university of groningen

faculty of science and engineering

University of Groningen

Outsourced Private Model Predictive Control

Bachelor Integration Project

To fulfil the requirements for the degree of
Bachelor of Science in Industrial Engineering & Management
at University of Groningen under the supervision of
dr. Nima Monshizadeh Naini
(ENTEG Institute, University of Groningen)
dr. ing. Alexander Hübl
(ENTEG Institute, University of Groningen)
and daily supervision of
Teimour Hosseinalizadeh
(University of Groningen)

Pim Schep (s4517946)

January 19, 2024

Abstract

This integration project explores the viability of outsourcing model predictive control (MPC) to cloud-based services, focusing on privacy risks and mitigation strategies. The study specifically examines a quadruple-tank system (QTS), a complex, multivariable physical process that poses significant challenges for real-time control systems. By shifting the computational demands of MPC to the cloud, the research aims to harness the extensive computational power available, while addressing the pivotal concern of data confidentiality. The QTS and the MPC are simulated in a MATLAB environment. This study places significant emphasis on the inference of key matrices which are pivotal to the MPC framework applied to the quadruple-tank system. The conclusion indicates that by outsourcing the computations of the MPC to the cloud, it is possible to derive certain critical matrices from the data known by the cloud, which represent the physical system and user preferences. Thereafter, two main cryptographic solutions, namely differential privacy and homomorphic encryption, are investigated for their efficacy in ensuring data privacy during cloud computation. Differential privacy is added to the system to prevent this matrix inference. The report outlines the conceptual design and problem statement, conducts a stakeholder analysis, and presents a comprehensive research framework leading to a detailed discussion of potential risks, encryption methodologies, and their practical implementation.

Key words: *Model predictive control, Cloud-based control, Quadruple-tank system, Privacy, Security, Encryption.*

Contents

List of Figures

1 Introduction

In recent times, a surge in connected devices getting smaller and more efficient can be seen. This has led to a greater demand for cloud services. In these services, a powerful central server stores and processes data for users. Cloud computing infrastructure investment accounts for more than 60% of all IT infrastructure spending worldwide in 2023. This surge in cloud investment is attributed to the flexibility and efficiency provided by cloud resources, which are essential for technology decision-makers [1].

Model predictive controllers (MPCs) have a very demanding characteristic and would form a perfect candidate to be outsourced. MPC stands as an effective strategy applied in practical settings across systems of diverse dimensions and configurations, even extending to cloud platforms [2]. Whether it involves competitive contexts like power grid energy generation, everyday domestic applications such as smart home heating control, or time-critical scenarios like traffic management, the computations done by this controller can be very demanding for computers.

When outsourcing the computations done by the MPC, information has to be sent to the cloud and back. This process forms a privacy risk which needs to be overcome. The control system must be accompanied by robust privacy assurances. It is crucial to have these protections to keep user information safe from unwanted access or possible leaks. In light of the privacy risks associated with cloud-based computation, particularly in the context of MPC, this research investigates two prominent encryption methodologies: differential privacy (DP) and homomorphic encryption (HE) [3].

Differential privacy is a technique that allows researchers to publish statistical information about a dataset without revealing any specific data about individuals. It works by adding a small amount of random noise to the results of queries on the dataset. This ensures that removing data from a single individual from the dataset does not significantly change the output of any analysis, thus preserving the privacy of individuals within the dataset. The concept is especially useful when dealing with large-scale data in MPC, where it is crucial to prevent any potential for deducing individual data from a collection of statistics [4].

Homomorphic encryption is an encryption technique that permits computation on encrypted data without decrypting it first. The result of such computation, when decrypted, is the same as if it were carried out on the original data. This allows cloud-based MPC systems to process data while maintaining its confidentiality, as the data remains encrypted throughout the computation process [5].

The quadruple-tank system (QTS) is a classic example for evaluating non-linear dynamics within multivariable processes [6]. Composed of interconnected tanks, the QTS is a laboratory setup that provides a flexible platform for examining multivariable interactions, non-linear responses, saturation effects, and operational constraints [7]. The inherent complexity and versatility of the QTS make it an ideal candidate for the application of model predictive control. MPC's ability to anticipate future events and take control actions aligns well with the system's characteristics, allowing for sophisticated management of the interdependent variables and non-linear behaviours in this process.

The remainder of this integration project is structured to provide a comprehensive exploration of cloud-based MPC systems and the encryption techniques employed to preserve data privacy. Chapter 2 delves into the simulation of MPC on the QTS, providing system parameters, control challenges, and the optimization problem. In Chapter 3, the results of the MATLAB file

running the MPC-controlled QTS are presented. Besides this, the matrices that can be inferred are estimated. Finally, the encryption methods are briefly compared in this Chapter. In Chapter 4 the conclusions are drawn for the whole report and in Chapter 5 the discussion is presented where a critical analysis is provided.

1.1 Conceptual Design

This research addresses the data security of a cloud-based MPC for a QTS. The research consists of two phases. First, the MPC is implemented on the QTS, this process has been done before in papers as [8] and [9]. After this, the privacy of the system including the MPC can be analysed. These steps are necessary to make sure that the privacy risks regarding the inputs, calculations in the cloud and outputs can be overcome, which can be assured by providing a potential solution making use of differential privacy and homomorphic encryption as in [3].

1.1.1 System Description & Scope

This study examines the application of model predictive control for a quadruple-tank system within a cloud computing framework. The QTS is a multivariable process with nonlinear dynamics, representing a considerable challenge for real-time control systems. MPC is particularly suited to address these challenges, given its ability to forecast future system behaviours and adjust control inputs for optimal performance. The conventional approach to MPC relies on localized computational resources, which can be restrictive due to their inherent limitations in processing power and scalability. By outsourcing the computational load to a cloud-based infrastructure, the aim is to make use of the virtually unlimited computational resources that the cloud can provide. Such an approach allows for a more robust and efficient processing of the complex algorithms inherent in MPC, facilitating improved performance and scalability of the control system.

The main focus of this research is on maintaining the confidentiality of the data during its transit to and within the cloud environment. To this end, we explore two encryption methodologies that allow computational tasks to be performed on encrypted data. This ensures that sensitive information with regards to the QTS remains secure, without revealing the details of the operations being performed or the data itself to the cloud service provider. Within the scope of this research, we will critically analyze and propose encryption techniques suitable for MPC computations in the cloud. These techniques must secure data effectively without degrading the accuracy of the control system. After the analysis, one of the techniques will be implemented to secure privacy. The cost-effectiveness of the imposed solutions is left out of the scope as well as other methodologies than DP and HE.

1.1.2 Problem Statement

The implementation of Model Predictive Control for systems such as the quadruple-tank system traditionally relies on local computational resources. This approach necessitates the handling of sensitive operational data, which poses a risk to the privacy and confidentiality of the data when making use of cloud computing solutions. The exposure of such data during the computation process is a significant privacy concern for entities that require confidentiality, particularly when data is transmitted to external cloud servers where it could potentially be accessed or compromised. This challenge is relevant and of high interest to the process control domain

within industrial applications, where the need for computational power, but more importantly, data privacy is of greatest importance.

1.1.3 Stakeholder Analysis

In the stakeholder analysis, the varying interests and degrees of influence among the parties involved in the implementation of the cloud-based MPC system are precisely evaluated. Their placement within figure 1 reflects the extent of their influence on the project and the corresponding impact upon them.

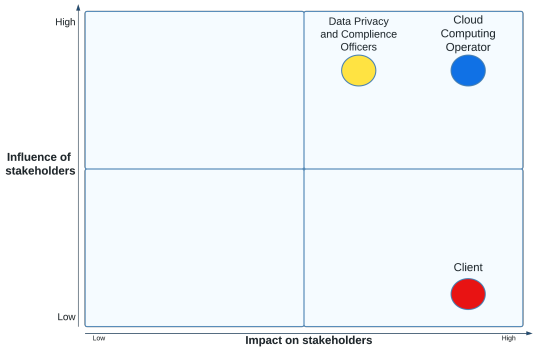

Figure 1: Stakeholder Analysis

- **Cloud Computing Operator**: This stakeholder is in the leading position of the cloud-based MPC system's operation. As the owner of the computational environment, the cloud computing operator's role is integral and multifaceted. Their responsibilities extend beyond the provisioning of services, they are the guardians of data privacy, tasked with implementing robust security measures to protect sensitive client information. Their influential position in the network's hierarchy stems from a deep-seated commitment to privacy and security, aligning their interests closely with the success and integrity of the system. As such, their influence is significant, and their impact is substantial, making them a cornerstone of the research effort.

- **Client**: The client, as the end-user of the MPC system, has a great interest in the system's performance and security. The efficiency of the system and the protection of their data directly influence their operational success. While their influence on the system's overall design and implementation may be moderate, their input is crucial for evaluating the system's success and ensuring that it meets user requirements and expectations. This means that they are a critical voice in the stakeholder landscape, and their feedback is instrumental in shaping the system's evolution.

- **Data Privacy & Compliance Officers**: Entrusted with the oversight of data management, these officers are the architects of privacy and guardians of regulatory compliance. Their role is pivotal in steering the cloud-based environment towards adherence to data protection laws and best practices. They command a substantial degree of influence by

setting the standards and protocols for data handling, thereby ensuring that sensitive control data is managed with the utmost confidentiality. Their impact is both direct and profound, as they play a key role in protecting the system against privacy breaches and maintaining trust in the cloud-based MPC solution.

1.1.4 Research Objective

The overarching objective of this study is to investigate the privacy risks for the QTS, making use of MPC within a cloud computing environment and provide a solution for any privacy risks. This objective is **specific** since it focuses solely on the cloud-integrated MPC of a QTS and focuses specifically on two types of cryptographic solutions. It also certainly is **measurable** since the leakage of information will be analysed. Based on earlier research, the objective of both implementing MPC into the system and achieving the potential privacy risks can certainly be **achieved**. As explained in the introduction the topic of the security of computational processes is very **relevant** in today's day and age. Finally, the research has a deadline on the 19th of January 2024 so is **time-bound**.

1.1.5 Research Framework

The research framework is systematically organized into a coherent flow of stages, each building upon the knowledge and findings of the previous one. Initially, the study will focus on the underlying theory necessary for understanding the complexities of the quadruple-tank system. This will involve a detailed review of MATLAB as the chosen computational platform due to its widespread use in the modelling and simulation of control systems. Thereafter, a comprehensive examination of the existing theories on model predictive control will be undertaken, focusing on its application in control systems similar to the QTS.

Upon establishing the theoretical foundation, the research will progress to the practical implementation of MPC on the QTS within a MATLAB environment. This will include the development and refinement of MPC algorithms tailored to the QTS's specifications. The performance of these algorithms will then be rigorously tested and analyzed to ensure they meet the control objectives effectively.

Subsequently, the research will identify cloud privacy risks relevant to the execution of MPC in a cloud environment and based on the theory of matrix inference. This involves mapping the risks to the specific contexts of cloud-based operations and analysing how they may impact the privacy and security of the control system's data.

In the following stage, the research will tackle the core challenge of running cloud-based MPC for the QTS while maintaining data privacy. This phase will explore the encryption techniques that can be utilized to secure data as it is processed in the cloud, ensuring that the control operations are private and the sensitive data remains confidential.

The last phase of the project will bring together the knowledge from both the study and the hands-on work to thoroughly examine the results. This review will closely look at how well the data protection methods worked, considering their ability to secure information while still ensuring the cloud-based MPC system runs smoothly and accurately. The aim is to draw useful conclusions and give practical advice for applying cloud-based MPC systems in the industrial sector in the future.

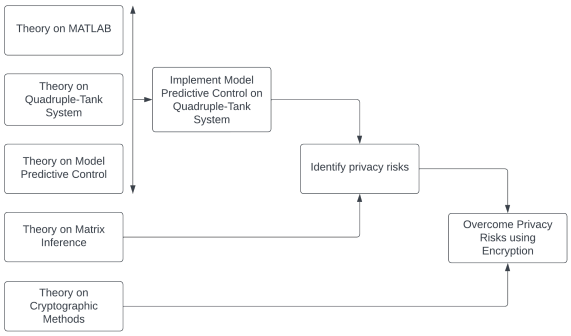

Figure 2: Research Framework

1.1.6 Research Questions

The research questions consist of one main question with multiple sub-questions to in the end answer the main question.

Main question:

What are the privacy risks of outsourcing MPC to a cloud-based environment?

Sub-questions:

- What are the key control challenges and performance objectives in implementing MPC for the quadruple-tank system in a cloud environment?
- What are the potential risks and vulnerabilities associated with cloud-based MPC in terms of data privacy and security?
- What measures and protocols can be implemented to mitigate data privacy and security risks in cloud-based MPC systems?

The subsequent table elucidates the tools and knowledge domains germane to each sub-question:

Field	**Tools and Knowledge Areas**
Control challenges	Control theory, systems engineering, performance metrics, cloud computing architectures
Privacy risks	Cybersecurity principles, risk analysis, cloud infrastructure, data encryption standards
Protecting methods	Cryptographic techniques, privacy-preserving algorithms, network security protocols, compliance standards

1.2 Related Research

The landscape of Model Predictive Control is being transformed by the integration of cloud-based solutions, making rigorous exploration of data privacy methodologies needed. The initial step of this research involves implementing MPC on the QTS, a process previously explored in studies [8] and [9]. The next step in this integration project investigates the cryptographic method of homomorphic encryption as outlined in [5], enabling secure computation on encrypted data, alongside the non-cryptographic approach of differential privacy, which ensures

the confidentiality of individual records even when the overall data is analyzed collectively [10]. Although homomorphic encryption offers robust protection, it brings computational complexity [5]. Differential privacy, conversely, adds noise to the data, trading off some accuracy for privacy, to the standards of [4]. This research will critically compare these methods, evaluating their implications on the privacy and performance of cloud-based MPC in the context of the QTS. The evaluation will consider a range of criteria, such as the system's ability to scale and its processing efficiency, to determine how suitable these methods are for modern industrial settings where maintaining privacy is crucial [3].

1.3 Model Predictive Control for Quadruple-Tank System

1.3.1 Quadruple-Tank System

The quadruple-tank system forms the foundation of this research and is visualised in figure 3. It is a type of apparatus commonly used in the field of control engineering for research and education. It consists of four tanks, arranged in a certain configuration, with two tanks placed higher and two tanks placed lower. The system is designed to study and demonstrate various principles of control systems, such as the dynamics of liquid levels in interconnected tanks, the behavior of multiple interacting control loops, and the effects of disturbances and nonlinearities. The process is a physical process but for this research, it is made digital. The metrics associated with the process are the same as in [11] and are as follows. The inputs for the process are given by $v1$ and $v2$ (voltage fed to the pumps), while the outputs are represented by $y1$ and $y2$ (voltage readings from level measurement devices converted to centimetres). Through the application of mass balances and Bernoulli's principle, the following equations are deduced:

$$\begin{aligned}
\frac{dh_1}{dt} &= -\frac{a_1}{A_1}\sqrt{2gh_1} + \frac{a_3}{A_1}\sqrt{2gh_3} + \frac{\gamma_1 k_1}{A_1}v_1 \\
\frac{dh_2}{dt} &= -\frac{a_2}{A_2}\sqrt{2gh_2} + \frac{a_4}{A_2}\sqrt{2gh_4} + \frac{\gamma_2 k_2}{A_2}v_2 \\
\frac{dh_3}{dt} &= -\frac{a_3}{A_3}\sqrt{2gh_3} + (1-\gamma_2)k_2\frac{v_2}{A_3} \\
\frac{dh_4}{dt} &= -\frac{a_4}{A_4}\sqrt{2gh_4} + (1-\gamma_1)k_1\frac{v_1}{A_4},
\end{aligned} \tag{1}$$

where:

A_i : cross-section of tank I;
a_i : cross-section of the outlet hole;
h_i : water level.

For pump i, the given voltage is denoted as v_i while the associated flow is represented by $k_i v_i$. Based on the valve configurations before any experiment, parameters γ_1 and γ_2, both within the interval (0, 1), are set. The flow directed to tank 1 is calculated as $\gamma_1 k_1 v_1$ while the flow leading to tank 4 can be expressed as $(1-\gamma_1)k_1 v_1$, with similar expressions for tanks 2 and 3. The gravitational force is represented by the symbol g. The level indicators are defined by $k_c h_1$ and $k_c h_2$. Below is a table detailing the lab process parameters:

Parameter	Measurement
A_1, A_3	$[cm^2]$ 28
A_2, A_4	$[cm^2]$ 32
a_1, a_3	$[cm^2]$ 0.071
a_2, a_4	$[cm^2]$ 0.057
k_c	$[V/cm]$ 0.50
g	$[cm/s^2]$ 981.

The process is examined across two specific operational stages, namely P_- and P_+. For P_-, the system behaves with a consistent phase pattern, while P_+ sees the system with a varying phase pattern. This has to do with the initial position of the valves of the system that distribute the water supply after the pumps. Here are the corresponding parameter values for these stages:

Parameter	Value at P_-	Value at P_+
(h_1^0, h_2^0)	$[cm]$ (12.4, 12.7)	$[cm]$ (12.6, 13.0)
(h_3^0, h_4^0)	$[cm]$ (1.8, 1.4)	$[cm]$ (4.8, 4.9)
(v_1^0, v_2^0)	$[V]$ (3.00, 3.00)	$[V]$ (3.15, 3.15)
(k_1, k_2)	$[cm^3/Vs]$ (3.33, 3.35)	$[cm^3/Vs]$ (3.14, 3.29)
(γ_1, γ_2)	(0.70, 0.60)	(0.43, 0.34)

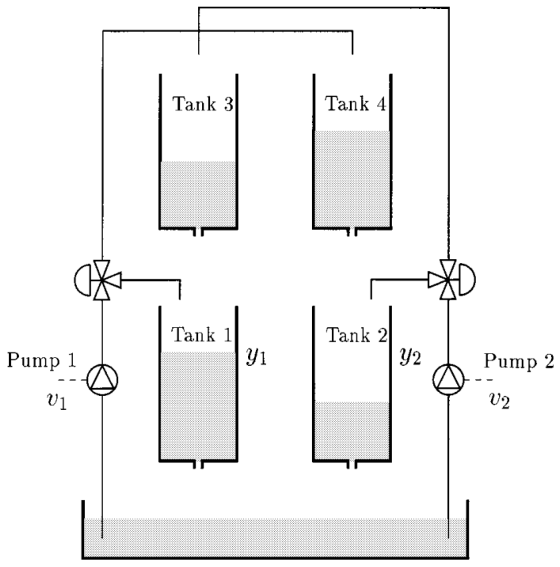

Figure 3: Schematic Overview of the Quadruple-Tank System [11]

To use (1) in the MPC the non-linear equations should be linearized. First we introduce the variables $x_i := h_i - h_i^0$ and $u_i := v_i - v_i^0$. Using the linearization method with the Jacobean matrix we end up with the following state space equation:

$$\frac{dx}{dt} = \begin{bmatrix} -\frac{1}{T_1} & 0 & \frac{A_3}{A_1 T_3} & 0 \\ 0 & -\frac{1}{T_2} & 0 & \frac{A_4}{A_2 T_4} \\ 0 & 0 & -\frac{1}{T_3} & 0 \\ 0 & 0 & 0 & -\frac{1}{T_4} \end{bmatrix} x + \begin{bmatrix} \frac{\gamma_1 k_1}{A_1} & 0 \\ 0 & \frac{\gamma_2 k_2}{A_2} \\ 0 & \frac{(1-\gamma_2) k_2}{A_3} \\ \frac{(1-\gamma_1) k_1}{A_4} & 0 \end{bmatrix} u, \tag{2}$$

$$y = \begin{bmatrix} k_c & 0 & 0 & 0 \\ 0 & k_c & 0 & 0 \end{bmatrix} x, \tag{3}$$

where $c_1 = T_1 k_1 k_c / A_1$ and $c_2 = T_2 k_2 k_c / A_2$, and the time constants are given by

$$T_i = \frac{A_i}{a_i \sqrt{\frac{2h_i^0}{g}}}, \quad i = 1, \ldots, 4. \tag{4}$$

The first matrix in (2) is called matrix A. It represents how the system works over time without any control inputs and is of size $A \in \mathbb{R}^{n \times n}$. The second matrix is matrix B. This matrix represents the reaction of the system to certain control inputs. It is of size $B \in \mathbb{R}^{n \times m}$. The matrix in (3) is matrix C. It maps the state vector into the output vector. It is of size $C \in \mathbb{R}^{p \times n}$.

1.3.2 Model Predictive Control

Model predictive control has established itself as a robust and versatile control approach, finding application in various systems, ranging from simple setups to intricate configurations. Its adaptability has also seen its implementation on cloud platforms, highlighting the flexibility and prowess of MPC in modern control scenarios [2].

The controller makes use of a linear or linearized model that predicts the system's future behaviour based on current states, disturbances, and control actions. At each iteration, MPC works out the best control actions by solving an optimization problem, focusing on reducing a set cost over a certain time horizon. This is visualized in figure 4.

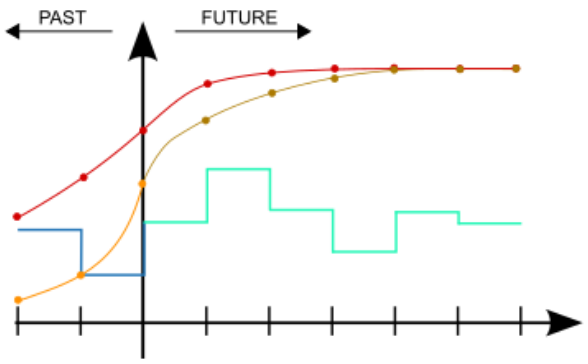

Figure 4: Schematic Overview of how MPC Works [12]

The red line represents the reference trajectory, and the orange line before the vertical axis represents the measured output whereas the brownish line after the vertical axis represents the predicted output based on the control inputs of the MPC. The blue and green lines represent the past and predicted control inputs respectively. The MPC does these computations over a time horizon N. This time horizon is vital for MPC. It sets the span the MPC system considers for future actions. While a lengthier period might improve results, it can also make calculations more complex. One of the main strengths of MPC is its capability of handling constraints. It takes into account the system's state and control limits when optimizing. This means that the control actions the MPC recommends will always stay within the system's set boundaries or limits. With regards to the Quadruple-Tank System, this means that the MPC can for example take the volumes of the tanks and the maximum capacity of the pumps into account. Finally, there is a cost function. At each time step, MPC solves a constrained optimization problem with a generalized criterion for designing linear multivariable control systems:

$$\begin{aligned} \min_{x,u} \quad & J(x,u) = {x_N}^\top P x_N + \sum_{k=0}^{N-1} ({x_k}^\top Q x_k + {u_k}^\top R u_k) \\ \text{s.t.} \quad & u_{\min} \leq u_k \leq u_{\max}, \quad k = 0, \ldots, N-1 \\ & y_{\min} \leq y_k \leq y_{\max}, \quad k = 1, \ldots, N, \end{aligned} \tag{5}$$

where $x \in \mathbb{R}^n$ and represents the current states of the system and $u \in \mathbb{R}^m$ which represents the optimal control inputs over time generated by the MPC.

The cost function in MPC is structured to optimize system performance over a prediction horizon N. This function can be broken down as follows:

- **Prediction Horizon** N: The prediction horizon, denoted by N, refers to the number of future time steps over which the model predictive control algorithm forecasts the behaviour of the system being controlled. It represents the discrete period into the future that the controller considers when optimizing the control inputs. The prediction horizon in MPC is the time the controller looks ahead to make decisions. It uses this to predict what will happen and to adjust the controls early to get the performance it wants. The length of the prediction horizon is selected based on the dynamics of the system, computational constraints, and the specific objectives of the control application. A longer prediction horizon provides a more extensive future outlook, potentially leading to more informed control decisions, but at the cost of increased computational demand.

- **Matrix** Q **(State Weighting Matrix)**: This matrix assigns weights to the states in the cost function, emphasizing the importance of accurately tracking the desired state. A higher weight in Q for a particular state variable signifies a greater emphasis on minimizing the deviation of that variable from its desired value. This matrix must be strictly positive and is of size $Q \in \mathbb{R}^{n \times n}$.

- **Matrix** R **(Control Weighting Matrix)**: R penalizes excessive control action, promoting smoother and more gradual changes in control inputs. This helps in preventing aggressive control actions that might be harmful to the system's stability or operation. This matrix is positive semi-definite.

- **Matrix** P **(Terminal State Weighting Matrix)**: P is used in the final step of the prediction horizon, providing a weight to the terminal state. It ensures that the end state of

the prediction horizon aligns well with the long-term goals of the control strategy. This matrix is positive semi-definite.

1.4 Cloud Inference

For this research, a couple of assumptions when it comes to what the cloud has access to are made according to [13]. This is necessary to understand in what way the cloud can infer other matrices of the physical system. Besides this, some other things about the values of certain matrices and parameters are assumed.

- **Matrices known by the Cloud**: In this case, the assumption is made that only the matrices H and F are known by the cloud. This situation is obtained by the cloud when the formula (5) is reorganised to:

$$\begin{aligned} \min_{z} \quad & \frac{1}{2} z^\top H z + x_0^\top F z + x_0^\top Y x_0 \\ \text{s.t.} \quad & Gz \leq W + O x_0, \end{aligned} \tag{6}$$

where

$$z = \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{N-1} \end{bmatrix}$$

and

$$H := 2\bar{R} + 2\bar{S}^\top \bar{Q} \bar{S}, \tag{7}$$

$$F := 2(\bar{T}^\top \bar{Q} \bar{S})^\top, \tag{8}$$

$$Y := 2\bar{Q} + 2\bar{T}^\top \bar{Q} \bar{T}. \tag{9}$$

Matrices $F^\top$ and H are utilized within the cost function. Matrix F relates the predicted states and inputs to the initial state, while matrix H, which is symmetric and positive definite, encapsulates the quadratic nature of the cost function, ensuring the penalization of deviations from the desired trajectory and excessive control effort. The careful calibration of these matrices within the cost function is essential for the predictive capability of the MPC, balancing the system's performance with the economy of control actions over the prediction horizon. These three matrices are built from the following matrices:

$$\bar{R} := \begin{bmatrix} R & 0 & \cdots & 0 \\ 0 & R & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & R \end{bmatrix} \in \mathbb{R}^{Nm \times Nm},$$

$$\bar{S} := \begin{bmatrix} B & 0 & \cdots & 0 \\ AB & B & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ A^{N-1}B & A^{N-2}B & \cdots & B \end{bmatrix} \in \mathbb{R}^{Nn \times Nm},$$

$$\bar{Q} := \begin{bmatrix} Q & 0 & 0 & \cdots & 0 \\ 0 & Q & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & & \vdots \\ 0 & \cdots & 0 & Q & 0 \\ 0 & 0 & \cdots & 0 & P \end{bmatrix} \in \mathbb{R}^{Nn \times Nn}$$

and finally the matrix $\bar{T}$ described as:

$$\bar{T} := \begin{bmatrix} A \\ A^2 \\ \vdots \\ A^N \end{bmatrix} \in \mathbb{R}^{Nn \times n}.$$

The last matrix matrix implemented on the diagonal of $\bar{Q}$ is P since this matrix represents the terminal state. For $\bar{R}$ a similar approach is taken. The matrices $\bar{S}$ and $\bar{T}$ are constructed to predict future states and inputs over the prediction horizon.

- **Stability Matrix** A: Matrix A is assumed to be Schur stable because this property ensures that the system's response to any initial condition will asymptotically decay to zero without external input, which means the system is inherently stable. This is critical in MPC.
- **Prediction Horizon** N: A large prediction horizon N is favourable because it allows the control algorithm to consider more future states, leading to more informed decision-making. Besides this by considering a longer sequence of future events, the controller can optimize the control actions more effectively. This can lead to smoother control actions, reduced control effort, and better overall system performance. Because of these reasons, we assume that
$$N \to \infty$$
.

With the assumptions in place, the focus is turned to the matrix Y, which is included in the cost function (6). The matrix Y is constructed as follows:

$$Y = \sum_{i=0}^{N-1} (A^T)^i Q A^i + (A^T)^N P A^N, \tag{10}$$

where Q and P are matrices that weigh the importance of states and controls in the system, and A represents how the system evolves over time. When looking ahead into the future,

considering more and more time steps, the value of Y settles into a steady matrix called $\bar{Y}$, which has real numbers and is of size $\bar{Y} \in \mathbb{R}^{n \times n}$. This is captured by the following equation:

$$\lim_{N \to \infty} Y = \bar{Y}. \tag{11}$$

Neither Y nor its steady-state $\bar{Y}$ are known to the cloud initially. This forms the basis of the theory of [13]. The theory states that when the assumptions mentioned earlier hold, and the pair $(A, B^\top \bar{Y})$ is observable the cloud can obtain the matrices A and B from the system as well as the control weighing matrix R. Besides this, if the following set is a singleton, the cloud infers the state weighing matrix Q and the terminal state weighing matrix P.

$$\mathcal{J} = \left\{ X \in \mathbb{R}^{n \times n} \mid \hat{Q} + X - A^\top X A \succeq 0,\ \hat{P} + X \succeq 0,\ XB = 0 \right\} \tag{12}$$

Here $\hat{Q}$ and $\hat{P}$ are defined as

$$\left\{ \hat{Q}, \hat{P} \in \mathbb{R}^{n \times n} \mid \hat{\mathcal{Q}} := \begin{bmatrix} \hat{Q} & 0 & \cdots & 0 \\ 0 & \hat{Q} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \hat{P} \end{bmatrix} \text{ and } \begin{bmatrix} \bar{S} \\ \bar{T} \end{bmatrix} \hat{\mathcal{Q}} \bar{S} = \frac{1}{2} \begin{bmatrix} H - 2\bar{R} \\ F \end{bmatrix} \right\}, \tag{13}$$

where $\bar{S}$, $\bar{T}$ and $\bar{R}$ are the estimated matrices.

If the set of all possible solutions for $\hat{Q}$ and $\hat{P}$ that satisfy the given conditions is a singleton, it means that there is only one $\hat{Q}$ and one $\hat{P}$ that meet these requirements, and the cloud would infer Q and P.

1.5 Encryption in Control Systems

In the era of interconnected control systems, the security of communication channels and the privacy of data have become paramount. Traditional control systems operated in isolated environments where security concerns were primarily focused on physical access. The transition to digital and networked infrastructures has necessitated robust encryption mechanisms to prevent unauthorized access and ensure the confidentiality and integrity of control signals and parameters. This Chapter explores the role of encryption in control systems, particularly emphasising differential privacy and homomorphic encryption as methods to enhance data security and privacy.

1.5.1 Differential Privacy

Differential privacy provides a framework for sharing information about a dataset by describing the patterns of groups within the dataset while withholding information about individuals in the dataset. In the context of control systems, differential privacy can be instrumental in protecting the information of individual components or users within a networked system. The application of differential privacy in control systems is a balancing act between data utility and privacy. Techniques to achieve differential privacy, such as adding noise, some small disturbances, to the aggregate data or using secure multi-party computation, have been widely discussed in the literature [10], [14], [15]. This method is visualised in figure 5.

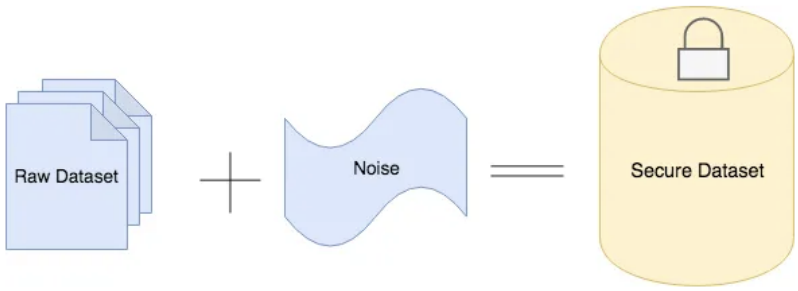

Figure 5: Differential Privacy Mechanism [16]

Differential privacy employs noise addition to mask individual data entries, thus safeguarding privacy while enabling the study of overall patterns. Among the noise types, Laplacian and Gaussian are most prominent due to their distinct properties and the balance they offer between privacy and data utility.

Laplacian noise is linked to a strict privacy standard known as ε-differential privacy. It involves adding noise that diminishes quickly as it moves away from the centre, which means it is sharply peaked at the mean. The intensity of Laplacian noise depends on a single privacy parameter, ε, with lower values indicating higher privacy and more noise. In practice, this could mean that individual data points may vary significantly due to the noise, potentially affecting the precision of the data [17].

Gaussian noise is used when a slightly relaxed privacy standard, termed as (ε, δ)-differential privacy, is acceptable. Gaussian noise, which forms a bell curve, gently tapers off and is less intense at the edges. This results in a smoother noise addition, often causing less drastic changes to the data. The privacy parameters ε and δ work together: ε dictates the overall strength of the privacy, while δ allows for a tiny probability of the privacy being compromised [18].

In applying these noises to control systems, the choice between Laplacian and Gaussian is between the desired privacy level versus the need to maintain the integrity of the system's functionality. Laplacian noise is the go-to for tighter privacy, while Gaussian is preferable when a balance is needed between protecting information and retaining a high degree of data usability.

1.5.2 Homomorphic Encryption

Homomorphic encryption stands out as a groundbreaking technology that enables computations to be carried out on encrypted data without needing to decrypt it first. This form of encryption is particularly beneficial for control systems that leverage cloud computing for data processing and storage. Homomorphic encryption allows for the offloading of computation tasks to third-party service providers without revealing the underlying sensitive data. Recent advancements in lattice-based cryptography have brought practical implementations of homomorphic encryption closer to reality, opening up new possibilities for secure control systems [19]–[21].

1.5.3 Challenges and Opportunities

The integration of these encryption technologies into control systems is not without challenges. Differential privacy must be carefully tailored to the specific use case to maintain the system's

functionality, and homomorphic encryption often comes with significant computational overhead. However, the opportunities they present for secure and private control systems are substantial. Ongoing research continues to refine these methods, making them more accessible and efficient for real-world applications [15], [22], [23].

2 Simulation

In this Chapter, we explore the implementation details of a MPC algorithm applied to a QTS using MATLAB. This part is dedicated to elucidating the intricate workings of the MPC algorithm within the context of a multi-input multi-output (MIMO) system, primarily focused on regulating water levels in four interconnected tanks. MATLAB has been selected as the platform of choice due to its comprehensive mathematical modelling capabilities and its integrated environment, which facilitates a seamless transition from conceptual modelling to practical simulation. The script features a MPC programmed by hand rather than using a black box to visualise the process of the controller and get a better understanding of the process and what information is sent to and received by the MPC. Additionally, MATLAB's advanced visualization tools play a crucial role in the analysis and interpretation of the system's behaviour, enabling a clear understanding of the control strategies' impacts on the system dynamics. This section aims to provide a detailed walkthrough of the MATLAB code, offering insights into the practical aspects of implementing MPC in a complex MIMO system.

2.1 System Parameters & Initial Conditions

The parameters used in [11] are defined in the first section of the script. These parameters entail the cross-sectional areas of the tanks and the holes at the bottom, which are pivotal for determining the system's hydrodynamic behaviour. Accurately capturing these dimensions is essential for modelling the fluid dynamics within the tanks. Gravitational acceleration, a fundamental constant, is specified to enable precise computation of water flow between the upper and lower tanks influenced by gravity. This aspect is vital for simulating the realistic behaviour of fluids within the tanks. Valve positions are described as dimensionless ratios, playing a key role in controlling the flow between interconnected tanks. The chosen valve positions of 0.43 and 0.34 make for the nonminimum-phase characteristics of the system, an aspect critical for understanding and controlling its behaviour. The system's equilibrium state is established with pre-defined tank heights, input voltages, and charge carrier mobility. These initial conditions are set to reflect a typical operational state of the system, providing a baseline for analyzing the impact of various control strategies implemented via the MPC algorithm. The initial deviation in height from the equilibrium is denoted as x_0 and can be chosen manually as long as they are within the constraints described later on.

2.2 Dense LQ

The parameters described earlier are included in the non-linear system dynamics. However, the MPC takes a linear model of the form in (14).

$$\begin{cases} x_{k+1} = Ax_k + Bu_k \\ y_k = Cx_k \end{cases} \tag{14}$$

where $y \in \mathbb{R}^p$ and the matrices A, B and C are defined as in (2) and (3).

The state space representation is the linear form of the system dynamics and forms the backbone of the control model for the QTS. This representation includes the time constants T_i, the areas of the tanks A_i, the valve positions γ_1 and γ_2 and the electric field strength k_c.

These equations describe how the system works. The first part is called matrix A. This matrix represents the system dynamics, correlating the system's current state to its next state. It captures how the internal state variables interact with each other, defining the system's inherent behaviour. Matrix B connects the external inputs to the system's state. It defines how control inputs, in this case, voltage used by the pumps in the QTS, influence the state variables. This matrix is crucial for understanding how external actions impact the system. The matrix C maps the internal state of the system to the observable outputs. It relates the water levels in the tanks to the output. Furthermore, it's important to note that MPC requires a discretized model of the system. While the QTS inherently operates in continuous time, the MPC algorithm works with a discretized version of the state space model. This discretization is essential for the MPC's computational process, allowing it to predict future states and make control decisions at discrete time intervals. This is done using the zero-order hold method.

2.3 Prediction Horizon & Cost Function Matrices in MPC

For the MPC, the prediction horizon plays a crucial role. It defines the future time frame over which the controller predicts the system's behaviour. A well-chosen horizon balances computational complexity with prediction accuracy, ensuring effective control action over an appropriate time scale. The prediction horizon is defined as N. The MPC algorithm employs a cost function to optimize control actions. This function is influenced by matrices Q, R, and P.

2.4 Setup of Matrices for MPC Cost Function

In the development of the MPC algorithm, the setup of specific matrices is fundamental to the formulation of the cost function. The MPC solves a cost function as in (5). The matrices Q, R and P are used in these specific matrices. The optimization problem is rewritten in a dense form in (6).

2.5 Constraint Formulation

Operational constraints are incorporated into the MPC formulation to ensure the system operates within safe and efficient bounds. These include input constraints, which limit the voltage that can be applied to the pumps, and state constraints, which ensure the water levels in the tanks remain within specified limits. This means that the cost function is subject to (15).

$$\begin{cases} u_{\text{min}} \leq u(t) \leq u_{\text{max}} \\ y_{\text{min}} \leq y(t) \leq y_{\text{max}} \end{cases} \tag{15}$$

The constraints are expressed mathematically using inequality expressions as the 'subject to' part of (6).

For the constraints on the inputs, the maximum and minimum values for the input have to be set for both pumps. These values are then integrated in the matrices u_{max} and u_{min} where $u_{\text{max}} \in \mathbb{R}^m$ and $u_{\text{min}} \in \mathbb{R}^m$. Using these matrices, the matrices for the input constraints following from the 'subject to' part of (6) can be constructed. This results in (16).

$$\begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \\ -1 & 0 & \cdots & 0 \\ 0 & -1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & -1 \end{bmatrix} z \leq \begin{bmatrix} u_{\max} \\ u_{\max} \\ \vdots \\ u_{\max} \\ -u_{\min} \\ -u_{\min} \\ \vdots \\ -u_{\min} \end{bmatrix}. \tag{16}$$

For the constraints on the outputs, first the maximum and minimum values for the outputs need to be defined. These values are then again integrated in the matrices $y_{\max}$ and $y_{\min}$ where $y_{\max} \in \mathbb{R}^p$ and $y_{\min} \in \mathbb{R}^p$. Again, when using these matrices, the form of (17) is obtained when using the general form.

$$\begin{bmatrix} CB & 0 & \cdots & 0 \\ CAB & CB & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ CA^{N-1}B & \cdots & CAB & CB \\ -CB & 0 & \cdots & 0 \\ -CAB & -CB & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ -CA^{N-1}B & \cdots & -CAB & -CB \end{bmatrix} z \leq \begin{bmatrix} y_{\max} \\ y_{\max} \\ \vdots \\ y_{\max} \\ -y_{\min} \\ -y_{\min} \\ \vdots \\ -y_{\min} \end{bmatrix} - \begin{bmatrix} CA \\ CA^2 \\ \vdots \\ CA^N \\ -CA \\ -CA^2 \\ \vdots \\ -CA^N \end{bmatrix} x_0 \tag{17}$$

2.6 Optimization Problem

At each sampling instance, the MPC solves a quadratic programming (QP) problem. The QP problem aims to find the optimal sequence of control inputs z over the prediction horizon N, that minimizes a cost function subject to the system dynamics and constraints. The cost function in (5) is derived from the state and control input weighting matrices and includes a quadratic term for control efforts and a linear term for state deviations. This function is solved using the function quadprog. This function finds the optimal value for z for each iteration with respect to the constraints.

2.7 Disturbance

The control system is designed to adjust to the impact of disturbances on the quadruple-tank process. The disturbance matrix E_d is of the size $E_d \in \mathbb{R}^{n \times m}$. The control objective is to counteract the process disturbance d_k. This disturbance represents an external influence on the system that decays exponentially with time. The system aims to be at rest, meaning that the control strategy should effectively nullify the influence of this disturbance over time, stabilizing the system and ensuring that the tanks return to their desired state without any lasting deviation from the steady-state water levels.

2.8 Estimate A, B & R

To infer the system matrices A, B, and R, the following steps are taken in the MATLAB environment. Firstly, the relevant blocks from the transformed F and H matrices are extracted. These blocks represent parts of the system's dynamics over the prediction horizon. For F, the blocks $F_{1,1}$ to $F_{1,n+1}$, each corresponding to a set of state equations at a specific time step within the prediction horizon, are extracted. Similarly, for H, the blocks $H_{1,1}$ to $H_{n+1,1}$ are obtained, each related to the input weighting matrices at each time step.

Using the extracted blocks, the matrix $\mathcal{C}$ is constructed from the first four blocks of F. This matrix is used in the equation

$$A^\top \begin{bmatrix} F_{1,1} & F_{1,2} & \dots & F_{1,n} \end{bmatrix} = \begin{bmatrix} F_{1,2} & F_{1,3} & \dots & F_{1,n+1} \end{bmatrix}, \tag{18}$$

which forms the basis for calculating the transpose of matrix A, $A^\top$. By solving this equation, $A^\top$, and consequently A by transposition are obtained.

Next, matrix B is addressed by setting up the equation

$$B^\top \mathcal{C} = \begin{bmatrix} H_{2,1}^\top & H_{3,1}^\top & \dots & H_{n+1,1}^\top \end{bmatrix}. \tag{19}$$

Solving for $B^\top$ yields the transpose of the input matrix B, which is transposed back to find B. Using the obtained matrix B, the pair $\bar{Y}B$ can be obtained from

$$\begin{bmatrix} A^\top \\ \vdots \\ (A^\top)^{n+1} \end{bmatrix} \bar{Y}B = \begin{bmatrix} F_{1,1} \\ \vdots \\ F_{1,n+1} \end{bmatrix}. \tag{20}$$

Finally, the matrix R can be inferred by the cloud by using the following formula:

$$R + B^\top \bar{Y} B = H_{1,1}. \tag{21}$$

Using the previously calculated $B^\top$ and pair $\bar{Y}B$, we resolve for R.

With these computations, the system matrices are effectively retrieved without direct knowledge of the system's internal structure, solely from the input-output data as encapsulated in the F and H matrices.

2.9 Estimate Q & P

To estimate the system matrices Q and P, the MATLAB environment executes a series of steps. Initially, the code constructs the matrices $\bar{S}$ and $\bar{T}$ by iterating over a prediction horizon N like $\bar{S}$ and $\bar{T}$ were described before but this time the estimated values of A and B are used. The estimation for the block-diagonal weighting matrix R is built as described earlier.

Now the matrices are substituted into (13) to obtain the matrix $\hat{\mathcal{Q}}$. The MATLAB code further processes the solution to extract the matrix $\hat{Q}$ by summing and averaging its block components. The estimated matrix $\hat{P}$ is retrieved from the last block of the solution.

A semidefinite program (SDP) is then formulated and solved using the CVX framework to check for the feasibility and uniqueness of the estimated matrices. The problem constraints are such that a symmetric matrix X must satisfy the inequalities for $\hat{Q}$ and $\hat{P}$ with respect to the discretized system dynamics. The feasibility and the potential uniqueness are examined by

finding the eigenvalues of the matrices involved in the inequalities. The script returns if it is likely that (12) is a singleton.

Through these computations, the MATLAB code effectively estimates the system matrices Q and P, which are essential for the design of optimal control strategies.

2.10 Noise

For the differential privacy, Gaussian noise is applied to the matrices H and F within a control system framework since it keeps a higher level of accuracy in comparison to Laplacian noise. In the context at hand, it is employed to safeguard the information about the system's internal components or states. The degree of privacy is modulated by two pivotal parameters: ϵ and δ. The variable ϵ, known as the privacy budget, inversely controls the level of noise added, the higher the value of ϵ, the smaller the amount of noise. This means that with a larger ϵ, individual data points are less masked, which could reduce privacy. On the other hand, the parameter δ allows for a small chance that privacy could be compromised. These two parameters, ϵ and δ are crucial for calculating the standard deviation, σ, which determines the spread of the Gaussian noise. The sensitivity of the function indicates the maximum possible change in the output for any single change in the input. This sensitivity level is directly related to how much noise is needed to maintain a certain level of privacy. Noise from a Gaussian distribution is created using the 'randn' function which create a correct sized matrix with random numbers and then adjusted by the value of σ to match the chosen privacy settings. This noise is then added to the matrices H and F. To make sure matrix H stays symmetrical, which is a requirement for certain optimization calculations like quadratic programming, the altered H matrix is made symmetrical by averaging it with its transpose. The equation for σ is given below as stated in [15].

$$\sigma = \sqrt{2 \cdot \log\left(\frac{1.25}{\delta}\right)} \cdot \frac{\text{sensitivity}}{\epsilon} \tag{22}$$

3 Results

3.1 Numerical Setup

In this section, the numerical setup foundational to the MPC analysis for the QTS are presented. Essential to the efficacy of the MPC framework is the precise definition and calibration of various parameters. This section details these values and the rationale behind their selection, laying the groundwork for the subsequent analysis of the MPC's performance.

3.1.1 Sample Time

Upon detailed analysis of the quadruple tank system's characteristics and the control objectives, a sample time of $Ts = 2$ has been determined to be the most suitable for discretizing the system. The natural response time of the tank levels to control inputs is relatively slow, which is typical for fluid dynamics systems where changes occur over several seconds or minutes. This delay within the system dynamics allows for a sample time that is sufficiently long to capture the critical dynamics without the need for high-frequency sampling that could result in excessive computational load. Besides this, the control strategy aims to balance responsiveness with stability. A 2-second sample time provides a good compromise, offering timely updates to the controller to react to disturbances, while also allowing for a manageable computational load. This ensures that the control algorithm can be executed within the time constraints, maintaining the real-time requirements of the system.

3.1.2 Prediction Horizon

In the development of the MPC framework for the QTS, careful consideration was given to the selection of the prediction horizon. After a thorough analysis of the system's dynamics and response characteristics, the prediction horizon was set to $N = 20$. This duration was chosen based on several key factors:

- **System Dynamics:** The QTS exhibits certain time-dependent behaviours and response characteristics that require a sufficient window of time to be accurately captured and controlled. A 20-second horizon provides a comprehensive view of the response to various control actions, ensuring that the MPC can effectively anticipate and mitigate fluctuations.
- **Control Objective:** The primary aim of the MPC in this context is to maintain stability and achieve desired operational targets within the QTS. A 20-second horizon aligns well with these objectives, offering an optimal balance between short-term reactivity and long-term planning.
- **Computational Feasibility:** While a longer prediction horizon can potentially lead to more informed control decisions, it also increases computational complexity. The 20-second horizon represents a practical compromise, capturing essential dynamics without imposing excessive computational demands on the system.

For certain analyses this value can still be changed, however for the standard model the prediction horizon is set to $N = 20$.

3.1.3 Cost Matrices

In the model predictive control framework applied to the quadruple tank system, the selection of the cost matrices Q, R and P is paramount to achieving the desired balance between state regulation and control effort. The chosen values for these matrices have been carefully considered to align with the operational objectives and constraints of the system.

- Q **:** This matrix has been determined to ensure that the system states are maintained within a tightly controlled range and is set to $Q = 2I_4$. This weighting, which is twice that of a standard identity matrix, reflects the system's sensitivity to deviations from the desired states. Due to the coupling between the tanks in the quadruple tank system, an increased penalty for state deviations is crucial. This increased penalisation enforces a stricter system response to any deviations from the desired state, thereby securing a significant degree of accuracy in maintaining the desired tank levels.
- R **:** The control effort matrix $R = I_2$ represents a balanced approach, penalising excessive control input while still allowing for the necessary adjustments to be made by the actuators. This matrix was chosen to reflect a linear relationship between control effort and associated costs. The identity matrix indicates that the effort of each actuator is equally weighted, which is appropriate given the similar roles and influence on system behaviour.
- P **:** Lastly, the terminal cost matrix $P = 0_4$ is indicative of the emphasis on the state trajectory over the prediction horizon rather than the terminal state. This approach is justified by the system's design for steady-state operation, negating the necessity for a terminal state penalty.

3.1.4 Constraints

In the MPC framework for the QTS, input and output constraints play an important role in ensuring the feasibility of the system operation. For the input, the two pumps are constrained at both the upper and lower levels. The constraints are determined so that the actuators do not receive unrealistically high or low voltage inputs to solve sudden changes in the system. This reason is why the constraints are set to deviate 1 volt from the voltage applied at steady-state. This gives the following matrices for u_{max} and u_{min}:

$$u_{\text{max}} = \begin{bmatrix} 1 \\ 1 \end{bmatrix},$$

$$u_{\text{min}} = \begin{bmatrix} -1 \\ -1 \end{bmatrix}.$$

The output constraints are necessary to keep the water levels within the limits of the tanks and make sure that the system does not allow the levels to deviate too much from the steady-state levels. Because of this reason, the constraints are set so that the water levels in the output tanks (which are tanks 1 and 2) cannot deviate more than 1 centimetre from their steady-state values. This results in the following matrices for y_{max} and y_{min}:

$$y_{\text{max}} := \begin{bmatrix} 1 \\ 1 \end{bmatrix},$$

$$y_{\min} := \begin{bmatrix} -1 \\ -1 \end{bmatrix}.$$

3.1.5 Disturbance

As described earlier, the control objective is to react to and overcome the effects of the disturbance d_k. In this case $d_k = [2 \quad -3]^\top (1/2)^k$ for $k \geq 12$. The disturbance matrix E_d should be constructed so that the system is optimally tested. This is for instance done by suddenly creating a surplus of water in the left side tanks and a shortage of water in the right side tanks. However, the sudden input should not conflict with the constraints. This results in the following matrix:

$$E_d = \begin{bmatrix} 0.1 & -0.1 \\ -0.1 & 0.1 \\ 0.1 & -0.1 \\ -0.1 & 0.1 \end{bmatrix}.$$

3.2 Plots

Using the numerical setup described above, the system is simulated over time. This is done in MATLAB over a period of 200 seconds. The system is set to start under steady-state conditions. Both the water levels in all four tanks in centimeters as well as the voltage applied to the two pumps are plotted over this period. This results in the plots in figure 6 & 7.

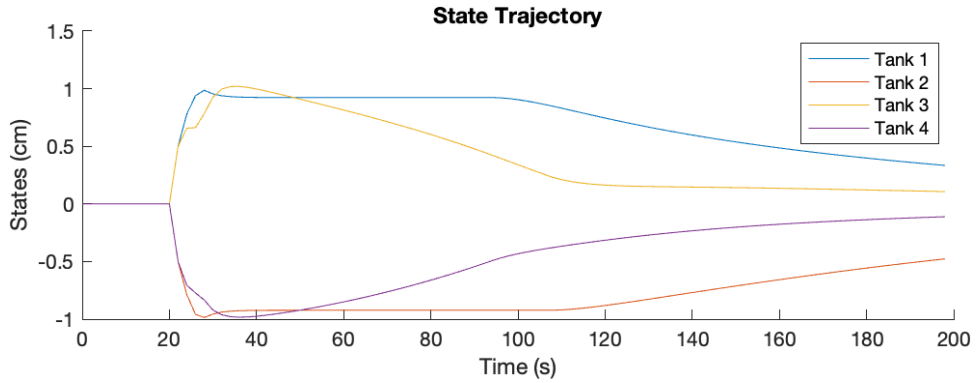

Figure 6: Water Levels over Time

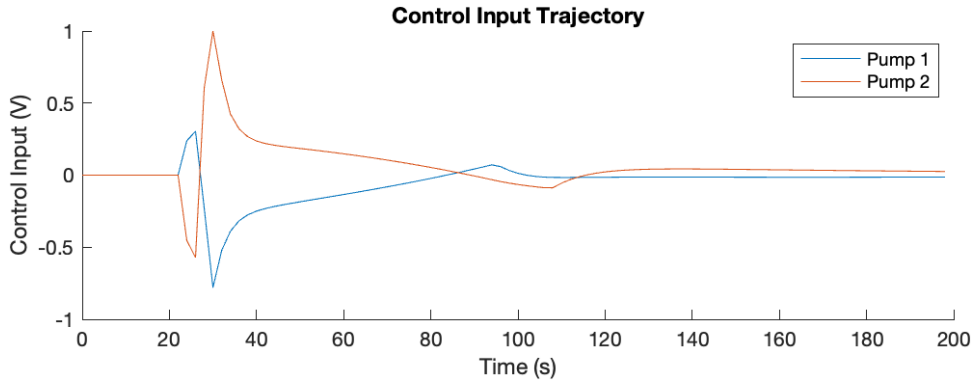

Figure 7: Applied Voltages to Pumps over Time

3.3 Estimations of Matrices A, B & R

The estimations of the system matrices A, B, and R have been conducted following the methodologies outlined in Chapter 2.8. To assess the performance and robustness of the estimations, we considered various prediction horizons. Specifically, we evaluated the estimations of the three matrices for prediction horizons $N = 5$, $N = 10$, $N = 20$, $N = 30$, $N = 40$ and $N = 100$. The outcomes are compared by using the Frobenius norm per estimated matrix per prediction horizon. The Frobenius norm, often used to measure the error of a matrix, is defined as the square root of the sum of the absolute squares of its elements. For an $m \times n$ matrix X, with elements x_{ij}, the Frobenius norm $||X||_F$ is expressed by the equation:

$$||X||_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}|x_{ij}|^2} \tag{23}$$

This norm provides a scalar measure of the magnitude of a matrix, which in the context of estimation errors, quantifies how much an estimated matrix deviates from the true matrix. The smaller the Frobenius norm, the closer the estimated matrix is to the true matrix, implying higher estimation accuracy. The outcomes for the matrices A, B, and R are given in the table below.

N	A	B	R
5	11.214	1.7454	8.8326e-04
10	1.3466	0.2178	1.7144e-04
20	0.3188	0.0579	6.8554e-06
30	0.1562	0.0337	1.7833e-06
40	0.1017	0.0248	1.1150e-06
100	0.0315	0.0141	7.4775e-08

Table 2: Frobenius Norm of the Error for Matrices A, B and R

The Frobenius norm results presented in Table 2 provide insightful data on the estimation accuracy for the matrices A, B, and R of the quadruple-tank system as a function of the sample size N. As N increases, we observe a clear trend of diminishing error norms, indicative of enhanced estimation precision. Particularly, for matrix A, the decrease in error norm from $N = 5$

to $N = 100$ is substantial, suggesting that a longer prediction horizon significantly improves the fidelity of the system model. Matrices B and R exhibit a similar trend, however, with a less pronounced decrease, which may be due to their respective roles and sensitivity in the MPC algorithm. Based on the assumptions delineated in Section 1.4, it is inferred that the prediction horizon asymptotically approaches infinity. This implies that, under these conditions, the accuracy of estimations is expected to surpass those obtained with a finite prediction horizon of $N = 100$.

The above estimations provide valuable insights into the cloud's ability to estimate the systems matrices when matrix H and F are known by the cloud.

3.4 Estimations of Matrices Q & P

Through the computational experiments conducted, it has become evident that the characterisation of the set $\mathcal{J}$, as defined by the conditions previously outlined, exhibits non-singularity when the estimated matrices A, B, and R are used. This results in the computational results not being aligned with the expectations.

When looking closely at the systems response to using the matrices A, B and R, it becomes clear that the set $\mathcal{J}$ in (12) does not become a singleton. This means that Q and P cannot be inferred using the current methods and system settings. Since the set is not a singleton, there are several possible answers, and finding one unique pair for the matrices Q and P becomes impossible.

The implications of these results are twofold. On the one hand, they highlight the sensitivity of the system to the accuracy of the underlying matrices. On the other hand, they unveil the complexity of ensuring the singularity of the set $\mathcal{J}$, which is a precondition for finding Q and P. Future investigations are necessitated to unravel the factors contributing to this phenomenon and to devise strategies that can guarantee the singularity of the set, thereby obtaining matrices Q and P.

3.5 Estimation of Matrices A, B & R with noise

Noise is added as described in Chapter 2.10 with the values $\epsilon = 10$, $\delta = 0.5$ and sensitivity set to 1. The noise is randomly added so no simulation will return the same figure. Below a figure for $N = 20$ is provided with a longer simulation time of 400 seconds to visualise that the system never reaches equilibrium due to the noise.

The figure shows that with noise, the states do not stay within the boundaries which are removed for this case. With the constraints the problem is infeasible. The estimations of the matrices A, B & R are revisited after adding noise to the matrices H and F as described in Chapter 2.10. However, this time the matrices H and F do not represent their true matrices because of the noise. Because of this, the Frobenius norm for the error for these matrices is also provided. The same prediction horizons are used and depicted in the table below.

As seen in the table, the matrices A, B and R are not properly estimated anymore. This is mainly because the matrices H and F are also not representing the system accurately.

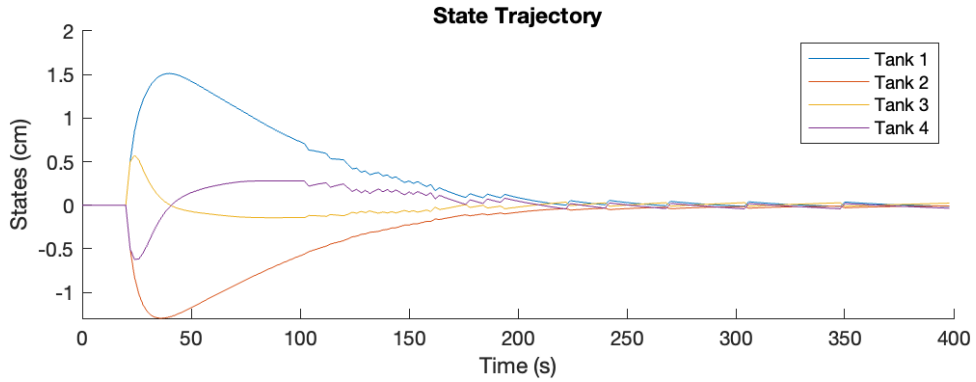

Figure 8: Water Levels with Noise over Time

N	A	B	R	H	F
5	2.8503	0.4021	0.6982	1.0295	0.7556
10	2.2896	0.6175	0.2936	2.0151	1.1176
20	2.4216	0.8124	2.9052	3.9045	1.6360
30	2.0428	0.3845	0.1484	5.6700	2.2126
40	2.8211	0.6490	12.3712	7.6184	2.7050
100	2.1487	0.5628	0.2873	19.0710	3.9386

Table 3: Frobenius Norm of the Error for Matrices A, B, R, H and F

3.6 Privacy Preservation in Control Systems

The two main ways to keep control systems private and secure are using differential privacy and homomorphic encryption. After looking into both, homomorphic encryption is a better choice for this case, even though the implementation of such encryption is not possible in MATLAB.

Differential privacy works by adding a bit of randomness to the data. This helps to hide the details about individuals in the data. But this can also make the data less accurate, which would not be favourable for control systems.

Homomorphic encryption is like locking data in a safe and still being able to use it without unlocking the safe. This method keeps data secure, even when it is being used or processed. It does not change or add anything to the data, so the results are precise. This accuracy is very important for control systems that need exact information to work properly. For the case of the quadruple-tank system, it would be of high importance to have the precise results back from the MPC because the optimal control inputs are desired. One big problem with homomorphic encryption is that it needs relatively much computing power when compared to differential privacy. Besides this, homomorphic encryption requires some special programming that MATLAB does not have. MATLAB is not set up to handle the complex math needed for homomorphic encryption.

4 Conclusion

4.1 Comprehensive Summary of Research

The research project "Outsourced Private Model Predictive Control" primarily revolved around the development of a sophisticated MATLAB code designed to facilitate the outsourcing of model predictive control to cloud services. This coding effort was directed towards creating a simulation environment for a quadruple-tank system that could effectively simulate the physical system and shed light on the possible privacy issues that come with outsourcing a MPC. The core of the research entailed the construction and refinement of this MATLAB code, ensuring its capability to simulate the QTS controlled by a MPC. The physical system has been simulated realistically and with a working MPC that controls the system effectively.

Following the development of the MATLAB framework, the research delved into the inference of data by the cloud. The matrices A, B and R were obtained by cloud which violates the privacy of the system. The matrices Q and P could not be inferred because the cloud could not obtain unique solutions for the matrices.

Finally, two pivotal cryptographic methods were examined to overcome privacy breaches: differential privacy and homomorphic encryption. The primary objective was to assess their applicability and efficiency in protecting data privacy within the cloud-computing environment of MPC. Differential privacy was explored for its potential to add statistical noise to data, thereby safeguarding sensitive information. The precision of this method turned out to be unsatisfactory not returning accurate system inputs while the data its privacy was maintained better than before as can be seen in Chapter 3.5. Homomorphic encryption was investigated for its ability to perform computations on encrypted data only in a theoretical way because of the limitations of implementing this method in MATLAB. In theory, the homomorphic method should provide a better solution to the problem.

4.2 Impact and Significance of Findings

The research findings on the outsourcing of MPC to cloud services have significant implications for the field of industrial engineering and cloud-based MPC systems. The simulation of how certain data can be obtained by only sending little pieces of data, in a MATLAB-based simulation environment paves the way for enhanced data security in industrial control systems. This breakthrough is particularly vital in today's era, where the integration of cloud computing with industrial processes is becoming increasingly relevant. The findings provide a valuable framework for industries to leverage cloud computing while ensuring data privacy, potentially revolutionizing the way industrial control systems are managed and operated. The research thus marks a significant step forward in balancing operational efficiency with stringent security measures in cloud-based industrial environments.

4.3 Contributions to the Field

This research significantly advances the understanding of data privacy in cloud-based MPC systems, particularly through the MATLAB simulation of a quadruple-tank system. By addressing potential privacy breaches, such as the inference of matrices A, B, and R by the cloud, this work highlights crucial security considerations in cloud computing for industrial applications.

This focus on the practical aspects of data privacy in a real-world physical system simulation enriches the existing body of knowledge, offering valuable insights for the secure integration of cloud technology in control systems.

5 Discussion

5.1 Critical Analysis of Findings

The research findings from the MATLAB simulation of the quadruple-tank system offer several strengths, notable weaknesses, and some unexpected results that can lead to further discussion.

5.1.1 Strengths

The main strength of this study lies in its rigorous approach to simulating a complex physical system within a MATLAB environment. The use of a well-established computational platform provided a robust foundation for accurate system modelling and analysis. Additionally, the discussion of potential privacy breaches, specifically regarding the inference of matrices A, B, and R, highlights the study's comprehensive consideration of privacy issues within cloud-based MPC systems.

5.1.2 Weaknesses

The integration of differential privacy into model predictive control systems presents a unique set of challenges that can significantly affect the performance and optimality of the control strategy. Differential privacy, by design, introduces noise into the system. However, this added noise can have adverse effects on the predictive accuracy of the MPC, as the controller relies on precise predictions of future system states to calculate optimal control actions.

The essence of MPC is to optimize control inputs over a defined horizon based on accurate model predictions. When differential privacy is applied, the noise introduced into the system's data can lead to inaccurate state estimates and predictions. Consequently, the MPC may compute suboptimal control actions that deviate from the true optimal path. The magnitude of this deviation is directly proportional to the intensity of the noise: higher levels of privacy (lower values of the privacy budget ϵ) result in greater noise and hence, larger deviations from the optimal control trajectory.

Besides these two points, the problem automatically becomes infeasible when the noise is added. This is because the water levels in tanks 1 and 2 deviate more than 1 centimetre from the steady-state levels. Because of this, the constraints are temporarily removed and are not maintained as described earlier.

Another one of the limitations of the current study is the lack of empirical testing of the cryptographic method of homomorphic encryption. Although this method was mentioned and its potential impact discussed, empirical evidence to support the theoretical findings would strengthen the study's conclusions.

5.1.3 Unexpected Results

Contrary to expectations, the study encountered difficulties in accurately estimating the matrices Q and P for all prediction horizons. This was an unexpected outcome, as it was initially

hypothesized that the simulation would yield these matrices with more precision across varying values of N. The root of this challenge was traced back to the use of estimated values within the MATLAB simulation, which did not maintain consistent accuracy for all N. The discrepancy raises questions about the sensitivity of the MPC algorithm to the precision of computed values. It suggests that the algorithm's robustness may be not fully accounted for in the simulation. This finding points to further investigation into the stability of the MPC algorithm's parameters and their influence on the system's predictive accuracy. Besides this, the set provided in (12) turned out to be unable to find a unique solution for the matrices Q and P even when using the true values for the matrices required in the estimation because of the lack of singularity.

5.1.4 Implications of the Analysis

The critical analysis of these findings underscores the necessity for a careful and balanced approach to cloud-based MPC system design. It emphasizes the importance of considering the trade-offs between computational feasibility and privacy, as well as the need for rigorous testing in conditions that closely mimic the complexities of real-world operations.

5.2 Limitations and Challenges

Throughout this research, several limitations and challenges were encountered, which are important to acknowledge as they provide context for the findings and suggest directions for future work.

Firstly, the simulation of the quadruple-tank system within a MATLAB environment, while robust, may not fully encapsulate the complexities and unpredictable nature of real-world physical systems. An example of this is the fact that the system is discretized. The simplifications required for computational modelling can lead to discrepancies between the simulated and actual systems. This represents a limitation in terms of the external validity and generalizability of the research findings.

Another challenge was the theoretical nature of the discussion surrounding the cryptographic methods of differential privacy and homomorphic encryption. Without empirical testing and validation of the homomorphic encryption within the QTS context, the conclusions drawn about its efficacy and practicality remain speculative. Future research should aim to implement this cryptographic technique in a practical setting to evaluate its real-world applicability.

Additionally, a notable challenge faced during the research was the substantial amount of time required to build the simulation model. Despite prior experience with MATLAB, the complexity and the iterative nature of developing an accurate and reliable simulation for the QTS extended beyond initial time estimates. The process of translating the physical system into a computational model involved numerous adjustments and refinements to ensure that the simulation accurately reflected the dynamics of the QTS controlled by a MPC.

Each of these limitations and challenges points to the need for continued research in the field, particularly with a focus on practical implementation and empirical testing to validate the theoretical models and simulations presented in this study.

5.3 Future Research Opportunities

The developed MATLAB model for the quadruple-tank system serves as a robust platform for future research, particularly in the domain of data privacy within cloud-based model predictive control systems. A significant opportunity for subsequent studies is to utilize the model to investigate how the knowledge of certain matrices could potentially lead to the inference of other system matrices.

Another primary opportunity involves the empirical testing of the homomorphic encryption, within the quadruple-tank system environment. Real-world implementation and testing would provide valuable data on its practicality and effectiveness in ensuring data privacy in cloud-based MPC systems.

Moreover, exploring alternative methods for securing data privacy, beyond the scope of differential privacy and homomorphic encryption, could yield novel solutions to the privacy concerns in cloud-based MPC. This could include investigating newer cryptographic techniques or developing proprietary methods tailored to the specific needs of industrial control systems.

5.4 Practical Implications and Industrial Applications

The development of a MATLAB model for simulating the quadruple-tank system has several immediate practical implications and potential applications within the field of industrial engineering, particularly in the design and operation of cloud-based model predictive control systems.

The insights gained from this research regarding the potential privacy breaches, such as the inference of matrices A, B, and R, are crucial for enhancing the security protocols of cloud-based MPC systems. Industries employing cloud-based MPC can use these findings as an example to eventually implement more robust privacy measures, ensuring that sensitive control system data remains protected against unauthorized access and inference attacks.

Acknowledgments

I would like to express my gratitude to my daily supervisor, Teimour Hosseinalizadeh, for his valuable support throughout this research. His willingness to answer questions and provide guidance on the technical aspects of this project has been essential in my learning and progress. His expertise and insights have significantly contributed to the completion of this research.

Additionally, I would like to thank my supervisors, Professor Nima Monshizadeh Naini and Dr. Alexander Hübl, for their time and effort in evaluating and guiding this project. Their expertise and constructive feedback have been crucial in shaping the research.

Their collective guidance has not only enhanced my academic experience but has also provided me with a deeper understanding of the field, for which I am grateful.

References

[1] MIT Technology Review Insights, "2023 global cloud ecosystem," *MIT Technology Review*, Nov. 2023.

[2] D. Q. Mayne, "Model predictive control: Recent developments and future promise," *Automatica*, vol. 50, no. 12, pp. 2967–2986, 2014.

[3] A. B. Alexandru, M. Morari, and G. J. Pappas, "Cloud-based mpc with encrypted data," in *2018 IEEE Conference on Decision and Control (CDC)*, 2018, pp. 5014–5019.

[4] C. Dwork, "Differential privacy: A survey of results," in *Theory and Applications of Models of Computation*, M. Agrawal, D. Du, Z. Duan, and A. Li, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 1–19.

[5] M. Tebaa, S. E. Hajji, and A. E. Ghazi, "Homomorphic encryption method applied to cloud computing," in *2012 National Days of Network Security and Systems*, 2012, pp. 86–89.

[6] D. A. Vijula, K. Anu, P. Honey, and P. S. Poorna, "Mathematical modelling of quadruple tank system," *International Journal of Emerging Technology and Advanced Engineering*, vol. 3, no. 12, 2013.

[7] I. Alvarado, D. Limon, W. García-Gabín, T. Alamo, and E. Camacho, "An educational plant based on the quadruple-tank process," *IFAC Proceedings Volumes*, vol. 39, no. 6, pp. 82–87, 2006, 7th IFAC Symposium on Advances in Control Education.

[8] K. Johansson, A. Horch, O. Wijk, and A. Hansson, "Teaching multivariable control using the quadruple-tank process," vol. 1, 807–812 vol.1, 1999.

[9] B. Ashok Kumar, R. Jeyabharathi, S. Surendhar, S. Senthilrani, and S. Gayathri, "Control of four tank system using model predictive controller," in *2019 IEEE International Conference on System, Computation, Automation and Networking (ICSCAN)*, 2019, pp. 1–5.

[10] J. Cortés, G. E. Dullerud, S. Han, J. Le Ny, S. Mitra, and G. J. Pappas, "Differential privacy in control and network systems," in *2016 IEEE 55th Conference on Decision and Control (CDC)*, 2016, pp. 4252–4272.

[11] K. Johansson, "The quadruple-tank process: A multivariable laboratory process with an adjustable zero," *IEEE Transactions on Control Systems Technology*, vol. 8, no. 3, pp. 456–465, 2000.

[12] *Model Predictive Control — Institute for Systems Theory and Automatic Control — University of Stuttgart — ist.uni-stuttgart.de*, `https://www.ist.uni-stuttgart.de/research/group-of-frank-allgoewer/model-predictive-control/`.

[13] T. Hosseinalizadeh, N. Schlüter, M. S. Darup, and N. Monshizadeh, *Privacy analysis of affine transformations in cloud-based mpc: Vulnerability to side-knowledge*, 2024. arXiv: `2401.05835 [eess.SY]`.

[14] C. Dwork, "Differential privacy," in *Proceedings of the 33rd International Conference on Automata, Languages and Programming - Volume Part II*, ser. ICALP'06, Berlin, Heidelberg: Springer-Verlag, 2006.

[15] C. Dwork and A. Roth, "The algorithmic foundations of differential privacy," *Foundations and Trends® in Theoretical Computer Science*, vol. 9, no. 3–4, 2014.

[16] A. Tandon, *Differential privacy*, Oct. 2019.

[17] P. Jain, M. Gyanchandani, and N. Khare, "Differential privacy: its technological prescriptive using big data," *Journal of Big Data*, vol. 5, no. 1, Apr. 2018.

[18] J. Dong, A. Roth, and W. J. Su, "Gaussian Differential Privacy," *Journal of the Royal Statistical Society Series B: Statistical Methodology*, vol. 84, no. 1, pp. 3–37, Feb. 2022. eprint: `https://academic.oup.com/jrsssb/article-pdf/84/1/3/49324238/jrsssb\_84\_1\_3.pdf`.

[19] C. Gentry, "A fully homomorphic encryption scheme," `crypto.stanford.edu/craig`, Ph.D. dissertation, Stanford University, 2009.

[20] Z. Brakerski and V. Vaikuntanathan, "Lattice-based fhe as secure as pke," ser. ITCS '14, Princeton, New Jersey, USA: Association for Computing Machinery, 2014.

[21] V. Vaikuntanathan, "Computing blindfolded: New developments in fully homomorphic encryption," in *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, Los Alamitos, CA, USA: IEEE Computer Society, Oct. 2011.

[22] C. Dwork, "A firm foundation for private data analysis," *Communications of the ACM*, vol. 54, no. 1, pp. 86–95, 2011.

[23] C. Gentry, "Computing arbitrary functions of encrypted data," *Commun. ACM*, vol. 53, no. 3, pp. 97–105, Mar. 2010.

Appendices

A MATLAB Script

```
clear; clc;

%% Parameters

% Constants and Parameters
Area = [28 32 28 32];                          % Area of the
   tanks in cm^2
a = [0.071 0.057 0.071 0.057];                 % Area of the
   holes at the bottom of each tank in cm^2
g = 981;                                       % Gravitational
   acceleration in cm/s^2
gamma = [0.43 0.34];                           % Valve positions,
    dimensionless (ratio)
kc = 0.5;                                      % Gain factor in V
   /cm
k = [3.14 3.29];                               % Pump constants
   in cm^3/V*s

% Initial state
h0 = [12.6; 13; 4.8; 4.9];                     % Initial height
   in cm
x0 = [0; 0; 0; 0];                           % Initial x

%% State Space Equations

% State-Space Equations
T1 = 63;                                       % Time constant
   for tank 1 in s
T2 = 91;                                       % Time constant
   for tank 2 in s
T3 = 39;                                       % Time constant
   for tank 3 in s
T4 = 56;                                       % Time constant
   for tank 4 in s

% Continuous-time system matrices
A = [-1/T1 0 Area(3)/(Area(1)*T3) 0;           % System matrix A
   in continuous time
        0 -1/T2 0 Area(4)/(Area(2)*T4);
        0 0 -1/T3 0;
        0 0 0 -1/T4];
```

```
B = [(gamma(1)*k(1))/Area(1) 0;                    % Input matrix B
   in continuous time
        0 (gamma(2)*k(2))/Area(2);
        0 ((1-gamma(2))*k(2))/Area(3);
        ((1-gamma(1))*k(1))/Area(4) 0];

C = [kc 0 0 0;                                     % Output matrix C
   in continuous time
        0 kc 0 0];

D = zeros(2,2);

% Continuous-time state-space system
sys_ct = ss(A, B, C, D);                           % Creating a
   continuous-time state-space object

% Discretize system
Ts = 2;                                            % Sample time in s
sys_dt = c2d(sys_ct, Ts, 'zoh');                   % Discretizing the
    system with zero-order hold method

% Extracting discrete-time matrices
Ad = sys_dt.A;                                     % System matrix A
   in discrete time
Bd = sys_dt.B;                                     % Input matrix B
   in discrete time
Cd = sys_dt.C;                                     % Output matrix C
   in discrete time

% Assuming the disturbance affects tanks 1 and 2
Ed = [0.1 -0.1;                                       % Disturbance
   matrix E
        -0.1 0.1;
        0.1 -0.1;
        -0.1 0.1];

%% Setup MPC

% Define the prediction horizon and control horizon
N = 20; % Prediction horizon

% Get number of states and inputs
n = size(Ad,1);                                  % Number of states
m = size(Bd,2);                                  % Number of inputs
p = size(Cd,1);                                  % Number of outputs
```

```
% Create Q, R and P matrix
Q = diag([2, 2, 2, 2]);                          % State weighting
   matrix
R = diag([1, 1]);                                % Control
   weighting matrix
P = diag([0, 0, 0, 0]);                          % Terminal state
   weighting matrix

%% Setup Matrices

% Initialize S and T matrices
S_bar = zeros(N*n, m*N);
T_bar = zeros(N*n, n);

% Fill S and T matrices
for i = 1:N
    % Build S bar matrix block row
    for j = 1:i
        S_bar((i-1)*n+1:i*n, (j-1)*m+1:j*m) = Ad^(i-j)*Bd;
    end
    % Build T bar matrix
    T_bar((i-1)*n+1:i*n, :) = Ad^(i-1);
end

% Create a cell array with N-1 Q matrices
Q_cells = repmat({Q}, 1, N-1);

% Add the P matrix as the last element in the cell array
Q_cells{end+1} = P;

% Create Q bar and R bar
Q_bar = blkdiag(Q_cells{:});
R_bar = blkdiag(kron(eye(N), R));

% Create H, F and Y matrices
H = (R_bar + S_bar'*Q_bar'*S_bar);
F_trans = T_bar'*Q_bar*S_bar;
Y = (Q + T_bar'*Q_bar*T_bar);

%% Constraints

% Input constraints
u_min = [-1; -1];
u_max = [1; 1];
```

```
% Create G matrix for inputs
G_u_upper = eye(m*N);
G_u_lower = -eye(m*N);

% Create W matrix for inputs
W_u_upper = repmat(u_max, N, 1);
W_u_lower = repmat(-u_min, N, 1);

G_u = [G_u_upper; G_u_lower];
W_u = [W_u_upper; W_u_lower];

% Output constraints
y_min = [-04616; -04616];                                    % Lower
   bound for all outputs
y_max = [04616; 04616];                                      % Upper
   bound for all outputs

% Initialize the matrix G_y with zeros
G_y_upper = zeros(p*N, m*N);

% Construct the controllability matrix
for i = 1:N
    for j = 1:N
        if i == j
            % Diagonal blocks are C*B
            G_y_upper(p*i-round(p/2):p*i, m*j-round(p/2):m*j)
               = Cd * Bd;
        elseif i < j
            % Upper triangular blocks are zeros
            G_y_upper(p*i-round(p/2):p*i, m*j-round(p/2):m*j)
               = zeros(p, m);
        else
            % Lower triangular blocks are C*A^(i-j)*B
            G_y_upper(p*i-round(p/2):p*i, m*j-round(p/2):m*j)
               = Cd * Ad^(i-j) * Bd;
        end
    end
end

G_y_lower = -G_y_upper;

% Create the repeated y_max and y_min vector
y_max_matrix = repmat(y_max, N, 1);
y_min_matrix = repmat(y_min, N, 1);

% Initialize the matrix
```

```
CA_powers_matrix = zeros(p*N, n);

% Calculate each Cd * Adˆi and assign it to the matrix
for i = 1:N
    CA_powers_matrix(p*i-round(p/2):p*i, :) = Cd * (Adˆi);
end

%% Differential Privacy

% Parameters for differential privacy
epsilon = 10; % The privacy parameter
delta = 0.5; % Additional parameter for Gaussian noise
sensitivity = 1; % The sensitivity of the function

% Scale of Gaussian noise
sigma = sqrt(2 * log(1.25/delta)) * sensitivity / epsilon;

% Add Gaussian noise to H and F
H_noisy = H + sigma * randn(size(H));
F_noisy = F_trans + sigma * randn(size(F_trans));

H_noisy = (H_noisy + H_noisy') / 2;

% Compute error matrices
E_H = H - H_noisy;
E_F = F_trans - F_noisy;

% Compute Frobenius norm
frob_norm_H = norm(E_H, 'fro');
frob_norm_F = norm(E_F, 'fro');

%% Simulation Loop

nSteps = 200;

xHistory = zeros(size(x0, 1), nSteps);
xHistory(:, 1) = x0;

for k = 2:nSteps

    % Perform the subtraction
    W_y_upper = y_max_matrix - (CA_powers_matrix * x0);
    W_y_lower = -y_min_matrix + (CA_powers_matrix * x0);

    G_y = [G_y_upper; G_y_lower];
    W_y = [W_y_upper; W_y_lower];
```

```
    G = [G_u; G_y];
    W = [W_u; W_y];

    % Define the disturbance d_k for k >= 12
    if k >= 12
        d_k = [2; -3] * (1/2)^(k-12);
    else
        d_k = [0; 0];
    end

    % Solve the MPC optimization problem to get the optimal
       control input
    [z] = Noise_Function(x0, H_noisy, F_noisy, Bd, G, W);

    % Apply the control input to the system
    x_next = Ad*x0 + Bd*z + Ed*d_k;

    % Update the system state
    x0 = x_next;

    % Store data for analysis and plotting
    xHistory(:, k) = x_next;
    zHistory(:, k) = z;
end

%% Plot Results

% Adjusted time vector
timeVector = Ts * (0:nSteps-1);

% Plot the results
figure;

% Plot for State Trajectory
subplot(2,1,1);
hold on; % Hold on to the current plot

% Assuming xHistory has multiple rows for different states
for i = 1:size(xHistory, 1)
    plotHandle = plot(timeVector, xHistory(i, :));
    set(plotHandle, 'DisplayName', ['Tank ' num2str(i)]); %
       Label each line
end

hold off; % Release the hold after plotting all states
```

```
title('State Trajectory');
xlabel('Time (s)');
ylabel('States (cm)');
legend('show'); % Display the legend

% Plot for Control Input Trajectory
subplot(2,1,2);
hold on; % Hold on to the current plot

% Assuming zHistory has multiple rows for different control
   inputs
for i = 1:size(zHistory, 1)
    plotHandle = plot(timeVector, zHistory(i, :));
    set(plotHandle, 'DisplayName', ['Pump ' num2str(i)]); %
       Label each line
end

hold off; % Release the hold after plotting all control inputs
title('Control Input Trajectory');
xlabel('Time (s)');
ylabel('Control Input (V)');
legend('show'); % Display the legend

%% Estimate A, B, R

% Get blocks from F
for i = 1:n+1
    F_blocks{i} = F_noisy(1:n, (i-1)*m+1:i*m);
end

% Get blocks from H
for i = 1:n+1
    H_blocks{i} = H_noisy((i-1)*m+1:i*m, 1:m);
end

% Build C
CF = [F_blocks{1}, F_blocks{2}, F_blocks{3}, F_blocks{4}];

% Build right hand side in A.28
F_trans_12_15 = [F_blocks{2}, F_blocks{3}, F_blocks{4},
   F_blocks{5}];

% Calculate right hand side in A.29
H_21_51_trans = [H_blocks{2}', H_blocks{3}', H_blocks{4}',
   H_blocks{5}'];
```

```
% Solve for A^T
Ad_trans_calc = F_trans_12_15 * pinv(CF);
Ad_calc = Ad_trans_calc';

% Solve for B^T
Bd_trans_calc = H_21_51_trans * pinv(CF);
Bd_calc = Bd_trans_calc';

% Build left column matrix in A.27
for i = 1:n+1
    % Build T bar matrix
    A_trans_power_col((i-1)*n+1:i*n, :) = (Ad_calc')^i;
end

% Build right hand side of A.27
F_trans_11_15_col = [F_blocks{1}; F_blocks{2}; F_blocks{3};
   F_blocks{4}; F_blocks{5}];

% Solve for Y_bar
Y_bar_Bd_calc = pinv(A_trans_power_col) * F_trans_11_15_col;

R_calc = H_blocks{1} - Bd_trans_calc * Y_bar_Bd_calc;

% Compute error matrices
E_A = Ad - Ad_calc;
E_B = Bd - Bd_calc;
E_R = R - R_calc;

% Compute Frobenius norm
frob_norm_A = norm(E_A, 'fro');
frob_norm_B = norm(E_B, 'fro');
frob_norm_R = norm(E_R, 'fro');

%% Estimate Q, P

for i = 1:N
    % Build S bar matrix block row
    for j = 1:i
        S_bar_calc((i-1)*n+1:i*n, (j-1)*m+1:j*m) = Ad_calc^(i-
           j)*Bd_calc;
    end
    % Build T bar matrix
    T_bar_calc((i-1)*n+1:i*n, :) = Ad_calc^(i-1);
end

R_bar_calc = blkdiag(kron(eye(N), R_calc));
```

```
S_bar_T_bar_calc_trans = [S_bar_calc'; T_bar_calc'];

Q_P_hat_calc = pinv(S_bar_T_bar_calc_trans) * (1/2) * [(H-2*
   R_bar_calc); F_trans] * pinv(S_bar_calc);

Sum_Q_hat_calc = zeros(n);

% Extract and sum all Q_hat_calc matrices from the block
   diagonal
for i = 1:N-1
    Current_Q_hat_calc = Q_P_hat_calc((i-1)*n+1:i*n, (i-1)*n
       +1:i*n);
    Sum_Q_hat_calc = Sum_Q_hat_calc + Current_Q_hat_calc;
end

% Calculate the average of the Q_hat_calc matrices
Q_hat_calc = Sum_Q_hat_calc / (N-1);

% Extract P_hat_calc from the block diagonal
P_hat_calc = Q_P_hat_calc(end-n+1:end, end-n+1:end);

cvx_begin sdp
    variable X(n,n) symmetric

    % Define the constraints for the feasibility problem
    subject to
        Q + X - Ad' * X * Ad >= 0;
        P + X >= 0;
        X * Bd == 0;
cvx_end

% Display the status of the problem
disp(cvx_status);

% If the problem is feasible, check for uniqueness
if strcmp(cvx_status, 'Solved')
    % Check if the solution is on the boundary of the feasible
       set
    eigenvalues_Q = eig(Q_hat_calc + X - Ad_calc' * X *
       Ad_calc);
    eigenvalues_P = eig(P_hat_calc + X);

    % Count the number of small positive eigenvalues close to
       zero
    threshold = 1e-5;
```

```
    num_small_eigenvalues_Q = sum(eigenvalues_Q > 0 &
       eigenvalues_Q < threshold);
    num_small_eigenvalues_P = sum(eigenvalues_P > 0 &
       eigenvalues_P < threshold);

    % If there are small positive eigenvalues, the solution
       may not be unique
    if num_small_eigenvalues_Q > 0 || num_small_eigenvalues_P
       > 0
        disp('The solution may not be unique.');
    else
        disp('The solution is likely unique.');
    end
else
    disp('The problem is not feasible, or it is unbounded.');
end
```

B MATLAB Function

```
function [z] = Noise_Function(x0, H, F_trans, Bd, G, W)

    % Quadratic term in the cost function
    f = (x0'*F_trans)';

    % Options and initial point
    options = optimoptions('quadprog', 'Display', 'iter', '
       Algorithm', 'interior-point-convex');

    % Solve the problem
    [z] = quadprog(H, f, G, W, [], [], [], [], [], options);

    % Extract the optimal control input
    z = z(1:size(Bd,2));                                % Only
       apply the first input

end
```