_C(q_C, OD_C) \\ \mu_T(q_C, \overline{d}_f, OD_T) \end{bmatrix} - \begin{bmatrix} \alpha_C(q_C, \omega_C) \\ \alpha_T(q_C, \overline{d}_f, \omega_T) \end{bmatrix} \right) + \begin{bmatrix} 0 \\ \overline{F}^T \end{bmatrix} (\overline{z} - \mathbb{1} z_o) \end{aligned} \tag{4.83}$$

$$B = \begin{bmatrix} I & -\overline{H}_C^T \overline{H}_T^{-T} \end{bmatrix} \tag{4.84}$$

$$B_d = \begin{bmatrix} 0 & \overline{H}_T^{-T} \end{bmatrix} \tag{4.85}$$

$$B \mathcal{J} B_d^T = -\overline{H}_C^T \overline{H}_T^{-T} \mathcal{J}_T \overline{H}_T^{-1} \tag{4.86}$$

$$B_d \mathcal{J} B_d^T = \overline{H}_T^{-T} \mathcal{J}_T \overline{H}_T^{-1} \tag{4.87}$$

Eq. 4.83 can be re-written as follows by using the above four Eq. 4.84, Eq. 4.85, Eq. 4.86 and Eq. 4.87

$$\begin{aligned} \begin{bmatrix} B \mathcal{J} B^T & B \mathcal{J} B_d^T \overline{F} \\ \overline{F}^T B_d \mathcal{J} B^T & \overline{F}^T B_d \mathcal{J} B_d^T \overline{F} \end{bmatrix} \begin{bmatrix} \dot{q}_C \\ \dot{\overline{d}}_f \end{bmatrix} = - \begin{bmatrix} I & -\overline{H}_C^T \overline{H}_T^{-T} \\ 0 & -\overline{F}^T \overline{H}_T^{-T} \end{bmatrix} \\ \left(\begin{bmatrix} \lambda_C(q_C) \\ \lambda_T(q_C, \overline{d}_f) \end{bmatrix} + \begin{bmatrix} \mu_C(q_C, OD_C) \\ \mu_T(q_C, \overline{d}_f, OD_T) \end{bmatrix} - \begin{bmatrix} \alpha_C(q_C, \omega_C) \\ \alpha_T(q_C, \overline{d}_f, \omega_T) \end{bmatrix} \right) + \begin{bmatrix} 0 \\ \overline{F}^T \end{bmatrix} (\overline{z} - \mathbb{1} z_o) \end{aligned} \tag{4.88}$$

Chapter 5

Lab setup

This chapter describes the AAU Smart Water Infrastructure Laboratory(SWIL) setup for the water network. SWIL consists of multiple modules. Modules are separated into a pumping station, a consumer unit and a pipe unit, symbols seen in figure 5.1.

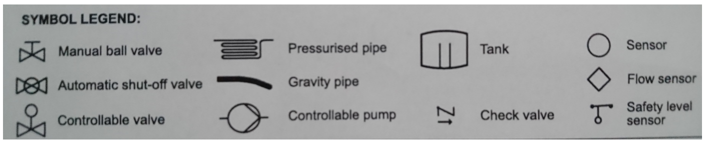

Figure 5.1: Meaning of symbols in SWIL module schematics [22]

The pumps are controlled by setting rotor speed between 0-100%. Pumps P4, P5 and P6 are responsible for recycling the water back into the system. Pumps P1, P2 and P3 supply the pipe and consumer units with water flow, see figure 5.2.

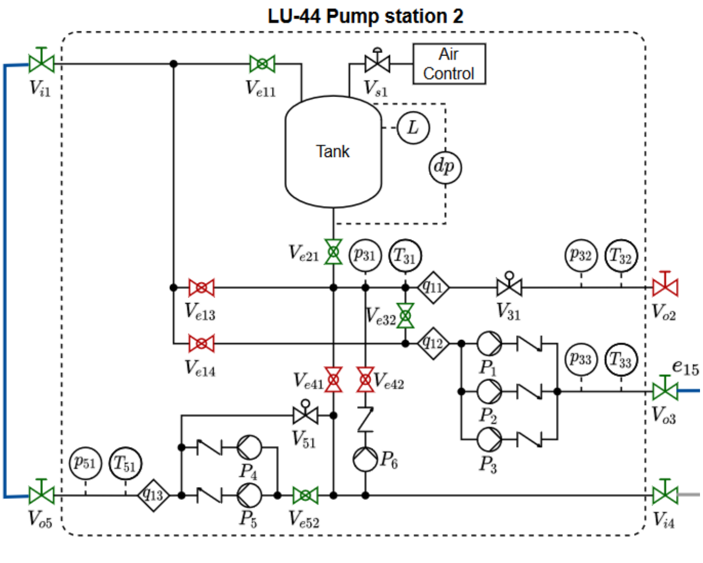

Figure 5.2: The schematic of a pumping station [22]

The pipe units in the network allow to change the pipe length and diameter as well as the possible setup of the water network, see figure 5.3.

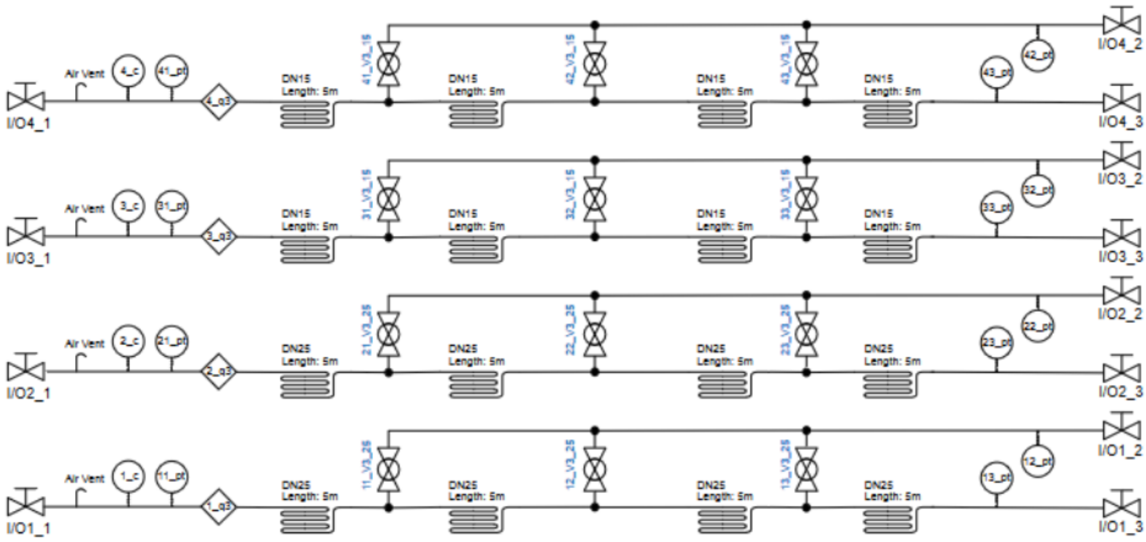

Figure 5.3: The schematic of a pipe unit [22]

Each pipe unit has 12 Input/Output valves. These valves are used to connect the pipe units with pumping stations, consumer stations or other pipe units. There are also 3 internal valves that change pipe length in each pipe unit. Pipe units also contain different variations of pipes, specifically with a diameter of 15 or 25.

The consumer units have controllable valves for each of the pipe unit connections. The controllable valves can be set to 0-100% open, see figure 5.4.

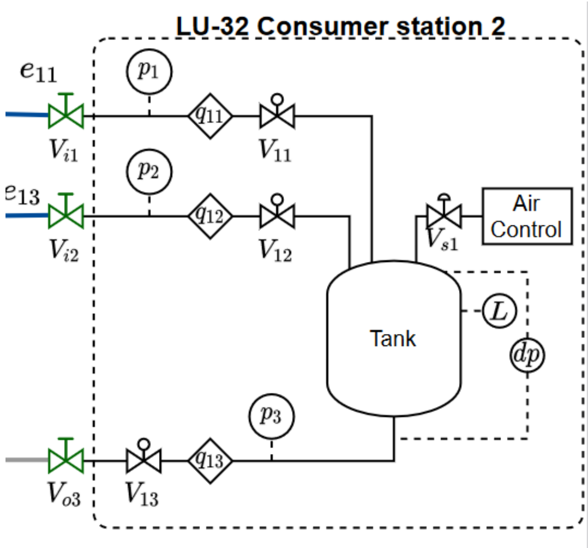

Figure 5.4: The schematic of a consumer station [22]

Two of the connections are for simulating water consumption with the water being stored in the consumer station tank. One connection is for recycling the water back into the system.

A simplified schematic of the lab setup is seen in figure 5.5.

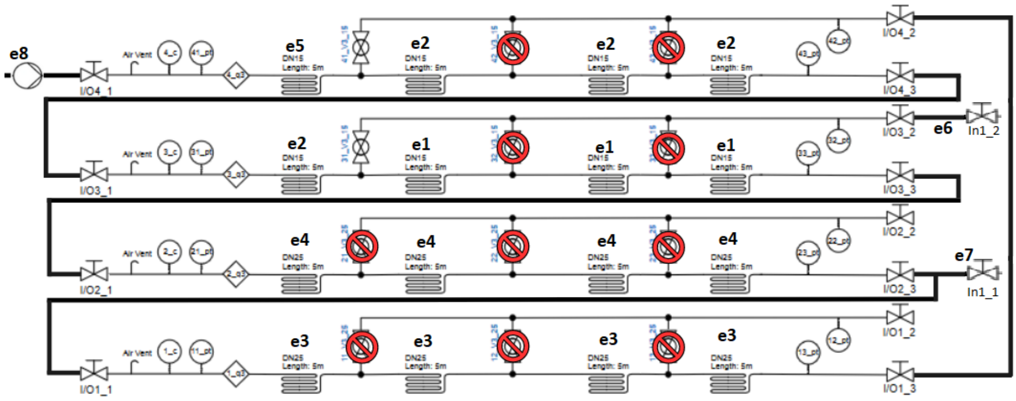

Figure 5.5: Simplified lab setup schematic with labeled edges. Valves In1_1 and In1_2 represent the consumers

Each module is used once for the lab setup. P3 in the pumping station is the only functional pump. For recycling water, P4 and P5 are turned on. See the complete schematic in Appendix A.1.

Chapter 6

Iterative learning pressure control

In this chapter, an Iterative Learning Control (ILC) is proposed for the WDN, where the end goal of this control scheme is to keep the pressure of the water in the consumers' ends at the desired pressure. In section 6.1 an overview of the ILC is presented which points to the advantage of using such control scheme, in section 6.2 an input update law is presented where it explains the control algorithm and in section 6.3 a stability analysis of the system controlled by the ILC is presented.

6.1 Overview of ILC

ILC can be classified as an intelligent control approach which is a method to enhance the transient performance of systems that function repetitively over a predetermined time interval[23].
In control theory, there is a wide range of control schemes that provide improvements to the response of a dynamic system but due to the presence of unmodeled dynamics or parametric uncertainties in real-world systems, it is not always possible to achieve the required desired performance[23]. Therefore, the advantage of using ILC when the system of interest operates repetitively is that it is not dependent on a model of the system[21].
The objective of the ILC is to iteratively design an input sequence for a system that performs the same task repetitively such that the output of the system is as close as possible to the desired output at all times[24][25][26][27].

6.2 Input update law

As mentioned in section 6.1 the objective of the ILC can be formulated in terms of the pressure at the measured node as follows,[21],

$$y_i(t) = r_i \tag{6.1}$$

where,
$y_i(t)$ is the pressure $p_i(t)$ at the **i**th measured node.
r_i is the desired reference pressure at **i**th node.

To control the pressure in the system the controller was placed at the inlet node of the network which in this project it is considered to be the pump specifically the rotational speed of the pump $u(t)$, the controller proposed in this project is feedforward ILC which was used in [21]. A basic configuration of feedforward ILC is presented in Fig. 6.1[23]. The controller gives a reference pressure r and forces the pressure at the outlet nodes $y(t)$

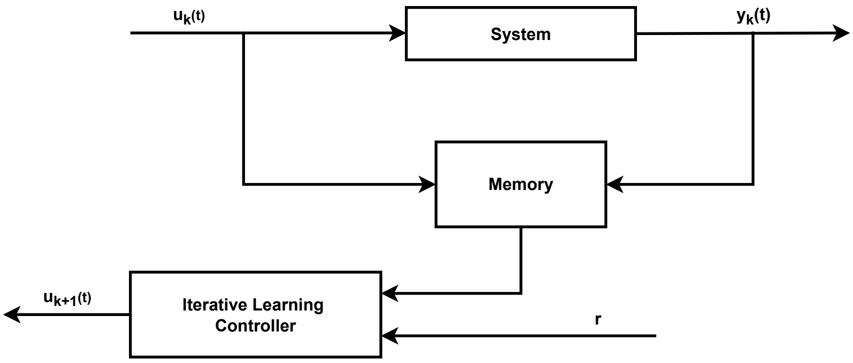

Figure 6.1: Basic ILC configuration

to approach the reference by updating the input speed sequence $u(t)$ after each iteration, to formulate this suppose that the system operates on a finite horizon given by $t \in [0, N]$ thus each iteration domain consists of a finite number of time points which can be written as follows,[23]

$$u_k = (u_k(0), u_k(1), \cdots, u_k(N-1)) \tag{6.2}$$

$$y_k = (y_k(0), y_k(1), \cdots, y_k(N-1)) \tag{6.3}$$

$$e_k = r - y_k = (e_k(0), e_k(1), \cdots, e_k(N-1)) \tag{6.4}$$

With these vectors the update law can be written as,[21]

$$u_{k+1} = u_k + Ke_k \tag{6.5}$$

Where,
u_{k+1} is the control parameter vector in $k+1st$ period and the initial control parameter u_0 is either set to zero or initialized appropriately by the system characteristics.[25]
K is constant learning gain.
e_k is the error which is the signal representing tracking accuracy on iteration k.

Equation (6.4) indicates the error computed from a single output, for multiple outputs as in this project the max function was used to obtain the error as in Eq. 6.6

$$e_k = \max\{\overline{r}(t) - \overline{y}_k(t)\} \tag{6.6}$$

Where, $\overline{r}(t)$ is the vector of reference pressures for each output in the system at time interval t.
$\overline{y}_k(t)$ is the vector containing the measurements from all outputs at time interval t of iteration k.

6.3 Stability analysis

The stability for the systems using ILC can be associated to the convergence by raising the following typical question:

Will the error approach zero as the number of iterations grow?[28]

For the stability to be well defined a few assumptions on the system controlled by ILC are given,[28]

Assumption 1

Every iteration k ends in a fixed time $N \in t$.

Assumption 2

A desired output r is given a priori over the time interval $t \in [0, N]$.

Assumption 3

Every output $y_k(t)$ can be measured and therefore the tracking error $e_k(t) = r - y_k(t)$ can be utilized in the calculation of $u_{k+1}(t)$.
In this section, the stability analysis was conducted on the static model as well as the dynamic model to choose the right gain K which will fulfill the error convergence of the system and therefore makes the ILC stable.

In static model the pressure Eq. 4.67 at **k**th period is given below,

$$p_{i,k}(t) = \alpha_i d_n^2 - \gamma_i + u_k(t) \tag{6.7}$$

The ILC error track is[25],

$$e_{k+1} = r - p_{i,k+1}(t) = r - \alpha_i d_n^2 + \gamma_i - u_{k+1}(t) \tag{6.8}$$

According to Eq. 6.5 the controller in Eq. 6.8 is written as follows,

$$e_{k+1} = r - u_k - Ke_k - \alpha_i d_n^2 + \gamma_i \tag{6.9}$$

From Eq. 6.7 the following is given,

$$u_k(t) = p_{i,k}(t) - \alpha_i d_n^2 + \gamma_i \tag{6.10}$$

Using this in Eq. 6.9 results the following,

$$e_{k+1} = r - p_{i,k}(t) + \alpha_i d_n^2 - \gamma_i - Ke_k - \alpha_i d_n^2 + \gamma_i \tag{6.11}$$

Since α_i and γ_i are constant and d_n^2 is a periodic term and it is invariant over the period then they cancel out each other, and the error tracking will be as follows,

$$e_{k+1} = e_k - Ke_k = (1-K)e_k \tag{6.12}$$

Where $e_k = r - p_{i,k}(t)$ from Eq. 6.11 The Eq. 6.12 is used in the convergence theorem which states that the $||.||_\infty$ of the ratio between the next error and the current error should be less that ϵ and the ratio between the errors is given as $\frac{e_{k+1}}{e_k} = ||(1-K)||$
In a mathematical form the convergence is given as,

$$||1-K|| \leq \epsilon < 1 \tag{6.13}$$

From Eq. 6.13 it is clear that the error convergence in the static model holds if $0 < K < 1$

The steps mentioned above in the static model are also used in the dynamical model for the control gain calculation.
The output pressure Eq. 4.78 at **k**th period is used,

$$p_{i,k}(t) = (\overline{H}_T^{-T}\mathcal{J}_T\dot{q}_T)_i + (\overline{H}_T^{-T}(\lambda_T(q_T) + \mu_T(q_T, OD_T) - \alpha_T(q_T, \omega_{T_k})))_i - (\overline{z} - \mathbb{1}z_o)_i + \mathbb{1}p_o \tag{6.14}$$

The equation is simplified into the following equation,

$$p_{i,k}(t) = D_i - Au_k - \gamma_i + \mathbb{1}p_o \tag{6.15}$$

Where,
$D_i = (\overline{H}_T^{-T}\mathcal{J}_T\dot{q}_T)_i + (\overline{H}_T^{-T}(\lambda_T(q_T) + \mu_T(q_T, OD_T) - \alpha_T(q_T)))_i$
$\alpha_T(q_T) = -a_{h2}|q_T|q_T$
$A = \overline{H}_T^{-T}a_{h0}$
$\omega_{T_k} = u_k$
$\gamma_i = (\overline{z} - \mathbb{1}z_o)_i$
The subscript $(\cdot)_i$ means the ith row of the matrix $(\cdot)$
By assuming that the water network is in steady state most of the time, the fast dynamics in the system can be neglected by setting $\dot{q_T} = 0$ then the
$D_i = (\overline{H}_T^{-T}(\lambda_T(q_T) + \mu_T(q_T, OD_T) - \alpha_T(q_T)))_i$
Where q_T can be taken as a constant according to the assumption above which makes the term D a constant term.
Then the ILC error tracking is,

$$e_{k+1} = r - p_{i,k+1}(t) = r - D_i + Au_{k+1} + \gamma_i - \mathbb{1}p_o \tag{6.16}$$

From Eq. 6.5 the update law is,

$$Au_{k+1} = Au_k + AKe_k \tag{6.17}$$

Using Eq. 6.17 in Eq. 6.16 results the following,

$$e_{k+1} = r - D_i + Au_k + AKe_k + \gamma_i - \mathbb{1}p_o \tag{6.18}$$

The controller term in Eq. 6.18 which is found from Eq. 6.15 is,

$$Au_k = -p_i(t) + D_i - \gamma_i + \mathbb{1}p_o \tag{6.19}$$

Using Eq. 6.19 the following equation is found,

$$e_{k+1} = r - D_i - p_i(t) + D_i - \gamma_i + \mathbb{1}p_o + AKe_k + \gamma_i - \mathbb{1}p_o \tag{6.20}$$

Assuming the terms D_i, γ_i and $\mathbb{1}p_o$ are constant the error track equation is expressed as follows,

$$e_{k+1} = e_k + AKe_k \tag{6.21}$$

Using the error tracking equation the gain can be determined by fulfilling the convergence theorem as shown below,

$$||1 - AK|| = ||1 - \overline{H}_T^{-T} a_{h0} K|| \leq \epsilon < 1 \tag{6.22}$$

Using the triangle inequality, Eq. 6.22 can be written as,

$$||1 - \overline{H}_T^{-T} a_{h0} K|| \leq ||1|| - ||\overline{H}_T^{-T} a_{h0} K|| \leq \epsilon < 1 \tag{6.23}$$

Therefore, for the ILC in the dynamic model to be stable it should hold $0 < ||\overline{H}_T^{-T} a_{h0} K|| < 1$

Chapter 7

Tests

In this chapter simulation tests of ILC applied on several versions of the models derived in Chapter 4 are presented as well as the network layout in graph representations of the models are presented. In Section 7.1 to Section 7.6 several simulation scenarios of the static model and dynamic model are presented, in Section 7.7 and Section 7.8 lab tests of the WDN controlled by the ILC carried out in the AAU SWIL are presented.

7.1 First static model

Description

This was the very first iteration model used to familiarize with how to set up the static model with a simple controller. This test was meant to see how the simplest iterative learning controller works on a sinusoidal wave. The sinusoidal wave in Fig. 7.1 is represented by using *cos* wave function with modifications that are represented such as: $65 + 35 * cos(\frac{9}{100} * pi * (t - 6))$. Value for t is a period from 1 to 24 which is represented in hours to reflect the entire day.

Procedure

After initial variables are initialized, the simulation goes through two *for* loops. The first loop represents how many days the iterative learning controller will run and update the sinusoidal wave accordingly to the error found in the second loop. The second loop is used to find pressure values, which then are compared to the reference values, in this case, the reference is equal to 0.2, to find the error value to be used in the first loop. By running this simulation for 4 days, the wanted reference was reached without any overshooting or undershooting reference value.

Results

Once simulation begins, the very first figure 7.1 is generated which shows the beginning of sinusoidal wave before any alterations from iterative learning controller.

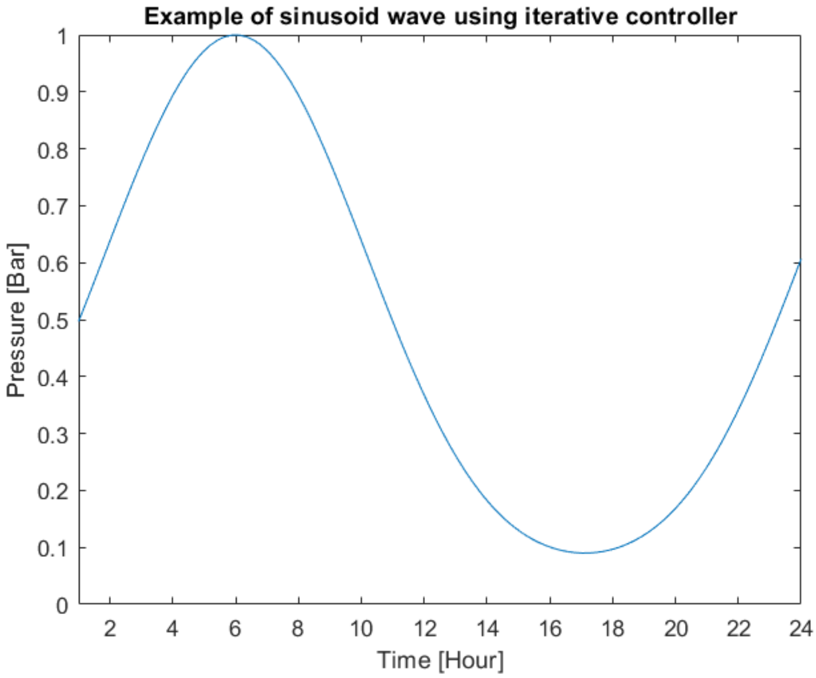

Figure 7.1: Cos sinusoidal wave before iterative controller changes

After running for 4 days, the sinusoidal wave was modified to reach the reference value of 0.2, which shows that the most basic example of ILC on a sinusoidal wave was achieved the wanted results. Such results are represented in Fig. 7.2.

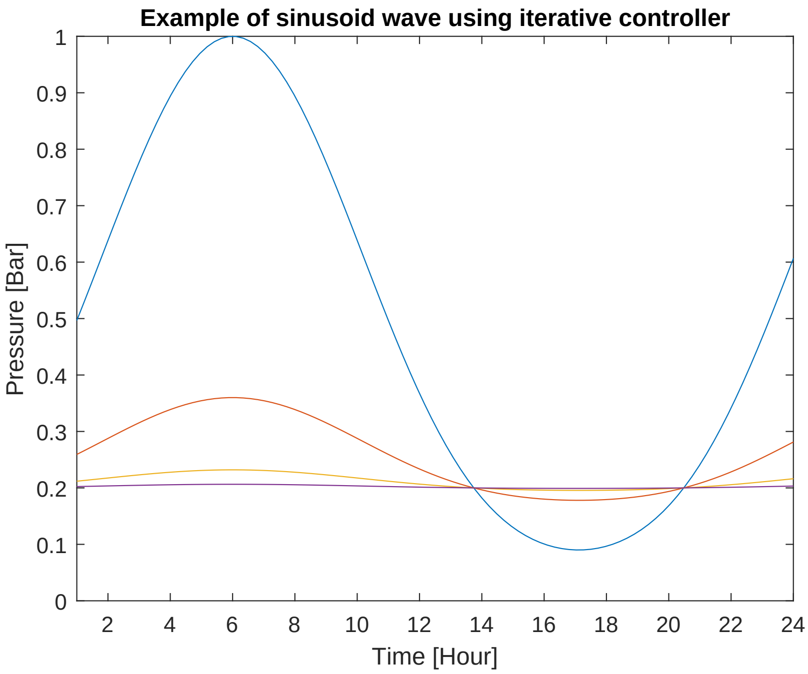

Figure 7.2: First working iteration of basic sinusoidal type static controller

7.2 Behaviour of consumers

In order to utilize the consumer patterns found in Fig. 2.6, the behavior of such patterns will be simplified and unified into a single figure, which represents the total consumption throughout the entire day. This behavior will then be used in both simulations: static model and dynamic model. The behavior in Fig. 7.3 will be used to simulate consumer valve openings throughout the entire day which will be used in simulations to generate the graphs.

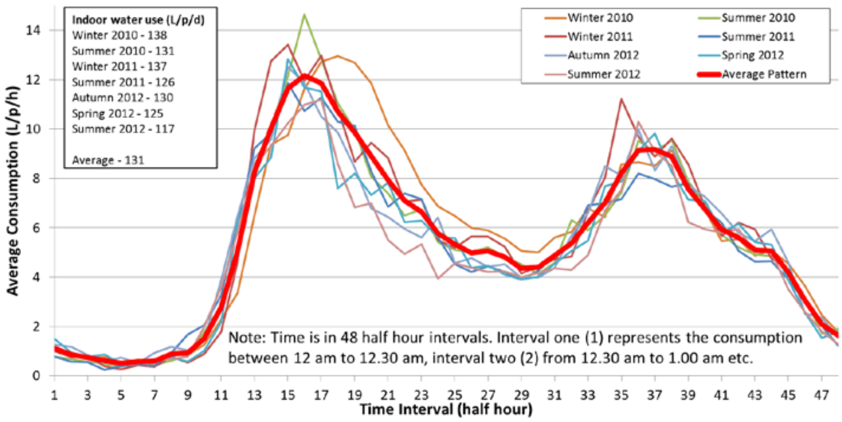

Figure 7.3: The average indoor water consumption through out a day[29]

7.3 Static model

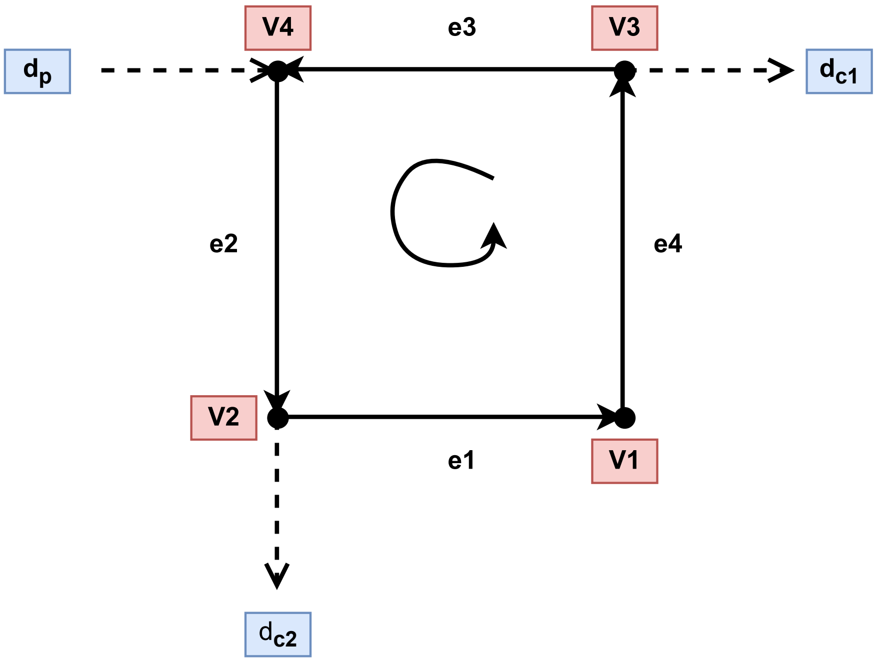

Figure 7.4: Reduced order graph of the static simulation model

$$H = \begin{bmatrix} -1 & 0 & 0 & 1 \\ 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & -1 \\ 0 & 1 & -1 & 0 \end{bmatrix} \tag{7.1}$$

$$\overline{H} = \begin{bmatrix} 1 & -1 & 0 & 0 \\ 0 & 1 & -1 & 0 \\ -1 & 0 & 0 & 1 \end{bmatrix} \tag{7.2}$$

$$B = \begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix} \tag{7.3}$$

In this section the static model of the water network in Fig. 7.4 was simulated using equations from section 4.2.3 using MATLAB. Since the model is static, the valve characteristics are disregarded and only the valve opening is set to follow the behavior of the average daily water consumption. In the following, two tests will be conducted. The first test will only have a single consumer while the second test will have two consumers. In both tests, the consumers mimic the behavior of the average daily water consumption found in Fig. 7.3.

7.3.1 Static model using one consumer

Description

This test will show how the pressure changes over time using ILC when the system only contains a single consumer. The consumer will be assigned values from the total consumption graph throughout the day shown in the Fig. 7.5, which then is repeated periodically for 3 days. The pressure graph will be updated for each iteration the controller makes and plots the results on top of the existing plot, such that changes to the plot can be seen by iterations.

Procedure

First, all the variables are declared including pipe dynamics, which then is used in the minimization problem to find the lowest possible a_c value. This value is then used to find the values for $function_g$. Then $function_g$ is used to find the pressure values that are used to find the error values. These error values will then be used to affect the next iteration control values. After each iteration, new plots will be added on top of old ones, to have all iterations in a single figure to see how they affected the simulation in general.

Results

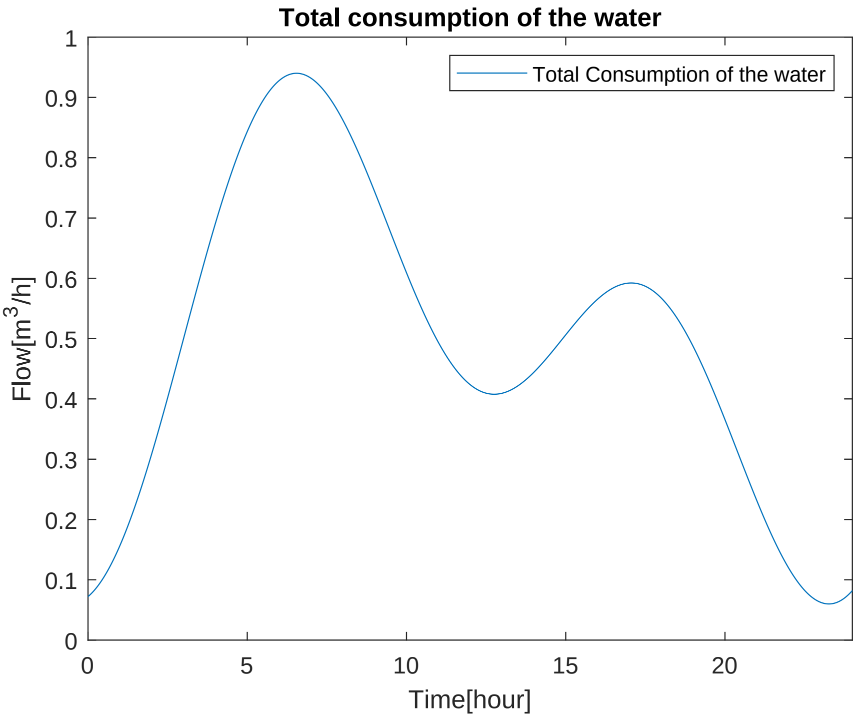

Figure 7.5: Total consumption of the water for singular consumer

The very first iteration shows quite much progress compared to the initial values of the consumer pressure, this might be the case due to having the gain value be initialized as 0.9. In this case, just in 3 days, the iterative learning controller managed to reach the required reference of 0.2 bars of pressure which is shown in Fig. 7.6.

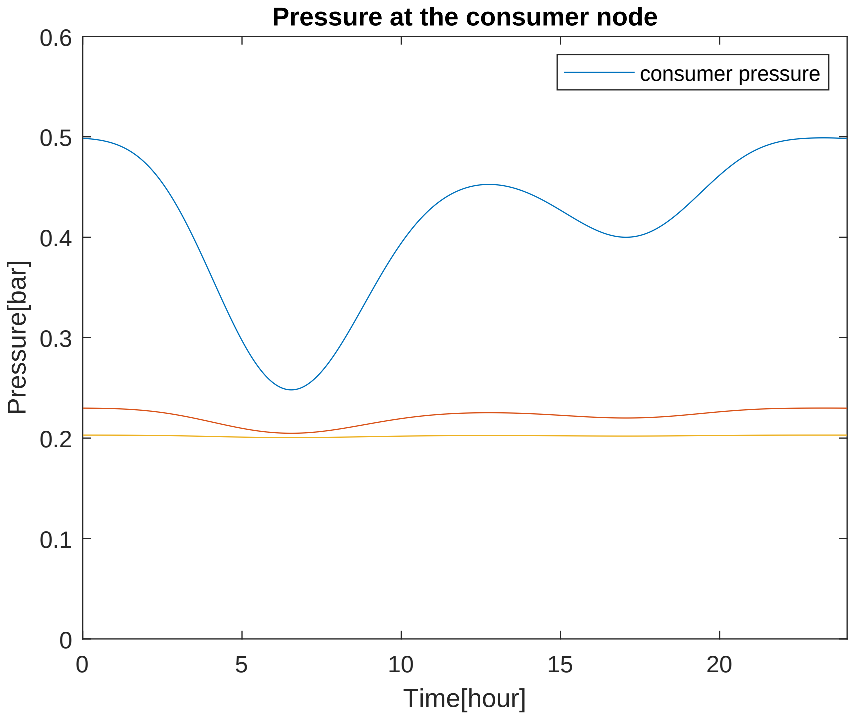

Figure 7.6: Pressure graph after using iterative learning controller for 3 days

7.3.2 Static model using two consumers

Description

Similar setup as the test with one consumer, the only difference being that now there are 2 consumers. Each consumer in this case has a different weight for the total consumption of the water. The first consumer's weight is set up to be 0.3 of total consumption, and the second consumer is at 0.7 of total consumption. The weights are added in a way, that the sum of the weights, needs to be 1 in order to reflect the same total consumption curve as in Fig. 7.3. The iterative learning controller will run for 3 days with a gain of 0.9 and a reference of 0.2.

Procedure

After initializing all required values for both consumers, the minimization problem will be solved in order to find the lowest possible value of a_c. This a_c is then used to find $function_g$, then it will be used to find pressure values. The only difference between using one and two consumers is in the error finding function. Since now the error is generated as a vector, *max* function needs to be introduced to find the biggest error value, which will be used to reflect upon the next iteration control signal. This will repeat for a total of 3 times to reflect 3 days of the real world and will be represented in the Fig. 7.8.

Results

The Fig. 7.7 shows the graph for both consumers after applying weight scalars on each of the consumer.

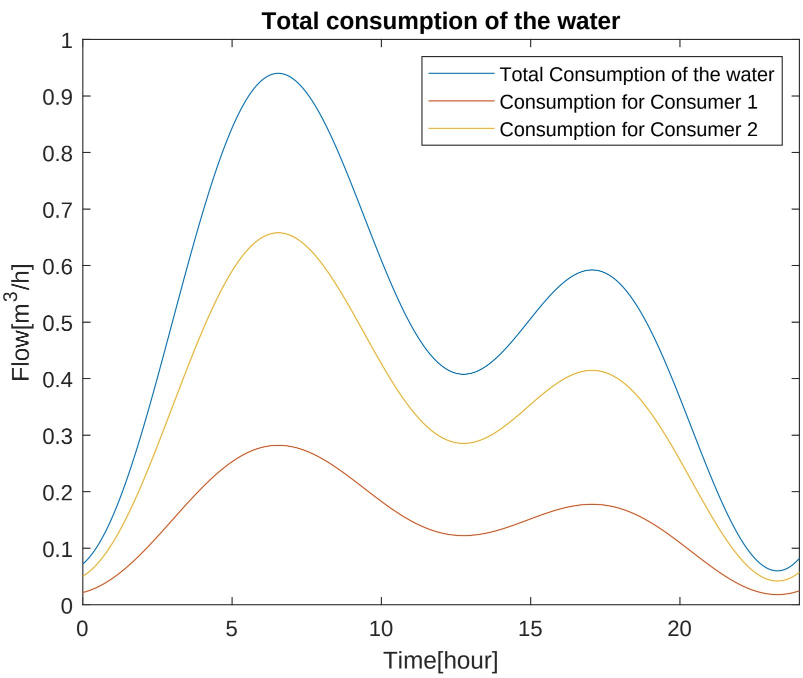

Figure 7.7: Total consumption of the water through entire day with two consumers

After 3 iterations, the following Fig. 7.8 is generated and can be seen that in 3 iterations, it managed to reach the required reference pressure.

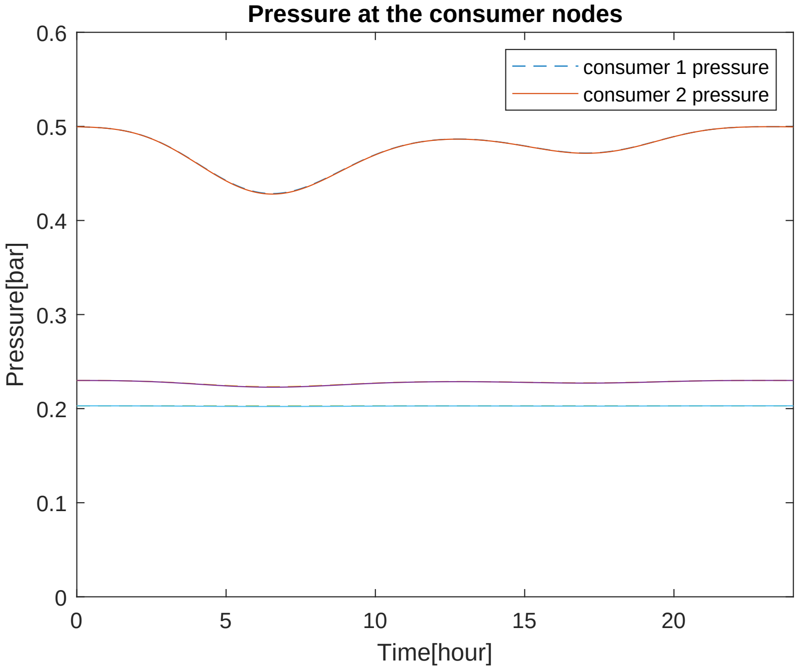

Figure 7.8: Pressure at the consumer nodes after iterative controller

Due to scaling, it is hard to see both consumers in a single graph, so zoomed in version of last iteration was made into the graph, see Fig. 7.9, for easier representation where each consumer pressure ends up at.

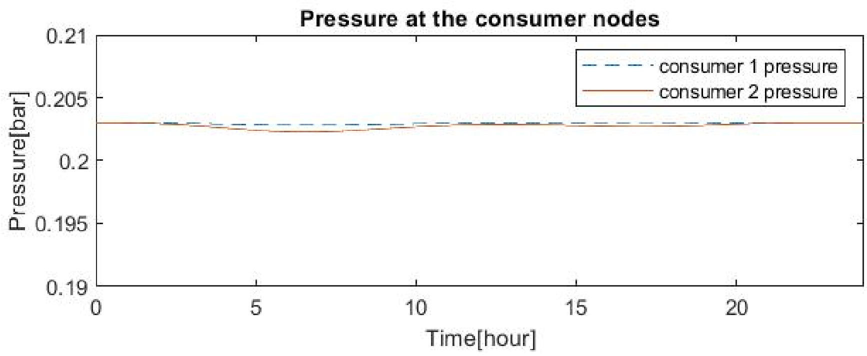

Figure 7.9: Zoomed in version for the last iteration with two consumers

7.4 Dynamic model

In this section the dynamic model was simulated using the graph representation illustrated in Fig. 7.10 the incidence matrix of the graph was obtained in Eq. 7.4 the reference node for this network was chosen to be node v_7 and the reduced incidence matrix Eq. 7.5 was obtained by removing the last row which represent the reference node v_7, the loop matrix for the network is derived in Eq. 7.6

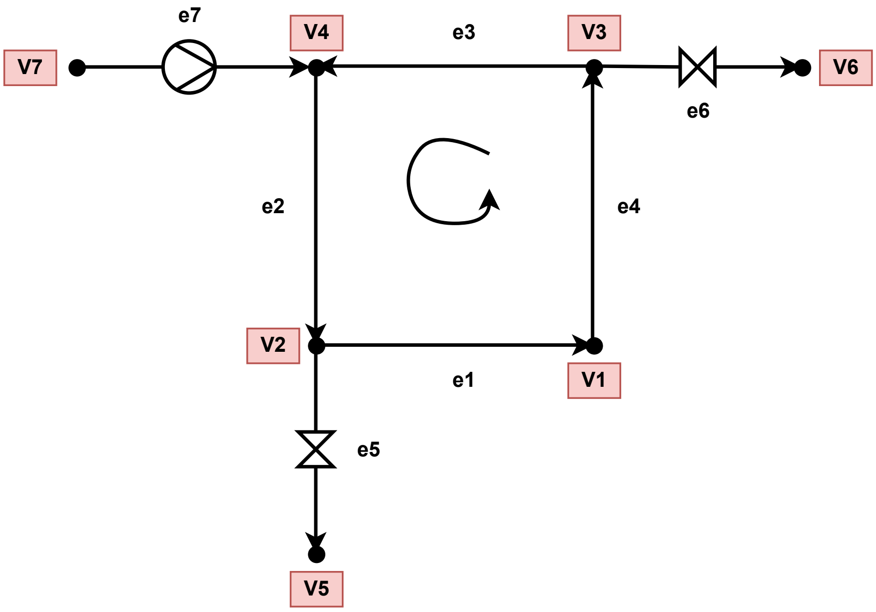

Figure 7.10: Graph representation of the dynamic model

The incidence matrix H of the dynamic model:

$$H = \begin{bmatrix} -1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 1 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{7.4}$$

The reduced incidence matrix $\overline{H}$ of the dynamic model:

$$\overline{H} = \begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 \end{bmatrix} \tag{7.5}$$

The loop matrix B of the dynamic model:

$$B = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 \end{bmatrix} \tag{7.6}$$

The dynamic model was simulated from the equations in section4.2.4 and section4.2.1 using MATLAB, unlike the static model the valve characteristics are taken into account in the dynamic model where the consumer's valve opening is set to mimic the behavior of the average daily water consumption in a residential building as shown in Fig. 7.3. The following tables are showing the parameters that were used in the different components of the network for simulating the dynamic model.

Valve	$K_{vs}[\frac{m^3}{h}]$
e_5	1
e_6	1

(a) Valve parameters

Pump	a_{h2}	a_{h0}
e_7	0.0367	$7.335 \cdot 10^{-5}$

(b) Pump parameters

Pipe	Length[m]	Diameter[mm]
e_1	20	25
e_2	20	25
e_3	20	15
e_4	20	15

(a) Pipe parameters

Node	Elevation[m]
v_1	1
v_2	1
v_3	1
v_4	1
v_5	1
v_6	1
v_7	1

(b) Elevation of the nodes in the network

The control signal in this model is the angular speed of the impeller blades of the pump and in this case is fixed to run at full speed 100%.

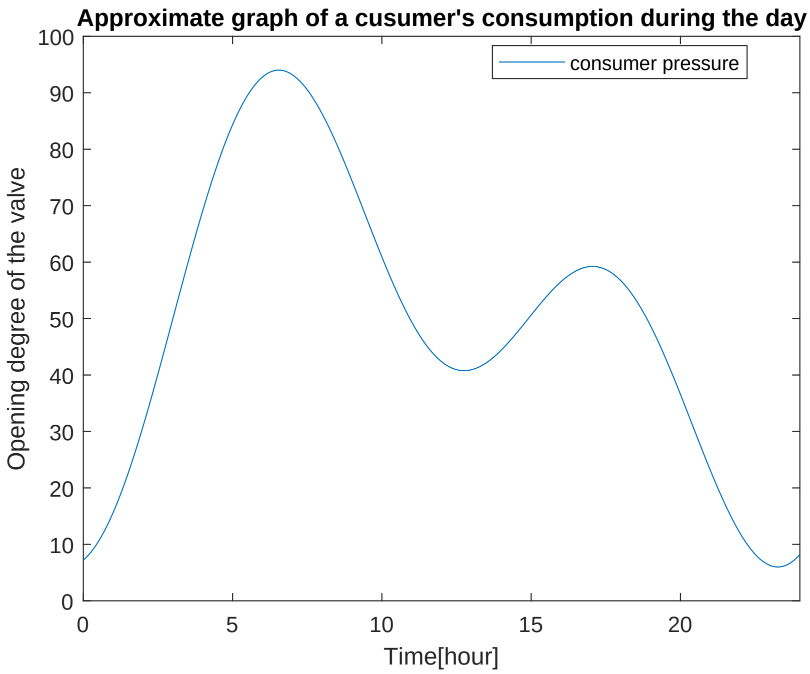

Figure 7.11: A simulated opening degrees of the valve that flows the consumers' water consumption during 24 hours

The curve in Fig. 7.11 shows that the valve opening degree is in the interval $(0, 100)$, to as shown in the graph the curve has two peaks one in the morning around 7 hour is the highest consumption of the water and another around 18 hour, this graph was constructed by adding two sinusoidal waves.

7.4.1 Results

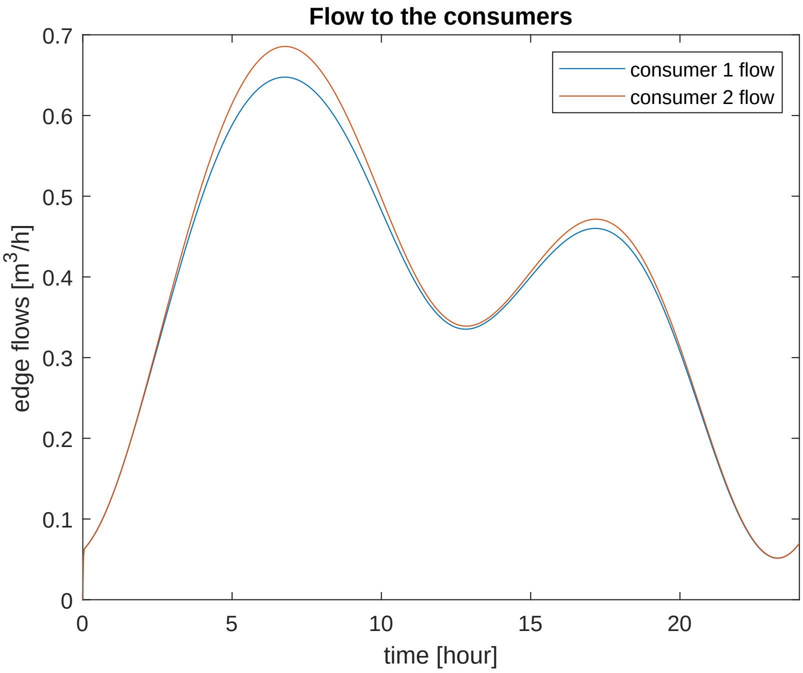

Figure 7.12: Flow to the consumers

The Fig. 7.12 shows the flow to consumers and as expected the changes in the valve opening degrees greatly effects how the flow should behave throughout the simulation, where around 6 o'clock the flow to the first consumer end node v_2 is $0.63m^3/h$ and the flow to the second consumer end node v_3 is $0.68m^3/h$.

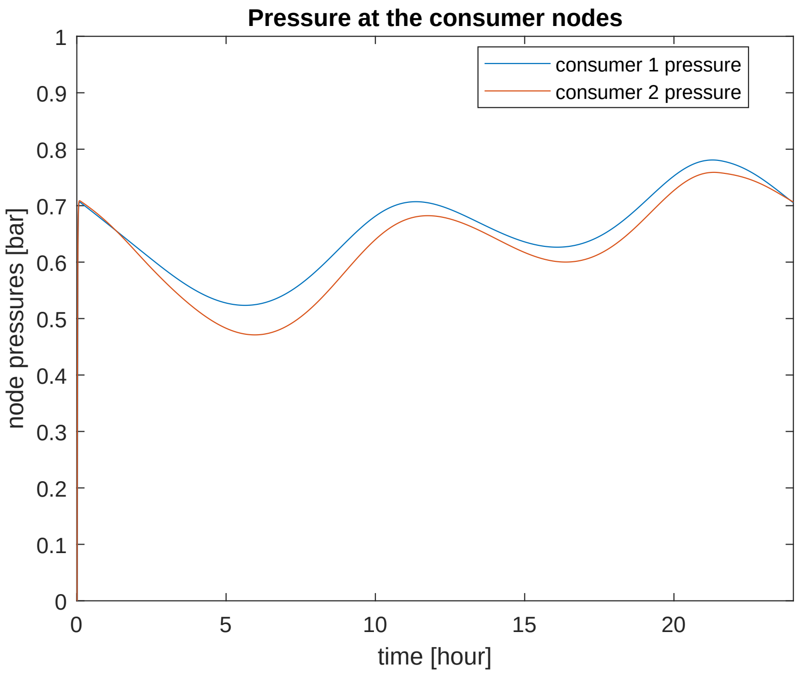

Figure 7.13: Pressure at the consumer nodes

The pressures at the consumers' ends are shown in Fig. 7.13, the control signal in this simulation has been taken to be a constant value of $u = 100$ which means 100% of the pump speed. In this graph the relationship between the pressure and the valve opening degrees can be seen, meaning that when the opening degree of the valve in one node is too low the pressure at that node will increase and when the valve is almost fully open the pressure will drop, this behavior can be clearly seen from Fig. 7.13 and Fig. 7.11

7.5 Dynamic model with one consumer

In this section, the dynamic model was simulated and the ILC derived in section 6.2 was applied to the dynamic model, the network in this section was structured to have one consumer where the consumer edge $e6$ was removed from the dynamic model network shown in Fig. 7.10 and the ILC was applied to control the pressure of that consumer, the properties of the consumer's valve was the same as in section 7.4.
The simulation was set to run 24 hours each iteration where after each iteration the ILC will produce a new control sequence to use in the next iteration, this simulation was run for 4 iterations to produce 4 different pressure sequences measurements and 3 control sequences as will be shown in the following section.

7.5.1 Results

The results of simulating the dynamic model with one consumer are presented in this section, the results are divided into three parts:

1. The flow to the consumer.
2. The input sequence or control sequence for each iteration.
3. The pressure measurement at the consumer end after each control sequence.

The flow to the consumer node in this simulation can be seen in Fig. 7.14, it can be observed that both the valve opening degrees and the control applied in the system are effecting the flow of the water during the day.

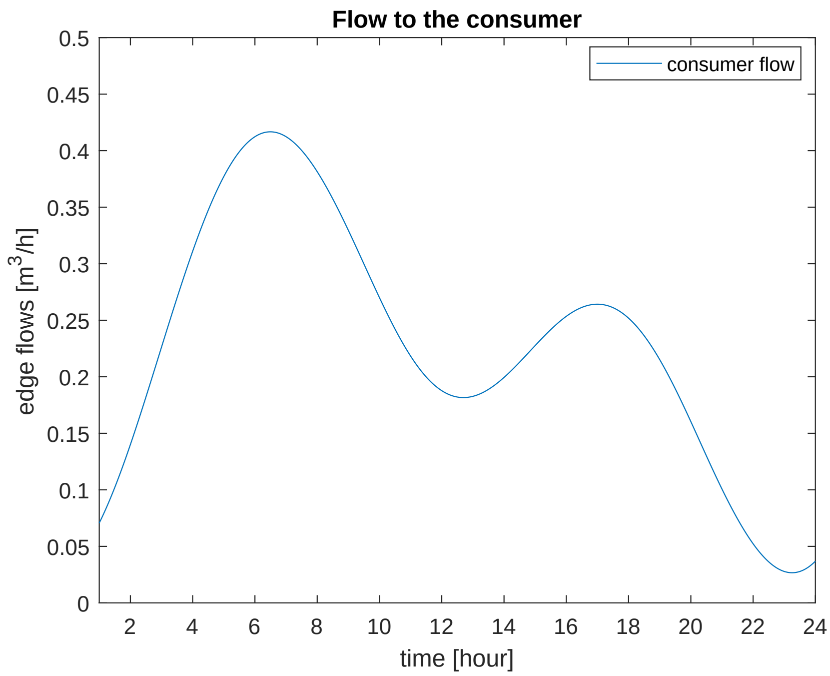

Figure 7.14: Flow to the consumer

The control sequence for the network which is the speed of the pump for the first iteration of the simulation is given to be 100% of the pump speed, then the ILC updates the control sequence every iteration using the previous control sequence and a gain of $K = 0.16$ multiplied with the error measurement of the previous iteration. These control sequences are shown in Fig. 7.15 where the blue line is the control sequence for the second iteration and the red line is the control sequence for the third iteration.

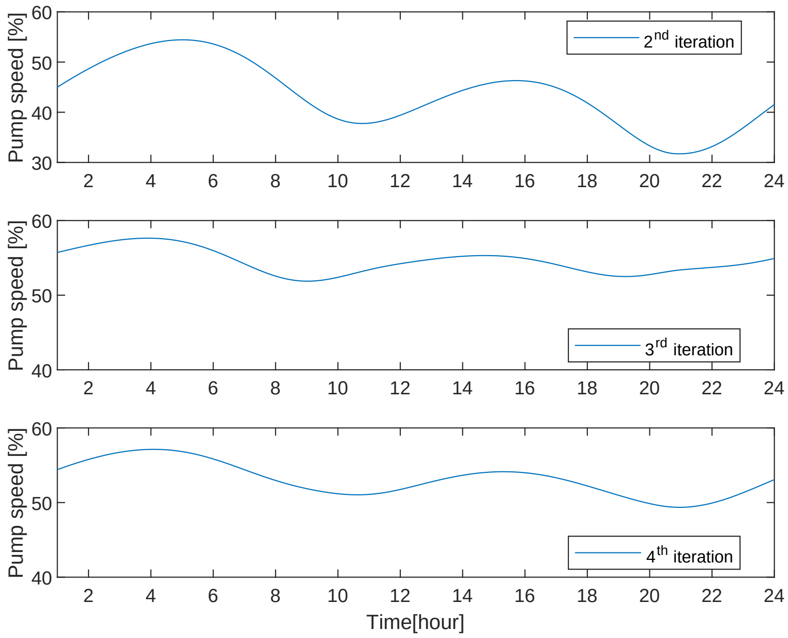

Figure 7.15: Control sequence for the 2^{nd} and 3^{rd} iterations of the simulation

Since The ILC updates the pump speed every 24 hours which is equivalent to one iteration, the pressure measurements for the consumer can be seen in Fig. 7.16. At the start of the simulation, the initial control sequence was applied to the network which gave the pressure measurements in the blue line, where the red line is the pressure of the second iteration and the yellow line is the pressure of the third iteration.

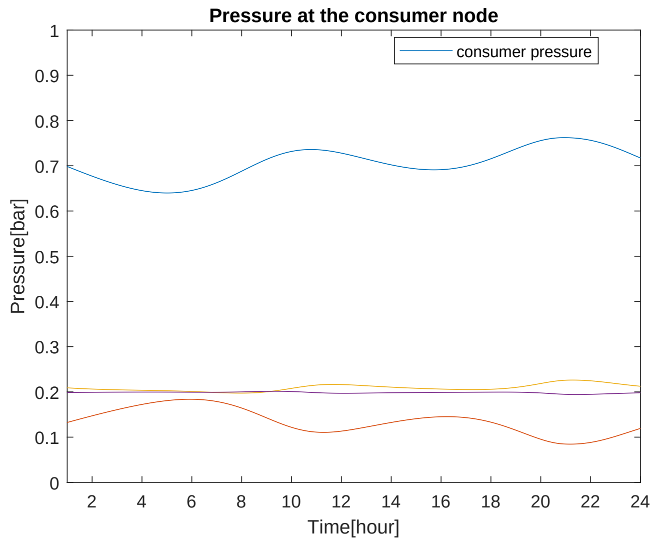

Figure 7.16: Pressure at the consumer node

The 4^{th} iteration of the pressure measurements can be seen in Fig. 7.17, which shows that the ILC managed to bring the pressure at the consumer end from 0.7 bar to 0.2 bar in four iterations.

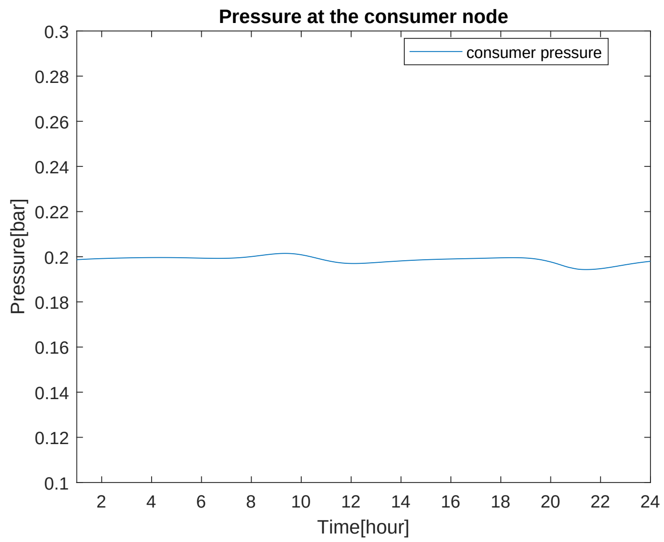

Figure 7.17: the 4^{th} iteration of pressure at the consumer node

7.6 Dynamic model with two consumer

In this section the second consumer has been added to the network with the same characteristics as the first consumer in terms of valve opening degrees.
Since the network has two consumers in this simulation the ILC updates the speed of the pump by taking the largest value in the error vector as shown in Eq. 7.7.

$$e_k(t) = max(\bar{r} - \bar{p}_k(t)) \tag{7.7}$$

Where $\bar{p}_k(t)$ is a vector consisting of the pressure from the two consumers and $\bar{r}$ is the vector of the reference pressure for each consumer.

7.6.1 Results

In this section, the results of the simulation of the dynamic model network with two consumers are presented. The results are shown in terms of:

1. The flow to the consumer.
2. The input sequence or control sequence for each iteration.
3. The pressure measurement at the consumers' ends after each control sequence.

As seen in section 7.5.1 the curve of flow to the consumer end nodes follows the shape of the valve opening degree since the opening degrees dictates how much flow should go through the valve, the flow to the consumers is shown in Fig. 7.18

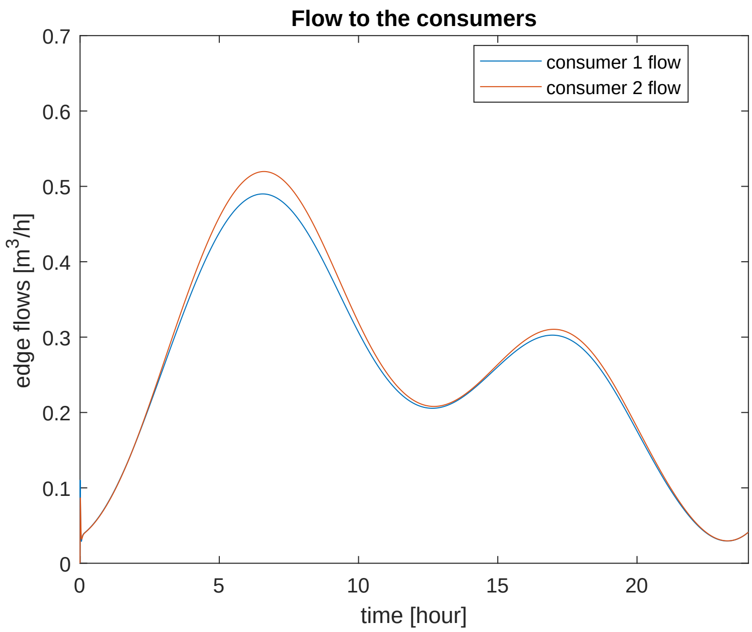

Figure 7.18: Flow to the consumers

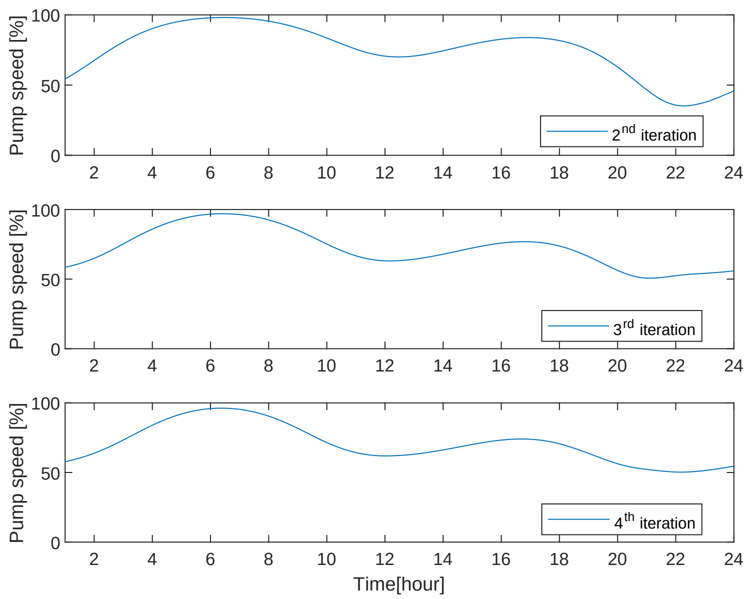

Figure 7.19: Control sequence for the 2^{nd}, 3^{rd} and 4^{th} iterations of the simulation

The initial control sequence for the network is given to be 100% of the pump speed, and the simulation of this network was set to run for 4 iterations (4 days) the control sequence obtained by the ILC for each iteration is shown in Fig. 7.19, the control sequence for each iteration was obtained from the previous control sequence and a gain of $K = 0.16$.
The significant change in the control sequences can be notice from the first iteration where the control sequence was 100 throughout the entire iteration and the fourth iteration where the pressure at the consumers end approach the reference as will shown in the next figures.

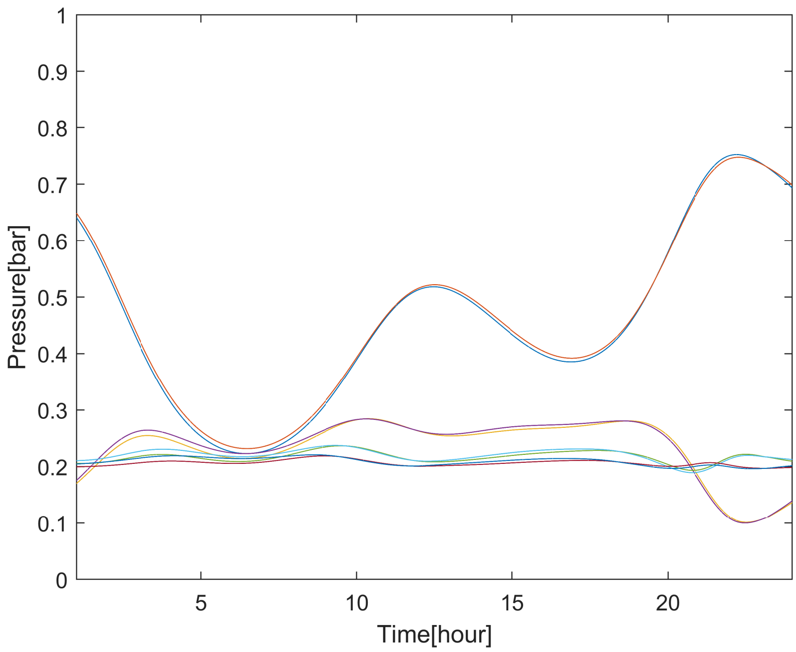

Figure 7.20: Pressure at the consumer nodes

The pressure at the consumer end nodes for all four iterations is shown in Fig. 7.20, the pressure for each consumer starts around 0.65 bar for the first iteration and it keeps approaching the reference pressure 0.2 bar. In Fig. 7.21 the pressures of the 4^{th} iteration in the simulation were shown, from that figure it is obvious that the ILC applied on the dynamic model network with two consumers is forcing the pressure at the consumers' ends to follow the reference pressure.

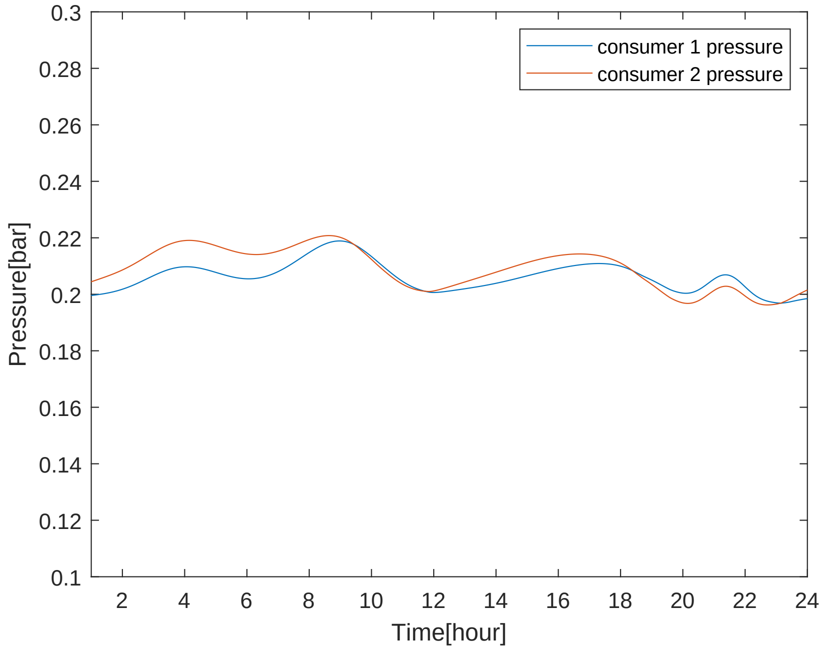

Figure 7.21: 4^{th} iteration of pressures at the consumer end nodes

7.7 Lab test 1

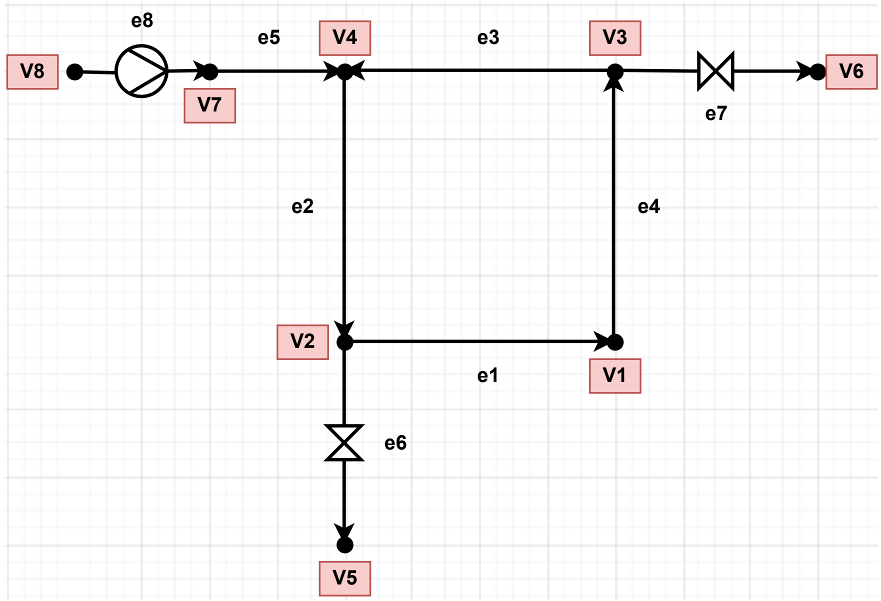

Figure 7.22: Graph representation of the network used in SWIL tests

This section deals with the water network model in the AAU water lab. The model consists of 1 pump described as *e8*, 2 consumers(valves) which are *e6* and *e7*, and 4 pipes (*e1*,*e2*,*e3*,*e4*). All of these components are constructed as shown in Fig. 7.22.

The model for the water lab setup was partially given as a template, which then was modified with an extra function block where the iterative learning controller resided.

The lab test was set up to run an indefinite amount of time since ILC on the real system is quite slow and because of that, each iteration for the control to change is made only every 48 minutes. This means that every 2 min, the simulation changes the control value from the vector of controls generated over a single iteration. After all 24 values are generated, the control vector will get updated with the new values, and those values will be used for the next iteration of the control.

The valve opening degrees were set to be 80% for the first consumer and 50% for the second consumer and both consumers were set to be static values. This means that even when the simulation runs for a long time, the consumer opening degree will not change. The initial pump speed to apply for the initial entry was taken to be 75% of the full speed and the desired pressure for the system to reach was set to be 0.5 bar.

In this lab test, the following parameters for the pipes were used to construct the model.

Pipe	Length[m]	Diameter[mm]
e_1	15	13
e_2	20	13
e_3	20	25
e_4	20	25
e_5	5	13

Table 7.3: Pipe parameters for the model in the water lab

The first iteration of the pressure reading can be found in Fig. 7.23 and the first iteration control signal can be found in Fig. 7.24. From the control signal figure, it can be seen that the pressure is almost stable at 0.5 bar as the reference required. This happened due to the initial pump propeller speed initialized at 75% speed.

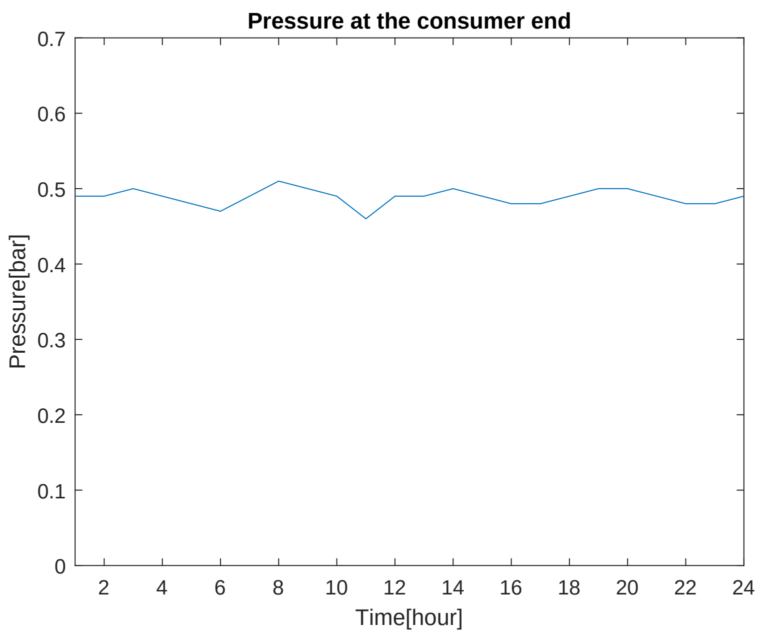

Figure 7.23: Pressure readings after first iteration of ILC in the lab

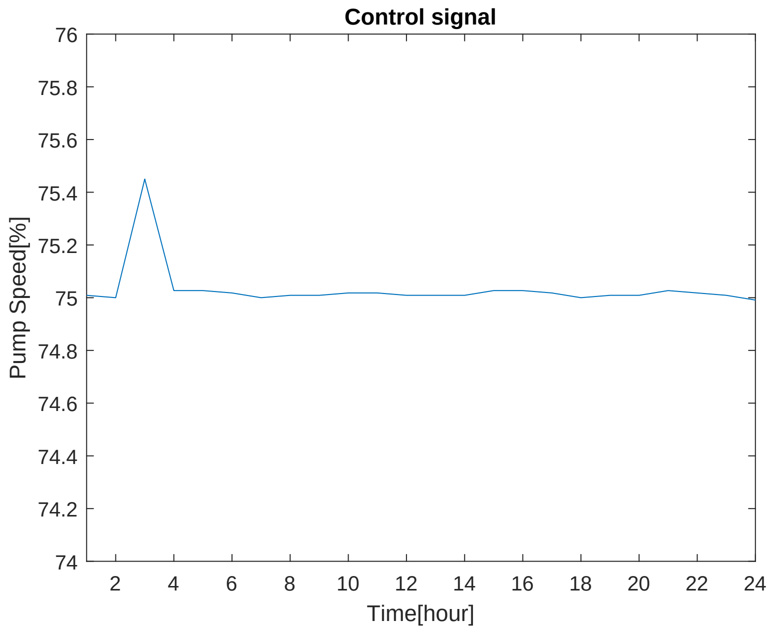

Figure 7.24: Control signal values for the pump after first iteration of ILC in the lab

After running the simulation for a couple of days, the latest iteration was chosen to see the changes that occurred compared to the very first iteration. The full last iteration that the simulation managed to run was 28^{th}. The pressure after 28^{th} iteration is shown in Fig. 7.25 and the control signal after 28^{th} iteration looks like this in Fig. 7.26.

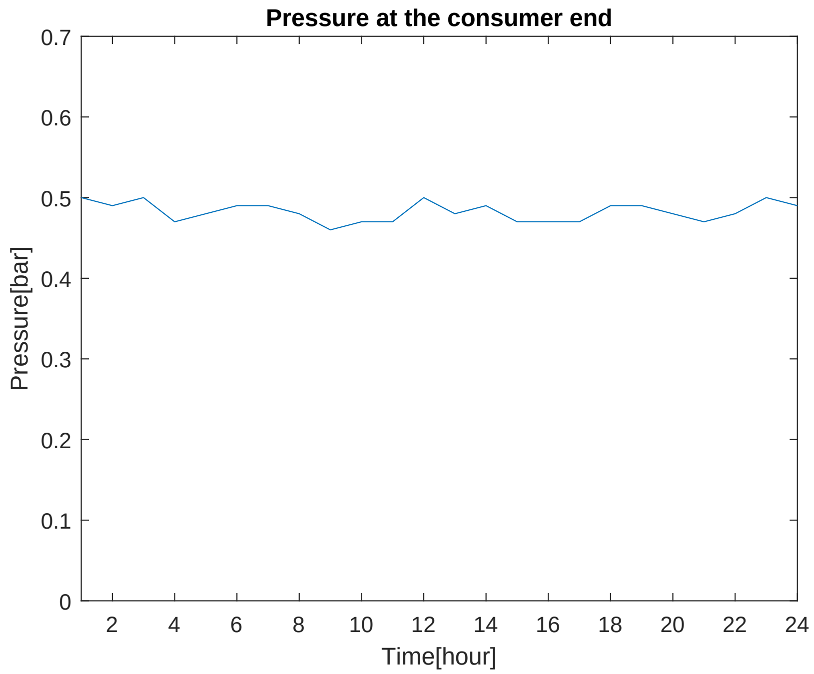

Figure 7.25: Pressure readings after 28^{th} iteration of ILC in the lab

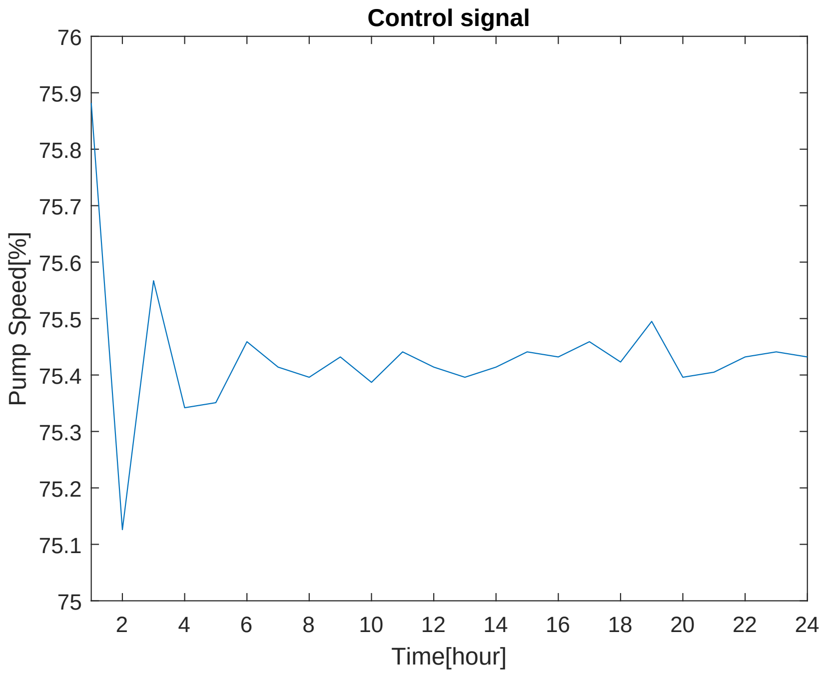

Figure 7.26: Control signal values for the pump after 28^{th} iteration of ILC in the lab

The differences between the iterations will be further discussed in the Discussion chapter.

7.8 Lab test 2

In this section a second lab test was conducted in the SWIL, the network used in this test has the same structure as the previous test, this simulation test was set to run 24 minutes each iteration and it ran for 113 iterations, the consumer end valves are set to open according to Fig. 7.27, this figure was designed by adding two sinusoidal waves with an amplitude of 1, a phase shift of 3 and a period T of 24 and 12.

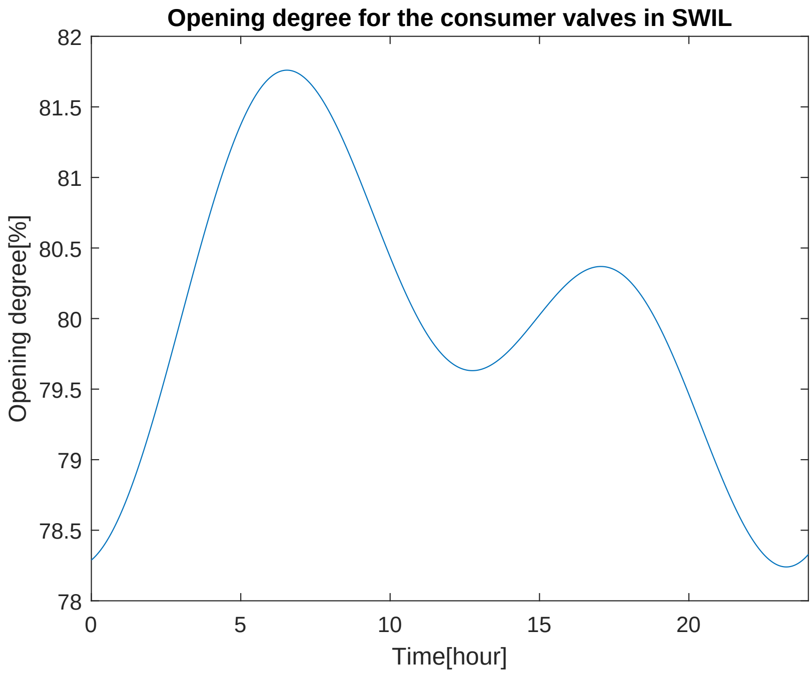

Figure 7.27: Opening degrees of the valves used in the test throughout each iteration

The results obtained from the test are compared between the first 20 iteration and the last 20 iteration of the simulation. In this test the initial control sequence was taken to be 85% of the pump speed, then the ILC updates the control sequence for k^{th} iteration from the previous control sequence iteration $k-1$ along with the error sequence computed from taking the maximum value of the difference between the reference pressure 0.5 bar and the consumer pressure measurements from the $(k-1)^{th}$ iteration and a gain of $K = 0.16$. Using the parameters mentioned above the pressure measurements for the two consumers was acquired as shown in Fig. 7.28, the figure compares between the pressure measurements at the first 20 iterations and the last 20 iterations.

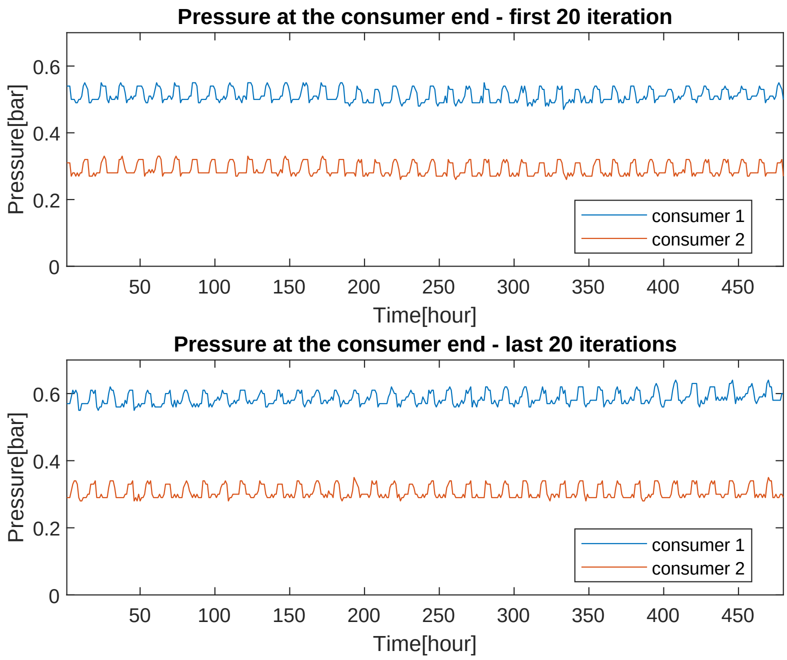

Figure 7.28: Pressure measurements of the first 20 and last 20 iterations taken from the test made in SWIL

The pressure at the first consumer starts at 0.5 bar which is equal to the reference pressure and the second consumer starts at 0.3 bar, therefore, the ILC lifts the pump speed in order for the pressure at the second consumer to reach the reference pressure, in the last 20 iteration, it showed a noticeable increase of the pressure at the first consumer and slightly increase at the second consumer.

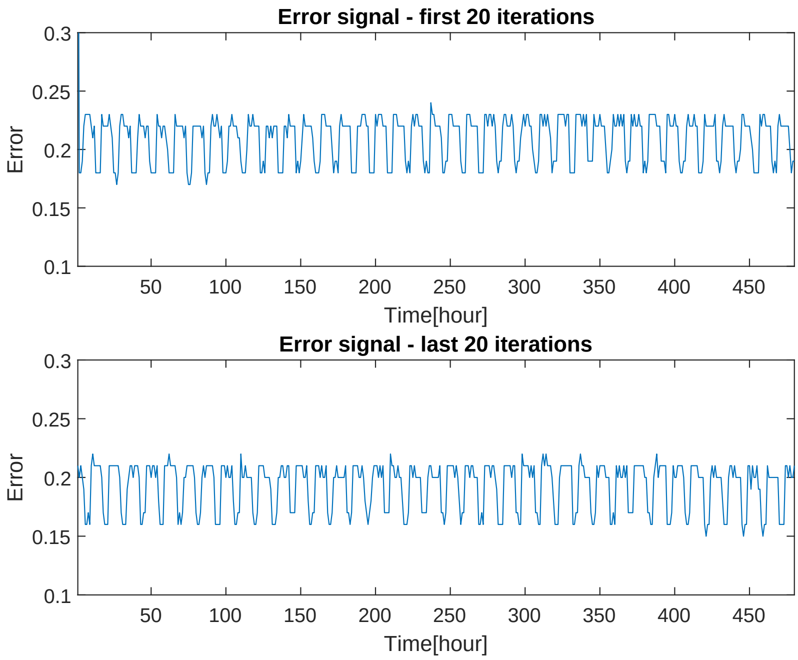

Figure 7.29: Error measurements for the first 20 and last 20 iterations from the lab test

in Fig. 7.29 the error sequence obtained from the maximum value of the difference between the reference and pressure measurements of the two consumers are shown, from these measurements it is clear that the second consumer is dominant in generating the control sequence for the system which explains the increase in the input or control sequence seen in Fig. 7.30 where the control signal increased from 85% to 88% of the full pump speed.

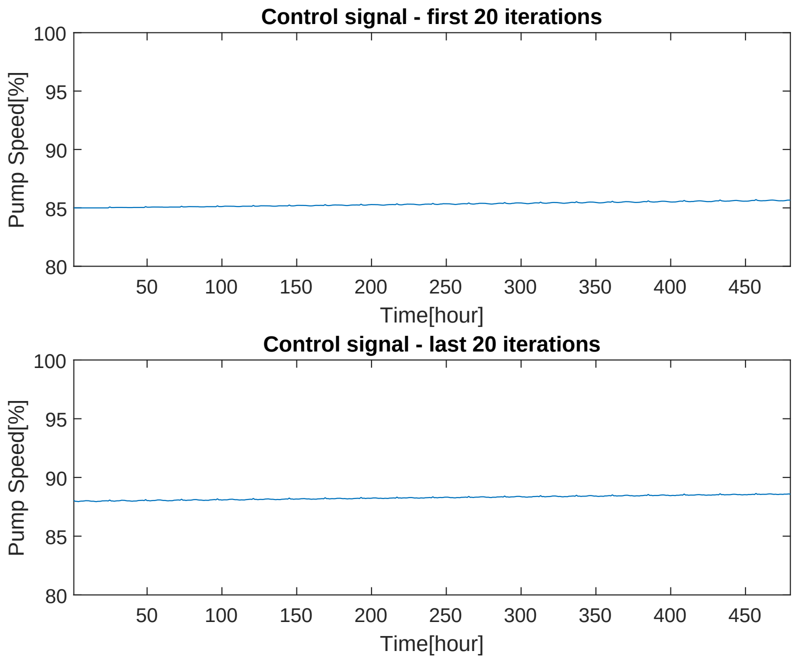

Figure 7.30: Control sequence for the first 20 and last 20 iterations from the lab test

Chapter 8

Leakage detection

In this chapter a leakage detection method is presented, this detection method is based on the residuals of the pressure in the water distribution network, and a general structure of the leakage detection method is presented where it uses the static model as the estimation model to generate the pressure residuals for the dynamic model.

8.1 Residual generation for leakage detection

In this section, the general principle of the leakage detection used in this project is illustrated as shown in Fig. 8.1 where the generation of the residuals r_i is the groundwork of this leakage detection method, the residuals contain information about the changes in the water distribution network where any deviation from estimated pressure $\hat{p}_i(t)$ in the static model will be seen from the residuals, then the residuals are compared with a threshold to identify whether there is a leakage in the water distribution network or not.

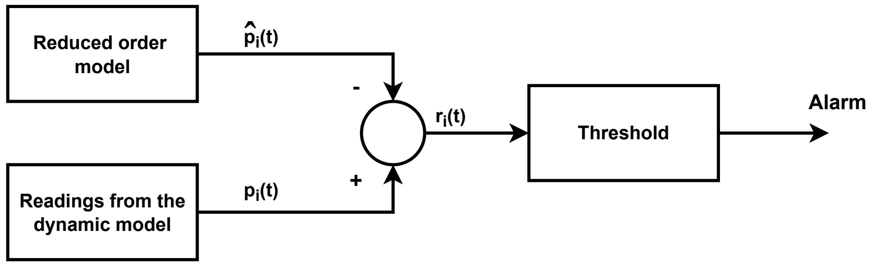

Figure 8.1: Block diagram of the leakage detection structure

The residuals are generated from the subtraction of the measured readings from the actual system which in this case is the dynamic model (Section 7.6) from the static model (Section 7.3.2),

$$r_i(t) = p_i(t) - \hat{p}_i(t) \tag{8.1}$$

Where,
$r_i(t)$ is the residual pressure at node v_i.
$p_i(t)$ is the output pressure of the dynamic model at node v_i.

$\hat{p}_i(t)$ is the estimated pressure at node v_i.
To estimate the pressure $\hat{p}_i(t)$ at the ith node the pressure equation of the static model (4.67) can be used[30][31],

$$\overline{p}_i(t) = \alpha_i d_n(t)^2 - \gamma_i + u(t) \tag{8.2}$$

Since the demand profile in the water distribution network d_n exhibit a periodic behaviour which means $d_n(t+T) = d_n(t)$ where T is the length of the period, then only the parameters α_i and γ_i needs to be found to estimate $\hat{p}_i(t)$[32].
With the estimation model in place a leakage will be introduced to the dynamic model where a leakage demand vector $\overline{d}_l$ is added to the nominal demand of the model as shown in equation (8.3)

$$q = B^T q_c + \begin{bmatrix} 0 \\ \overline{H}_T^{-1} \end{bmatrix} \overline{d} + \begin{bmatrix} 0 \\ \overline{H}_T^{-1} \end{bmatrix} \overline{d}_l \tag{8.3}$$

Where,
$\overline{d}_l = [d_{l,1}, d_{l,2}, \cdots, d_{l,n-1}]'$ and n is the number of nodes in the network. Using the same derivation used in section 4.2.4 the dynamic model with the leakage can be written as,

$$\begin{aligned} \begin{bmatrix} B\mathcal{J}B^T & B\mathcal{J}B_d^T\overline{F} \\ \overline{F}^T B_d \mathcal{J} B^T & \overline{F}^T B_d \mathcal{J} B_d^T \overline{F} \end{bmatrix} \begin{bmatrix} \dot{q}_C \\ \dot{\overline{d}}_f \end{bmatrix} &= - \begin{bmatrix} I & -\overline{H}_C^T \overline{H}_T^{-T} \\ 0 & -\overline{F}^T \overline{H}_T^{-T} \end{bmatrix} \\ \left(\begin{bmatrix} \lambda_C(q_C) \\ \lambda_T(q_C, \overline{d}_f) \end{bmatrix} + \begin{bmatrix} \mu_C(q_C, OD_C) \\ \mu_T(q_C, \overline{d}_f, OD_T) \end{bmatrix} - \begin{bmatrix} \alpha_C(q_C, \omega_C) \\ \alpha_T(q_C, \overline{d}_f, \omega_T) \end{bmatrix} \right) &+ \begin{bmatrix} 0 \\ \overline{F}^T \end{bmatrix} (\overline{z} - \mathbb{1} z_o) \\ &- \begin{bmatrix} B\mathcal{J}B_d^T \\ \overline{F}^T B_d \mathcal{J} B_d^T \end{bmatrix} \dot{\overline{d}}_l \end{aligned} \tag{8.4}$$

The output pressure of the dynamic model $\overline{p}(t)$ after including the leakage demand to the network can be given as,

$$\overline{p}(t) = \overline{H}_T^{-T} \mathcal{J}_T \dot{q}_T + \overline{H}_T^{-T} (\lambda_T(q_T) + \mu_T(q_T, OD_T) - \alpha_T(q_T, \omega_T)) - \overline{z} - \mathbb{1} z_o + \mathbb{1} p_o \tag{8.5}$$

Where,
$q_T = -\overline{H}_T^{-1} \overline{H}_c q_c + \overline{H}_T^{-1} \overline{F} \overline{d_f} + \overline{H}_T^{-1} \overline{d}_l$
$\dot{q}_T = -\overline{H}_T^{-1} \overline{H}_c \dot{q}_c + \overline{H}_T^{-1} \overline{F} \dot{\overline{d_f}} + \overline{H}_T^{-1} \dot{\overline{d}}_l$

As mentioned before the estimated pressure for node v_1 $\hat{p}_1(t)$ can be found by calculating the parameter α_1 and γ_1 using the values obtained from simulating the static model in section 7.3.2 which results in the following,

$$\overline{p}_1(t) = -0.0808 \cdot d_n(t)^2 - 0 + u(t) \tag{8.6}$$

Using this estimation of $\hat{p}_1(t)$ along with dynamic model before adding the leakage demand vector $\overline{d}_l$ to the model, the residual of the pressure at node v_1 was computed, this is shown in Fig. 8.2

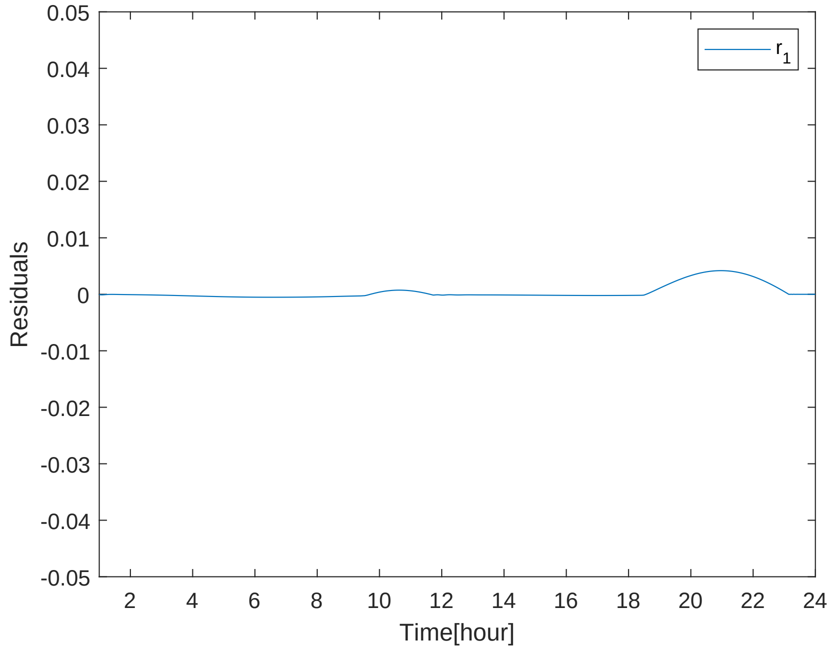

Figure 8.2: A residual of the pressure at node v_1 with no leakage in the network

The threshold for the residuals has been chosen to be at 0.02 which means the pressure from the dynamic model should not be greater than the estimated pressure of more than 0.02 bar.

Chapter 9

Discussion

In this chapter, the results found in Chapter 7 and Chapter 8 are presented below with some discussions surrounding the results.

9.1 Simulation results

This section will describe the results gathered from the static and dynamic models as well as lab test results.

9.1.1 Static model results

- The first test results were gathered from the very first static model. The point of this test was to see if it is possible to make an iterative learning controller work on a static example. From the Fig. 7.1 it can be seen that the iterative learning controller managed to decrease the sinusoidal wave to stabilize around the required reference value of 0.2. With this test, the groundwork for the iterative learning-based controller is done. This controller then is used in all other tests.

- The second test introduced consumers into the equation, which follows a specific valve opening graph represented in Fig. 7.3. Adding to this, pipe dynamics were also added to simulate a water lab setup. With 2 new uncertainties, the code had to be rewritten to accommodate these changes. A new sinusoidal wave was created to look as close as possible to Fig. 7.3. The results are plotted on the x-axis to be the time in hours, while the y-axis is the pressure in bar. Since the reference is 0.2 bar, based on the graph 7.6, the controller managed to reach required goal. The sharp dips can be seen from the initial phase at around 7 hour mark as well as 18. This happened due to average consumption at these specific times being highest, resulting in a pressure drop.

- The third test was similar to 2^{nd} one, with the only difference being that now there were 2 consumers. Since the test had to make both consumers reach 0.2 bar pressure, it was decided to not have both consumers with the same water consumption, but add a small variation. This variation was introduced by using scalar weights on each consumer. Consumer 1 had a 0.3 weighting factor, while consumer 2 had 0.7. The sum of all the weight scalars had to be equal to 1 in order to reflect the same total consumption of the water. The distribution of water consumption can be seen in Fig. 7.7. After running the simulation for 3 times, the Fig. 7.8 was generated. We can see smaller dips at the 7 and 18 hour mark, since both consumers contributed to the usage of the water. The 2^{nd} iteration lowered drastically, due to the gain being quite high at 0.9. After the last iteration, it reached close to the required reference. The new Fig. 7.9 was made for better representation of the consumers. In this figure, it can be seen that the goal of 0.2 bar has not been met. This might occur due to increased complexity and requires more days to stabilize at 0.2 bar. Maybe if the simulation was increased from 3 days to 4 days, the required goal could be achieved.

9.1.2 Dynamic model results

- The first simulation of dynamic model was carried out using the network graph representation given in figure Fig. 7.10 the matrices related to the network were derived in Eq. 7.4,Eq. 7.5 and Eq. 7.6. The parameters used in the simulation were also given in table Table 7.1a, Table 7.1b, Table 7.2a and Table 7.2b, in the simulation the valve opening degrees for the consumers were set to operate as shown in Fig. 7.11 so that the flow to the consumer will imitate the average indoor water consumption as given in Fig. 7.3. The result is given in Fig. 7.12 where the flow has two peaks, one in the morning around 6o'clock where the flow is around $0.63m^3/h$ and $0.68m^3/h$ for the first consumer and the second consumer respectively. The second peak is in the evening around 18 o'clock where the flow is $0.45m^3/h$ and $0.46m^3/h$. From the shape of the figure it can be seen that the behavior of the flow imitates the average indoor consumption. The simulation was run to output the pressure of the consumer, a fixed control of 100% of the full pump speed was applied in this simulation, the pressure measurements of such input is given in Fig. 7.13. The valve opening has an affect on the pressure behaviour at the consumer nodes where closing the valve will build up the pressure and opening it will release the pressure as shown in Fig. 7.13.
- The second simulation was implemented to introduce the ILC in the dynamic model network that has 1 consumer which is the edge e_5 in the network graph representation Fig. 7.10. The simulation was set to run 24 hours each iteration. Valve opening degrees were set to have the same characteristics as in Fig. 7.11 so that the flow to the consumer Fig. 7.14 will follow the concept of the average indoor consumption given in Fig. 7.3. The ILC was used to generate a new control sequence after completing each iteration and applied to the next iteration as shown in Fig. 7.15. It can be seen from the figure that the control sequence updates every iteration. It reduces the pressure at the consumer node to stabilize around reference pressure of 0.2 bar. The effect of these control sequences can be seen from the pressure measurements in Fig. 7.16 and in Fig. 7.17, where the pressure of the 4^{th} iteration is shown that it reached the reference pressure.
- The third simulation was executed using the dynamic model with both consumers

in the network as shown in Fig. 7.10. The ILC was also introduced in this simulation to bring both consumer pressures to the reference pressure, and the controller uses the maximum value of the difference between the reference pressure vector and the pressure measurement vector for both consumers. The initial control sequence was set to be 100% of the full pump speed which results in a pressure of 0.65 bar at the start of the first iteration as shown in Fig. 7.20. Then the control sequence updated, as seen in Fig. 7.19, where the changes in the control sequence from iteration to iteration is barely noticed. This is due to the pressure of both consumers being almost the same. In the second iteration, the pressure measurements are almost at the reference pressure which will not require high changes in the control sequence for the next iteration.

9.2 Lab results

- The first lab test was conducted over couple of days of the period. Each iteration for the system was applied every 48 minutes with sampling every 2 minutes. This means that the system only gets 24 samples total for each iteration. After the first iteration, in Fig. 7.23, the average pressure can be said to be around the reference point which is set up at 0.5 bar. This happened due to correctly estimating the required propeller speed in the pump and setting it up as the initial value. No testing was done prior to looking for the most optimal propeller speed to not get biased results. The propeller speed values after the first iteration can be seen in Fig. 7.24. There is a spike around 3^{rd} hour, due to a small pressure drop. After the first iteration of ILC, the simulation was kept running throughout the night, and after a couple of days, the simulation was stopped. The last full iteration that it managed to finish was 28^{th}. After the final iteration, the data was taken and projected into 2 figures: Fig. 7.25 and Fig. 7.26. As seen in the figures, the pressure values stayed almost identical, with small variations, due to propeller speed values being close to what was initialized in the beginning. On the other hand, the control signal values seem to be more unstable than the initial ones. From the Fig. 7.26, it can be seen that the values increased from the initial 70% to approximately 75.4%. The instabilities that occurred might come from another pump, that refills the system tank from time to time. Once a certain level threshold drops in the tank, the pumps will kick in, and entire pressure will drop in the system until the pumps will stop after stabilizing the levels in the tanks. To further test this, it might have been required to increase the gain or run the simulation even long to reflect on the pump changes.

- The second lab test was carried out using both consumers in the ILC calculations of updating the control sequence. This simulation test was running for 113 iteration where each iteration was set to be 24 minutes. The opening degrees of the valves in consumer ends were chosen to vary from 78% to 82% of the full opening degree as shown in Fig. 7.27. The results obtained from the test were presented as a comparison between the first 20 iterations and the last 20 iterations of the test. The pressure measurements gathered from using such opening degrees in the consumer valves in addition to the ILC were presented in Fig. 7.28, where in the first iterations the pressure at the first consumer node was 0.5 bar. It fulfilled the pressure requirement for that consumer, while for the second consumer the pressure was at 0.3 bar. The control sequence was updated in each iteration to raise the pressure at that consumer to the reference pressure. In the last iteration it can be seen that the pressure at the first consumer has increased to 0.6 bar and for the second consumer the pressure was slightly increased. Firstly this might have happened due to using just one pump in the network which can only give one value at each instance of time, secondly the relation between the pressure and the flow was also part of this problem. Since the consumption on the second consumer was found to be high, which causes the pressure to stay low. The control sequences and the error signals for the two sets of the iterations were presented in Fig. 7.30 and Fig. 7.29. In the error signal figure it can be seen that the second consumer is the dominant consumer and the control sequence was obtain based on the second consumer. It is due to ILC using the maximum value from the error vector as shown in Eq. 6.6 where the farthest point from the reference value is chosen in order to generate the control sequence.

9.3 Leakage detection

The leakage detection method proposed for this project was dependent on a threshold that will decide if there is a leakage in the network or not. The static and dynamic models were used as an estimation model and measurement model respectively to find the residuals in the network. Then the residuals would pass through a threshold to check the presence of the leak in the system. A case was considered where the pressure readings are taken from the consumer end node v_1 before introducing the leakage demand the residuals r_i were calculated and it showed that there is no leakage in the network. Since the residuals have not reached the threshold.
To introduce a leakage in the network a leakage demand vector $\overline{d}_l$ was added to the total demand of the network and the dynamic equations were changed to integrate the leakage dynamics. The leakage demand vector $\overline{d}_l$ and its derivative $\dot{\overline{d}}_l$ were plotted as shown in Fig. 9.1 and Fig. 9.2.

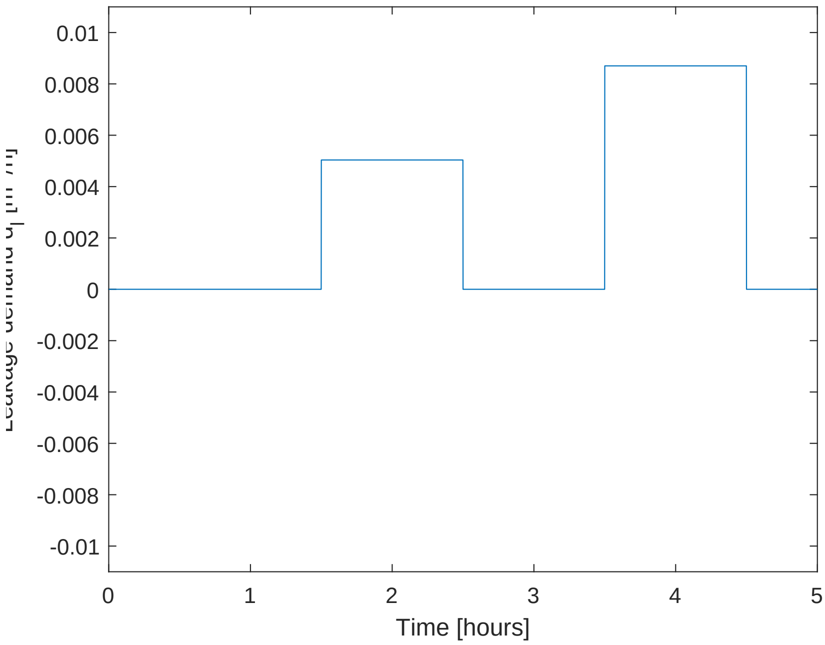

Figure 9.1: A leakage demand vector

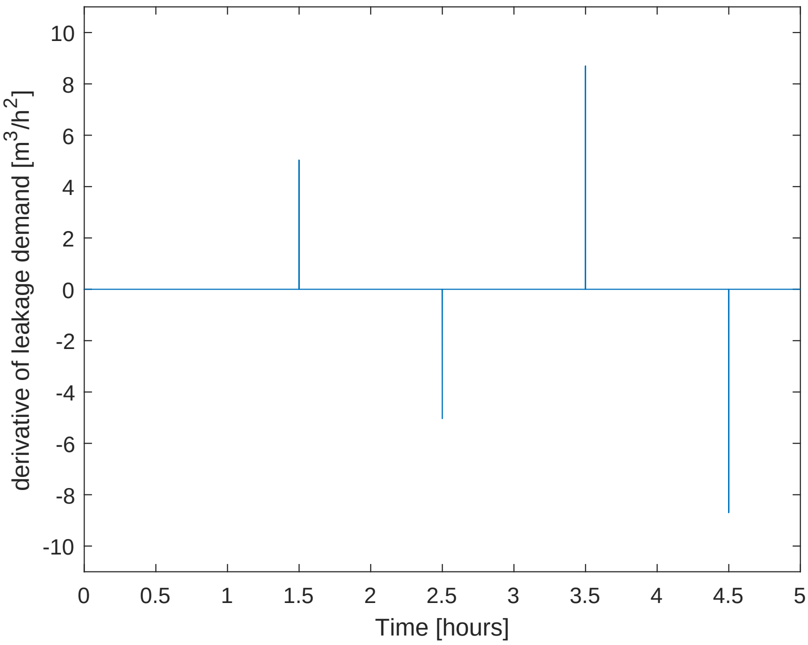

Figure 9.2: Derivative of the leakage demand vector

The output pressure of the dynamic model was unstable at this moment and testing the leakage detection method was not possible.

9.4 Requirement discussion

9.4.1 Simulation requirements

- The first requirement "Iterative learning control should stabilize the system at the reference point after five iterations, if the consumer consumption is considered to be a sinusoidal wave" was achieved under just 3 iterations. The Fig. 7.2 shows all 3 iterations, and necessary steps for achieving the reference pressure. It started as a sinusoidal wave, and after running 3 iterations, the simulation stopped, since the required reference was reached.
- The second requirement "Iterative learning control should stabilize the system at the reference point of 0.2 Bar even with more consumers that act as disturbances to the network". This requirement was achieved in all static and dynamic tests that were done with simulation. The first test was done with the static model and 1 consumer. From Fig. 7.6 it can be seen that the test managed to achieve desired results. The

second test was done with the static model but with 2 consumers this time. From Fig. 7.9 it can be seen that the simulation did not managed to exactly achieve required reference. However, judging from the figure and the number of iterations it is fair to say that it will fulfill the requirement if the simulation had been run for a longer time. Looking into dynamic simulation tests, the first test of the dynamic model with 1 consumer has fulfilled this requirement by achieving a pressure of 0.2 bar as shown in Fig. 7.17, the second test of the dynamic model using the ILC, both consumer pressures have achieved pressure measurements that are very close to the reference pressure 0.2 bar as shown in Fig. 7.21. Therefore, the second requirement is fully fulfilled.

9.4.2 Lab requirements

- The first requirement "Can the iterative learning controller control the pressure to reach the desired reference of 0.5 Bar when a constant valve opening degree is used in the network?", this requirement was achieved, since the first lab result from Fig. 7.25, shows that the system after 28^{th} iterations, managed to reach 0.5 bar.
- The second requirement "Using the iterative learning control, water pressure has to reach the desired reference of 0.5 Bar when the valve dynamics has a periodic opening degree", this second lab test was used to test this requirement, at the start of the simulation the pressure at first consumer end has fulfilled the requirement wherein the second consumer the pressure was too low, therefore, the requirement was not fulfilled in pressure at the second consumer. At the end of the simulation, the pressure at both consumers' ends did not achieve the requirement this may happen due to: firstly, the ILC generating the control sequence that applies to only one pump which affects both consumers at the same time and leads to increase the pressures for both consumers, as shown in Fig. 7.28 secondly, the consumption of the second consumer was high which made the pressure low, and therefore did not reached the reference pressure.
- The third requirement "Iterative learning control should never go above 100% of the maximum propeller speed" was achieved in both lab tests. The first test Fig. 7.26 shows that the propeller speed never went above 76% of maximum speed. For the second test, the propeller speed for the start and end of the simulation was presented in Fig. 7.30 where it shows that the speed has reached 88% of the full speed.

9.4.3 Leakage detection requirement

- The requirement: "The generated pressure residuals should indicate whether or not there is a leakage in the network model", a case was simulated where there are no leakage in the network, the pressure residuals generated in this case indicated the absence of the leakage (Section 8.1), and therefore, the requirement in this case was fulfilled. Another case was attempted to simulate where a leakage demand vector was introduced to the dynamic model to imitate a leak in the network, due to the simulation model becoming unstable after adding the leakage demand, testing this requirement was not possible (Section 9.3).

9.5 Future work

- Introducing a second controller to control the consumer consumption (flow to the consumer) while using the iterative learning control for controlling the pressure at the pump, using these two controllers simultaneously will allow having an adequate flow and the desired pressure at the consumers' end nodes. This suggestion is made based on the results from the second lab test Section 7.8 where the consumption of the consumer was affecting the pressure measurements.
- During lab tests it was noticed that one pump could not supply the water network with enough flow and pressure. This was due to having two consumers with a valve opening degree of up to 100%. A solution would be to reduce the maximum opening degree of the valves, since at the time of writing other pumping station pumps, that supply piping unit with water, did not function properly.
- It is important to mention that only one of the household water consumption patterns were used in the lab tests. Outside of lab environment all of the demand patterns shown in Fig. 2.6 would be seen. Also the demand patterns would differ between before COVID-19 and during COVID-19 patters, seen in Fig. 2.7. This would be due to different lockdown measures in each country. Since the ILC was not tested on other demand patterns it not known how it would impact the system, if the patterns would change daily peak hours of consumption.
- Introducing the leakage demand to the AAU lab set up and applying more concrete detection methods such as Generalized likelihood ratio, CUSUM algorithm, or Change-Point-Analysis to the residuals, since the measurements from the lab set up will contain noise and using the threshold to detect the leakage will not be sufficient.
- Using generated pressure residuals at all nodes to localize the leakage by comparing these residuals and residuals from an estimation model and looking for pressure variations to indicate where to search for the leakages.

Chapter 10

Conclusion

This section assess the approaches and the results that have been obtained throughout the report. The project aimed to develop an iterative learning control for the water distribution network along with a leakage detection method, which leads to the problem formulation:

How iterative learning based controller at the inlet, in addition to a leakage detection method at the outlets, can accommodate the pressure requirements in the water network, which has a periodic disturbance?.

To fulfill the goals of the project dynamic models were derived for the components of the water distribution network namely the pipes, the valves, and the pump, later on, these component models were used to derive mathematical models for the static and dynamic water networks (Chapter 4). The control scheme proposed for the project was the iterative learning control where the pressure at the inlet - pump - is controlled to achieve the pressure requirement at the outlets - consumer end valves - of the water distribution network. The iterative learning control algorithm updates the input sequence to the network after each iteration was presented (Section 6.2), the stability analysis of the controller was also conducted to determine the gain that will make the controller stable using the error convergence method (Section 6.3).

Several simulation tests were made to investigate the behaviour of the iterative learning control, both static and dynamic network models were used to test the controller. Simulations were carried out to observe whether the controller will achieve the requirements specified in Chapter 3. The control tests were executed with different layouts of the water distribution network and it was seen that all of the simulation tests using iterative learning control have fulfilled the requirement of having the pressure at the consumers to be equal to the reference pressure (Section 7.1 to Section 7.6). Two lab tests were conducted in the AAU smart water infrastructure laboratory where the layout of the water distribution network was set up as shown in Fig. A.1. Iterative learning control was used to achieve pressure requirement in the lab tests, the first test shows that the consumer node pressure is at the reference pressure and thereby fulfills the requirement. The second test did not manage to fulfill the requirement, this might have happened due to the consumption of the second consumer being high enough to keep the pressure low, and therefore, force the input sequence for the next iteration to go higher, which led to the increased pressure at the first consumer where it exceeded the reference pressure (Section 7.7 and Section 7.8).

The leakage detection method proposed in this project was generating pressure residuals from the estimated pressures obtained by the static model and using the dynamic model as the actual system. Then the residuals will go through a threshold, where it indicated whether there is a leakage in the network or not. A case with no leakage is introduced to the dynamic model. In tests it showed the network has no leakage because of the residuals not reaching the threshold (Chapter 8). Introducing a leakage demand to the dynamic model was attempted, which made the network unstable and therefore the leakage detection test was not possible to complete in this case (Section 9.3).

Bibliography

[1] UN Committee on the Rights of the Child (CRC). *General comment No. 15 (2013) on the right of the child to the enjoyment of the highest attainable standard of health (art. 24)*. Available: https://www.refworld.org/pdfid/4538838d11.pdf. Accessed on: October 21, 2021. [Online]. Jan. 2003.

[2] World Water Council, ed. *Istanbul Water Consensus For Local and Regional Authorities*. Istanbul, Turkey: Proceedings of the 5th World Water Forum, 2009.

[3] National Research Council. *Drinking Water Distribution Systems: Assessing and Reducing Risks*. Washington DC: The National Academies Press, 2006.

[4] Kazeem Adedeji et al. "Pressure Management Strategies for Water Loss Reduction in Large-Scale Water Piping Networks: A Review". In: (Jan. 2018), pp. 465–480. DOI: 10.1007/978-981-10-7218-5_33.

[5] Jingyu Chen, Xin Feng, and Shiyun Xiao. "An iterative method for leakage zone identification in water distribution networks based on machine learning". In: *Structural Health Monitoring* 20.4 (2021), pp. 1938–1956. DOI: 10.1177/1475921720950470. eprint: \url{https://doi.org/10.1177/1475921720950470}. URL: %5Curl%7Bhttps://doi.org/10.1177/1475921720950470%7D.

[6] Xin Li et al. "Leakage localization using pressure sensors and spatial clustering in water distribution systems". In: *Water Supply* 22.1 (July 2021), pp. 1020–1034. ISSN: 1606-9749. DOI: 10.2166/ws.2021.219. eprint: \url{https://iwaponline.com/ws/article-pdf/22/1/1020/993595/ws022011020.pdf}. URL: %5Curl%7Bhttps://doi.org/10.2166/ws.2021.219%7D.

[7] Sanghoon Jun et al. "Response Surfaces for Water Distribution System Pipe Roughness Calibration". In: *Journal of Water Resources Planning and Management* 148.3 (2022), pp. 1–12. DOI: 10.1061/(ASCE)WR.1943-5452.0001518.

[8] Nourhan Samir et al. "Pressure control for minimizing leakage in water distribution systems". In: *Alexandria Engineering Journal* 56.4 (2017), pp. 601–612. ISSN: 1110-0168. DOI: https://doi.org/10.1016/j.aej.2017.07.008. URL: https://www.sciencedirect.com/science/article/pii/S1110016817302363.

[9] M. Farley, Z.B. Ghazali G. Wyeth, and S. Singh A. Istandar. *The Manager's Non-Revenue Water Hand book, A Guide to Understanding Water Losses*. DAI, 2008.

[10] DTK Hydronet Solutions. *Pressure Management in Water Distribution Networks*. Apr. 2020. URL: %5Curl%7Bhttps://www.dtkhydronet.com/post/pressure-management-in-water-distribution-networks%7D.

[11] Halidu Abu-Bakar, Leon Williams, and Stephen H. Hallett. *Quantifying the impact of the COVID-19 lockdown on household water consumption patterns in England*. npj Clean Water 4. Accessed on: October 22, 2021. [Online]. Jan. 2003.

[12] Benjamin Keldberg Andersen. *Optimal Control in Water Distribution*. Fredrik Bajers Vej 7B, DK-9220 Aalborg, 2020.

[13] Saruch Satishkumar Rathore. *Nonlinear optimal control in water distribution network*. Fredrik Bajers Vej 7B, DK-9220 Aalborg, 2020.

[14] Joannes Reynskor Artis Revelis Ibrahim Abdiweli Jama. *Detection and localization of leakages in water pipes*. Fredrik Bajers Vej 7B, DK-9220 Aalborg, 2021.

[15] Prabhata K. Swamee. *Design of Water Supply Pipe Network*. Wiley-Blackwell, 2008.

[16] Narshingh Deo. *graph theory with applications to engineering and computer science*. Dover Publications, INC., 2016.

[17] U.S.R. Murty J.A. Bondy. *Graph Theory*. springer, 2008.

[18] Robin J Wilson. *Introduction to Graph Theory*. 5th ed. England: Pearson Education Limited, 2010.

[19] Saruch Satishkumar Rathore et al. "Leakage Localization in Municipal Water Supply using Self Adaptive Reduced Network Models and Sensitivity Analysis". In: *2021 5th International Conference on Control and Fault-Tolerant Systems (SysTol)*. 2021, pp. 199–204.

[20] R. Wisniewski. C. S. Kallesøe T. N. Jensen. "Adaptive reference control for pressure management in water networks". In: *European Control Conference (ECC)* (2015), pp. 3268–3273.

[21] Jan Dimon Bendtsen Tom Nørgaard Jensen Carsten Skovmose Kallesøe and Rafał Wisniewski. "Iterative Learning Pressure Control in Water Distribution Networks". In: (2018), pp. 583–588.

[22] Carsten Kallesøe Jorge Val Ledesma Rafal Wisniewski. *Smart Water Infrastructures Laboratory (SWIL)*. 2019.

[23] YangQuan Chen Hyo-Sung Ahn and Kevin L. Moore. "Iterative Learning Control: Brief Survey and Categorization". In: *IEEE Transactions on systems, Man, and Cybernetics* 37.6 (2007), pp. 1099–1121.

[24] Ola Markusson. "Model and system inversion with applications in nonlinear system identification and control". PhD thesis. Sweden: Kungliga Tekniska Hogskolan, 2002.

[25] Tong H. Lee Jian-Xin Xu Sanjib K. Panda. *Real-time Iterative Learning Control -Design and Applications*. Springer-Verlag London Limited, 2009.

[26] A.G. Alleyne D.A. Bristow M. Tharayil. "A survey of iterative learning control". In: *IEEE Control Systems Magazine* 26.3 (2006), pp. 96–114.

[27] Chai Sheng Yu Miao. "A Survey on High-Order Internal Model Based Iterative Learning Control". In: *IEEE Access* 7 (2019), pp. 127024–127031.

[28] Mikael Norrlöf and Svante Gunnarsson. "Time and frequency domain convergence properties in iterative learning control". In: *International Journal of Control* 75.14 (2002), pp. 1114–1126. DOI: 10.1080/00207170210159122. eprint: https://doi.org/10.1080/00207170210159122. URL: https://doi.org/10.1080/00207170210159122.

[29] Thulo Ram Gurung et al. "Smart meters for enhanced water supply network modelling and infrastructure planning". In: *Resources, Conservation and Recycling* 90 (July 2014), pp. 34–50. DOI: 10.1016/j.resconrec.2014.06.005.

[30] Tom Nørgaard Jensen et al. "Residual generation for leakage signatures in water supply networks with multiple inlets". In: *IFAC-PapersOnLine* 51.24 (2018), pp. 717–722.

[31] Tom Nørgaard Jensen and Carsten Skovmose Kallesøe. "Application of a Novel Leakage Detection Framework for Municipal Water Supply on AAU Water Supply Lab". In: *2016 3rd International Conference on Control and Fault-Tolerant Systems (SysTol)*. 2016, pp. 428–433.

[32] Mark Christensen Esben H. Nielsen and Simon N. Gnap. *Leakage detection and localization in water distribution networks with multiple inlets*. Fredrik Bajers Vej 7B, DK-9220 Aalborg, 2021.

Appendix A

Appendix

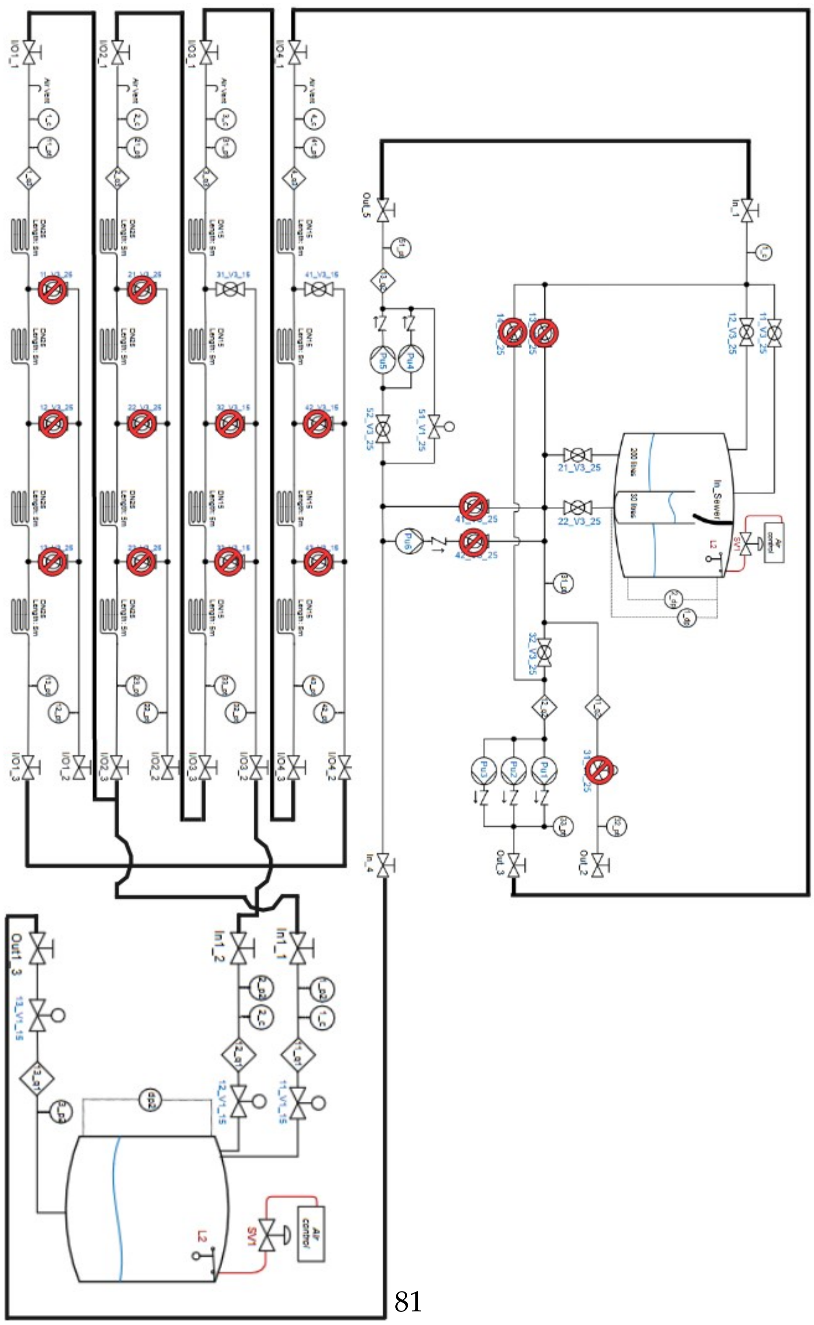

Figure A.1: SWIL setup for the lab tests.

A.1 Static Model MATLAB

```
%% Network setup
%    e1 e2  e3  e4
H=  [1   0   0  -1   %v1
    -1  -1   0   0   %v2
     0   0  -1   1    %v3
     0   1   1   0];  %v4
   % n1 n2 n4
F = [ 1, 0, 0;  % n1
        0, 1, 0;  % n2
        0, 0, 0;  % n3
        0, 0, 1]; % n4

rho = 1000;
g = 9.81;

f1=0.0324;
L1 = 20;
D1 = 0.015;
r1 = 2*f1*8*L1*rho/(pi^2*D1^5);

f2 = 0.0324;
L2 = 20;
D2 = 0.015;
r2 = 2*f2*8*L2*rho/(pi^2*D2^5);

f3 = 0.0301;
L3 = 20;
D3 = 0.025;
r3 = 2*f3*8*L3*rho/(pi^2*D3^5);

f4=0.0301;
L4 = 20;
D4 = 0.025;
r4 = 2*f4*8*L4*rho/(pi^2*D4^5);

R = [r4;r1;r2;r3];
R= R/(1*1e5*(3600^2));
z_nodes =[1;  % n1
          1;  % n2
```

```
          1;  % n3
          1];   %n4
%% Submatrices
rHt = H(1:end-1, 1:end-1);
rHc = H(1:end-1,4);
rF = F(1:end-1, 1:end-1);
rz = z_nodes(1:end-1);
z0 = z_nodes(end);
n = [0 1 0]; % when there is just 1 consumer in the network
n = [1 0 0;
       0 1 0];% when there is 2 consumer in the network
w = [0;1;0];
gamma = rho*g*(rz - z0);
gamma = gamma/(1*1e5);

%% Simulation
simlength=24000;
t=linspace(0,24,simlength);
dn = (25*sin(0.42*pi*0.2*(t-3)) + 25*sin(0.42*pi*0.4*(t-3))+50)/100;
plot(t,dn)
K=0.9;
r=0.2; % when there is just 1 consumer in the network
r=[0.2;0.2];% when there is 2 consumer in the network
days=3;
p=zeros(3,simlength);
b=ones(3,1);
P_n = ones(simlength,50)*0.5;
e=zeros(simlength,1);
p_i=zeros(1,simlength);

%% Minimization of a_c
objective = @(a_c) (R(1)*abs(a_c)*a_c ...
    - rHc'*inv(rHt')*(R(2:end).*abs(-inv(rHt)*rHc*a_c - inv(rHt)*w).*(-inv(rHt)*rHc*a_c - in
a_c0=0;
a_c = fmincon(objective,a_c0,[],[],[],[],[],[],[]);

%%
function_g = inv(rHt')*(R(2:end,:).*(abs(-inv(rHt)*w - inv(rHt)*rHc*a_c).*(-inv(rHt)*w - inv
for i=1:days
    for j=1:simlength
        p(:,j) = function_g*dn(j).^2+b*P_n(j,i)-gamma;
```

```
        p_i(j) = n * p(:,j);
        e(j) = r - p_i(j); % when there is just 1 consumer in the network
        e(j) = max(r - p_i(:,j));% when there is 2 consumer in the network

    end
    P_n(:,i+1) = P_n(:,i) + K * e;
    plot(t,p_i) % when there is just 1 consumer in the network
    plot(t,p_i(1,:),t,p_i(2,:)) % when there is 2 consumer in the network
end
```

A.2 Dynamic model with one consumer MATLAB

```
%% Network setup

%    e1 e2  e3  e4  e5  e6
H=  [1   0   0  -1   1   0    %v1
      -1  -1   0   0   0   0    %v2
       0   0  -1   1   0   0    %v3
       0   1   1   0   0  -1    %v4
       0   0   0   0  -1   0    %v5
       0   0   0   0   0   1];  %v6
 ah0=(7.335e-5*1e5);
%  e1 e2 e3 e4 e5 e6
G=[0   0   0   0   0  ah0];
%   n5, n6
F = [ 0,  0;   % n1
        0,  0;   % n2
        0,  0;   % n3
        0,  0;   % n4
        1,  0;   % n5
        0,  1];  % n6

rho = 1000;
g = 9.81;

f1=0.0324;
L1 = 20;
D1 = 0.015;
r1 = 2*f1*8*L1*rho/(pi^2*D1^5);
A1 = pi*(D1/2)^2;
J1 = L1*rho/A1;
```

```
f2 = 0.0324;
L2 = 20;
D2 = 0.015;
r2 = 2*f2*8*L2*rho/(pi^2*D2^5);
A2 = pi*(D2/2)^2;
J2 = L2*rho/A2;

f3 = 0.0301;
L3 = 20;
D3 = 0.025;
r3 = 2*f3*8*L3*rho/(pi^2*D3^5);
A3 = pi*(D3/2)^2;
J3 = L3*rho/A3;

f4=0.0301;
L4 = 20;
D4 = 0.025;
r4 = 2*f4*8*L4*rho/(pi^2*D4^5);
A4 = pi*(D4/2)^2;
J4 = L4*rho/A4;

a6=-(0.0367*1e5)*3600^2;

J = diag([J1, J2, J3, J4, 0, 0]);
R = diag([r1, r2, r3, r4, 0, 0]);
Alpha= diag([0, 0, 0, 0, 0, a6]);
z_nodes =[1;  % n1
                1;  % n2
                1;  % n3
                1;  % n4
                1;  % n5
                1]; % n6

%% Submatrices
rHc = H(1:end-1,4);
rHt = H(1:end-1, [1:3 5:end]);
rF = F(1:end-1, 1:end-1);
rz = z_nodes(1:end-1);
z0 = z_nodes(end);
```

```
B  = [1 -rHc'*inv(rHt')];
Bd = [zeros(5,1) inv(rHt')];
Jex = [B*J*B',  B*J*Bd'*rF;  rF'*Bd*J*B',  rF'*Bd*J*Bd'*rF];
Dex = [B; rF'*Bd];
Eex = [zeros(1,5); rF'];
Tex = [B', Bd'*rF];

%% Simulation
dt = 0.001;
simTime = 24;
simLength = simTime/dt;
days=4;
t = zeros(1,simLength);
z = zeros(2,simLength);
u = 10000*ones(simLength,days);

mu5 = zeros(1,simLength);
k_vs=1;
OD5=OD(simTime,simLength);
k_v5=k_vs*OD5;
mu5=1./(k_v5.^2)*1e5*3600^2;

p0 = 0;
n = [1 0 0 0 0];
r = 0.2*1e5;
b = ones(5,1);
K = 0.16;
p_i = zeros(1,simLength);
e = zeros(simLength,1);
w = 100*ones(simLength,days);

for i=1:days
for k=1:simLength
 Mu = diag([0, 0, 0, 0, mu5(:,k), 0]);
  % Time
  t(k+1) = t(k) + dt;
 % Control
 w(k,:)= sqrt(u(k,:) );
%Dynamics
  q(:,k) = Tex*z(:,k);
  flow(:,k) = q([2:4 1 5:end],k)*(3600);
```

```
  dz     = Jex\(-Dex*(R([4 ,1:3, 5:end],[4 ,1:3, 5:end])*abs(q(:,k)).*q(:,k) ...
            + Mu([4 ,1:3, 5:end],[4 ,1:3, 5:end])*abs(q(:,k)).*q(:,k) ...
            - Alpha([4 ,1:3, 5:end],[4 ,1:3, 5:end])*abs(q(:,k)).*q(:,k) ...
            - G([4 ,1:3, 5:end])'*(w(k,i)^2)) + rho*g*Eex*(rz - z0));
  z(:,k+1) = z(:,k) + dt*dz;

  % Outputs
  qt = q(2:end,k);
  dq = Tex*dz;
  dqt = dq(2:end);
   p(:,k) = (rHt')\(J([1:3 5:end],[1:3 5:end])*dqt + R([1:3 5:end],[1:3 5:end])*abs(qt).*qt
       + Mu([1:3 5:end],[1:3 5:end])*abs(qt).*qt - Alpha([1:3 5:end],[1:3 5:end])*abs(qt).*q
       - G([1:3 5:end])'*(w(k,i)^2)) - rho*g*(rz - z0) + b*p0;
          p_i(k) = n * p(:,k);
         e(k) = r - p_i(k);
end
u(:,i+1) = u(:,i) + K * e;
    plot(t(1:end-1),p_i/(1*1e5))
end

function y=OD(t,simlength)
x= linspace(0,t,simlength);
y = (25*sin(0.42*pi*0.2*(x-3)) + 25*sin(0.42*pi*0.4*(x-3))+50)/100;

end
```

A.3 Dynamic model with two consumers MATLAB

```
%% Network setup

%     e1 e2   e3   e4   e5   e6   e7
H=   [1    0    0  -1    0    1   0      %v1
      -1   -1    0    0    1    0    0    %v2
       0    0   -1    1    0    0    0    %v3
       0    1    1    0    0    0   -1    %v4
       0    0    0    0   -1    0    0    %v5
       0    0    0    0    0   -1    0    %v6
       0    0    0    0    0    0    1];  %v7

 ah0=(7.335e-5*1e5);
```

```
%  e1 e2 e3 e4 e5 e6 e7
G=[0   0   0   0   0   0  ah0];

%   n5, n6, n7
F = [ 0,  0,  0;   % n1
        0,  0,  0;   % n2
        0,  0,  0;   % n3
        0,  0,  0;   % n4
        1,  0,  0;   % n5
        0,  1,  0;   % n6
        0,  0,  1];  % n7


rho = 1000;
g = 9.81;

f1=0.0418;
L1 = 25;
D1 = 0.015;
r1 = 2*f1*8*L1*rho/(pi^2*D1^5);
A1 = pi*(D1/2)^2;
J1 = L1*rho/A1;

f2 = 0.0418;
L2 = 25;
D2 = 0.015;
r2 = 2*f2*8*L2*rho/(pi^2*D2^5);
A2 = pi*(D2/2)^2;
J2 = L2*rho/A2;

f3 = 0.0418;
L3 = 25;
D3 = 0.015;
r3 = 2*f3*8*L3*rho/(pi^2*D3^5);
A3 = pi*(D3/2)^2;
J3 = L3*rho/A3;

f4=0.0418;
L4 = 25;
D4 = 0.015;
r4 = 2*f4*8*L4*rho/(pi^2*D4^5);
```

```
A4 = pi*(D4/2)^2;
J4 = L4*rho/A4;

a7=-(0.0367*1e5)*3600^2;

J = diag([J1, J2, J3, J4, 0, 0, 0]);
R = diag([r1, r2, r3, r4, 0, 0, 0]);
Alpha= diag([0, 0, 0, 0, 0, 0, a7]);

z_nodes =[1;  % n1
          1;  % n2
          1;  % n3
          1;   %n4
          1;   %n5
          1;  % n6
          1];  % n7

%% Submatrices
rHc = H(1:end-1,4);
rHt = H(1:end-1, [1:3 5:end]);
rF = F(1:end-1, 1:end-1);
rz = z_nodes(1:end-1);
z0 = z_nodes(end);

B  = [1 -rHc'*inv(rHt')];
Bd = [zeros(6,1) inv(rHt')];
Jex = [B*J*B',  B*J*Bd'*rF;  rF'*Bd*J*B',  rF'*Bd*J*Bd'*rF];
Dex = [B; rF'*Bd];
Eex = [zeros(1,6); rF'];
Tex = [B', Bd'*rF];

%% Simulation
dt = 0.001;
simTime = 24;
simLength = simTime/dt;
days = 4;
t = zeros(1,simLength);
z = zeros(3,simLength);
u = 10000*ones(simLength,days);
w = 100*ones(simLength,days);
p_i = zeros(2,simLength);
```

```
e = zeros(simLength,1);
b = ones(6,1);

mu5 = zeros(1,simLength);
mu6 = zeros(1,simLength);
k_vs = 1;
OD6 = OD(simTime,simLength);
OD5 = OD(simTime,simLength);
k_v6 = k_vs*OD6;
k_v5 = k_vs*OD5;
mu6 = 1./(k_v6.^2)*1e5*3600^2;
mu5 = 1./(k_v5.^2)*1e5*3600^2;

n = [1 0 0 0 0 0;
       0 1 0 0 0 0];
r = [0.2*1e5;
      0.2*1e5];
K = 0.16;
p0 = 0;

for i=1:days
for k=1:simLength
 Mu = diag([0, 0, 0, 0, mu5(:,k), mu6(:,k),0]);
  % Time
  t(k+1) = t(k) + dt;
  % Control
  w(k,:)= sqrt(u(k,:));
  % Dynamics
  q(:,k) = Tex*z(:,k);
  flow(:,k) = q([2:4 1 5:end],k)*(3600);
  dz     = Jex\(-Dex*(R([4 ,1:3, 5:end],[4 ,1:3, 5:end])*abs(q(:,k)).*q(:,k) ...
            + Mu([4 ,1:3, 5:end],[4 ,1:3, 5:end])*abs(q(:,k)).*q(:,k) ...
            - Alpha([4 ,1:3, 5:end],[4 ,1:3, 5:end])*abs(q(:,k)).*q(:,k) ...
            - G([4 ,1:3, 5:end])'*(w(k,i)^2)) + rho*g*Eex*(rz - z0));
  z(:,k+1) = z(:,k) + dt*dz;
  % Outputs
  qt = q(2:end,k);
  dq = Tex*dz;
  dqt = dq(2:end);
   p(:,k) = (rHt')\(J([1:3 5:end],[1:3 5:end])*dqt + R([1:3 5:end],[1:3 5:end])*abs(qt).*qt
       + Mu([1:3 5:end],[1:3 5:end])*abs(qt).*qt - Alpha([1:3 5:end],[1:3 5:end])*abs(qt).*q
```

```
       - G([1:3 5:end])'*(w(k,i)^2)) - rho*g*(rz - z0) + b*p0;
          p_i(:,k) = n * p(:,k);
        e(k) = max(r - p_i(:,k)); %error
end
    u(:,i+1) = u(:,i) + K * e;
    figure(2)
    plot(t(1:end-1),p_i(1,:)/(1*1e5),t(1:end-1),p_i(2,:)/(1*1e5))
end

function y=OD(t,simlength)
x= linspace(0,t,simlength);
y = (25*sin(0.42*pi*0.2*(x-3)) + 25*sin(0.42*pi*0.4*(x-3))+50)/100;
end
```