NTNU
Norwegian University of
Science and Technology

Nonlinear Model Predictive Control of Gravity Separators

Melissa Florine Dlima

Chemical Engineering
Submission date: June 2017
Supervisor: Sigurd Skogestad, IKP
Co-supervisor: Christoph Backi, IKP

Norwegian University of Science and Technology
Department of Chemical Engineering

Abstract

For economical reasons, it is necessary to operate separators such that the the purity specification is maximized while having the levels and the system pressure controlled. In this thesis, a nonlinear model predictive control is applied to a three phase subsea gravity separator system.

The main modeling principle used to describe the oil droplets rising and settling of the droplets is given by Stokes law. Nonlinear dynamic equations were derived for the water level, total liquid level and for the pressure of the gas in the system based on the inflow and outflow dynamics, and were implemented in MATLAB as script files. The equations were nonlinear due to the geometry of the separator. These nonlinear models were formulated as a semi-explicit DAE system.

The NMPC optimization problem used in this report is solved using direct collocation method using CasADi software within the MATLAB programming environment. Simulation studies have been carried out for four cases, namely setpoint tracking, disturbance rejection, sensitivity to measurement noise and the optimal water level investigation. The performance of the closed loop Nonlinear Model Predictive Control (NMPC) has been studied based on the simulation results.

Simulation results were carried out to analyze the effect of measurement noise on the performance of the controller. Based on this analysis, the performance of the system for remaining cases was studied. The controller was analyzed for the setpoint tracking scenario where step change in water level was introduced to the system. Additionally, the performance of the controller to reject the disturbances was studied. Finally, the optimal level of water in the separator was investigated based on maximization of the total volume of oil entering its native phase.

Preface

This master thesis was written for the process systems engineering group at the Norwegian University of Science and Technology. This work was carried out during the Autumn of 2017 at the Department of Chemical Engineering.

I would like to express my deepest gratitude to my supervisor professor Sigurd Skogestad, for giving me an opportunity to work on this project and also for his support and useful discussions during my work.

I am also extremely thankful to my co-supervisor Dr. Christoph J. Backi for his constant guidance throughout the project, for being available to me even while being away on conferences, for providing the technical background on which this project is based and for reviewing the thesis. I would also like to thank Ph.D candidates Dinesh Krishnamoorthy, Tamal Das and Post Doctoral candidate Eka Suwartadi for their key inputs in understanding CasADi. I am grateful for the opportunities given by the process systems engineering group to participate in the meetings and discussions.

Finally, I would like to thank my family and friends without whom this journey would not have been possible. I am especially grateful to my parents who believed in me and wanted the best for me. I am also thankful to my sister, Megan for encouraging me throughout this experience.

Declaration of Compliance

I declare that this thesis is an independent work in agreement with the exam regulations of the Norwegian University of Science and Technology (NTNU).

Trondheim, Norway

June 12, 2017

Melissa Florine Dlima

Table of Contents

Abstract i

Preface ii

Table of Contents vii

List of Tables ix

List of Figures xiv

List of Symbols xv

Abbreviations xix

1 Introduction 1

2 Background 5

2.1 Particle movement through the fluid 5

2.2 Horizontal Gravity Separator Description 7

2.3 Introduction to Optimization 8

2.3.1 Mathematical Formulation 9
2.4 Optimal Control Problem 14
2.4.1 Methods to solve optimal control problem 15
2.4.2 Open loop vs. closed loop optimal control 23
2.5 Software Package 24
2.6 Plantwide Control by Timescale Separation 25

3 Modeling of the process 29
3.1 Assumptions 29
3.2 Mathematical Models Description 30
3.2.1 Inventory Models 30
3.2.2 Droplet size distribution 35
3.2.3 Horizontal Velocities 37
3.2.4 Vertical Velocities 38
3.2.5 Horizontal and vertical residence time 39
3.2.6 Separation Efficiency 41
3.3 Formulation of Models in the DAE form 42

4 Model Predictive Control Theory 47
4.1 Introduction 47
4.2 MPC Elements 48
4.3 Formulation of NMPC Optimization Problem 50
4.4 Direct Collocation Method to solve NMPC Optimization Problem 53
4.5 Finite Dimensional NLP 55
4.6 Continuous Logic in an NMPC problem 57

5 Simulation Results and Discussion 59

6 Conclusion 77

Bibliography **81**

Appendices **85**

A Simulation Parameters **87**

B Quadrature coefficients **89**

C Residence time of droplet classes **91**

D Rate of change constraints **95**

E MATLAB CODE **99**

E.1 velocity_inflows.m 99

E.2 gravity_ode.m 100

E.3 oildroplets_in_water.m 102

E.4 waterdroplets_in_oil.m 103

E.5 NMPC.m 105

List of Tables

3.1 Droplet size distribution. 36

5.1 Setpoint values for the controlled variables for different cases . . . 60

A.1 Simulation Parameters (Backi and Skogestad, 2017; Laleh et al., 2013). 87

List of Figures

2.1 Forces acting on a droplet settling through a continuous phase. . . 6

2.2 Schematic of the three phase horizontal gravity separator. 8

2.3 The figure on the left shows convex set and the figure to the right indicates nonconvex set 11

2.4 The figure on the left shows a convex function and the figure to the right indicates nonconvex function 11

2.5 Classification of methods to solve optimal control problem 16

2.6 Schematic of Direct Single Shooting. 18

2.7 Schematic of Direct Multiple Shooting. Note the discontinuities of states in the intermediate stage of optimization, where the equality constraints are not yet satisfied. 20

2.8 Schematic of Direct Multiple Shooting where the shooting gap constraints are satisfied after the convergence of the NLP solver (Kirches, 2010) 21

2.9 Schematic of the Direct Collocation method. The state derivative of the approximated polynomial (black line) and the value of the system dynamics (red dotted line). The polynomial approximation (orange curve) and the true solution of the system dynamics (blue dotted curve). The control input is piecewise constant over the interval $[t_k, t_{k+1}]$ 22
2.10 Schematic of open loop optimal control 23
2.11 Schematic of closed loop optimal control 24
2.12 Typical control hierarchy in a chemical plant (Larsson and Skogestad, 2000). 26

3.1 Schematic of the three phase horizontal gravity separator indicating heights of inventory. 31
3.2 Droplet size distribution for 10 droplet classes 36
3.3 Schematic of the three phase gravity separator indicating velocity components on the droplets. 37
3.4 Illustration of the horizontal and the vertical residence times. . . . 40
3.5 Plot of $\arctan$ function on the left in the range $[-\pi/2, \pi/2]$ converted to a function on the right that takes binary values. 44

4.1 Schematic of basic MPC structure 48
4.2 Receding horizon strategy for a total simulation time of t_f seconds 50
4.3 Illustration of the predictive control algorithm (Findeisen and Allgöwer, 2002) 51
4.4 Schematic representation of third order direct collocation in the single sampling interval $[t_k \;\; t_{k+1}]$. The control input is piecewise constant in the interval $[t_k \;\; t_{k+1}]$. 55
4.5 Control structure for a gravity separator 57

5.1 Right: Manipulated variables for low magnitude noise addition (black) and high magnitude noise addition (blue) Left: State variables for low magnitude noise addition (black) and high magnitude noise addition (blue) and the setpoint values (red) 62

5.2 Setpoint tracking of gravity separator controlled with NMPC. State variables h_W, h_L and p on the left with their respective manipulated variables $q_{W,out}$, $q_{O,out}$ and $q_{G,out}$ on the right 64

5.3 Pulse disturbance in liquid inflow $q_{L,in}$ (blue) and disturbance in gas inflow $q_{G,in}$ (black) 65

5.4 Disturbance rejection when there is sudden increase in inflow liquid $q_{L,in}$ and inflow gas $q_{G,in}$. Left: h_W, h_L and p, and the dotted red line indicates the reference values. Right: Manipulated variables $q_{W,out}$, $q_{O,out}$ and $q_{G,out}$ 66

5.5 Controlled variables h_L and p on top with their respective manipulated variables $q_{O,out}$ and $q_{G,out}$ on the bottom 69

5.6 Top: Controlled variable h_W indicated in (blue) and the bounds on h_W in dotted red. Below: Manipulated variable $q_{W,out}$ 70

5.7 Efficiencies of oil removal from the water phase for different water levels 71

5.8 Distribution of oil droplets at the water outlet for water level = 0.9 m 72

5.9 Distribution of oil droplets at the water outlet for water level = 1.65 m 72

5.10 Distribution of oil droplets at the water outlet for water level = 2 m 73

5.11 Distribution of oil droplets at the water outlet for water level = 2.2 m 73

5.12 Total volume of oil droplets entering their continuous phase V_{OoW} for the respective water levels. 75

C.1 Horizontal and vertical residence times of oil droplets in the water phase for water level = 0.9 m. The horizontal residence time = 90.45 s. 91
C.2 Horizontal and vertical residence times of oil droplets in the water phase for water level = 1.65 m. The horizontal residence time = 204.715 s. 92
C.3 Horizontal and vertical residence times of oil droplets in the water phase for water level = 2 m. The horizontal residence time = 259.58 s. 92
C.4 Horizontal and vertical residence times of oil droplets in the water phase for water level = 2.2 m. The horizontal residence time = 289.96 s. 93

D.1 The control inputs without the inclusion of rate of change constraints 95
D.2 The control inputs with the inclusion of rate of change constraints 96
D.3 The control inputs without the inclusion of rate of change constraints 97

List of Symbols

Latin Letters

Symbols	Description	Unit
A_L	Cross sectional area of liquid in the separator	m^2
A_O	Cross sectional area of water in the separator	m^2
A_p	Reference area of the droplet	m^2
A_W	Area of water in the separator	m^2
b	Time invariant parameters	_
C_d	Drag Coefficient	_
d	Disturbance vector	_
F_b	Buoyant force	N
F_d	Drag force	N
F_g	Gravitational force	N
g	Acceleration due to gravity	ms^{-2}
h_L	Liquid level	m
h_L^{SP}	Liquid level setpoint	m
h_O	Oil level	m
h_W	Water level	m
h_W^{SP}	Water level setpoint	m
l_p	Droplet position	m
N_{Re}	Reynolds number	_
n	Initial number of droplets	_

n_G	Amount of gas	$moles$
p	System pressure	bar
p^{SP}	System pressure set point	bar
$q_{G,in}$	Gas inflow	m^3s^{-1}
$q_{G,out}$	Gas outflow	m^3s^{-1}
$q_{L,in}$	Liquid inflow	m^3s^{-1}
$q_{L,out}$	Liquid outflow	m^3s^{-1}
$q_{W,in}$	Water inflow	m^3s^{-1}
$q_{W,out}$	Water outflow	m^3s^{-1}
R	Universal gas constant	$Jmol^{-1}K^{-1}$
r	Radius	m
T	Temperature	K
T_c	Control horizon	s
T_p	Prediction horizon	s
u	Vector of manipulated variables	_
u_{opt}	Optimal control sequence	_
V_G	Volume of gas in the separator	m^3
V_L	Volume of liquid in the separator	m^3
V_{OoW}	Total oil volume leaving water phase	m^3
V_{Sep}	Volume of the separator	m^3
V_{WoO}	Total water volume leaving oil phase	m^3
v_h	Horizontal velocity	ms^{-1}
v_v	Vertical velocity	ms^{-1}
x	Vector of state variables	_

Greek Letters

Symbols	Description	Unit
α	Water cut	–
β	Oil cut	–
η_O	Efficiency of oil	–
η_W	Efficiency of water	–
μ_O	Viscosity of oil	$kg(ms)^{-1}$
μ_W	Viscosity of water	$kg(ms)^{-1}$
ϕ_{oo}	Fraction of oil entering the oil phase	–
ϕ_{ow}	Fraction of oil entering the water ohase	–
ϕ_{wo}	Fraction of water entering the oil phase	–
ϕ_{ww}	Fraction of water entering the water phase	–
ρ_G	Density of gas	kgm^{-3}
ρ_O	Density of oil	kgm^{-3}
ρ_W	Density of water	kgm^{-3}

Abbreviations

CasADi	Computer algebra system for Automatic Differentiation
CV	Controlled Variable
DAE	Differential Algebraic Equations
DMS	Direct Multiple Shooting
DOF	Degree of Freedom
DSS	Direct Single Shooting
HJB	Hamilton- Jacobi- Bellman
IPOPT	Interior Point OPTimizer
KKT	Karush-Kuhn-Tucker conditions
LP	Linear Programming
LQG	Linear Quadratic Gaussian
LTI	Linear Time Invariant
MINLP	Mixed Integer Nonlinear Programming
MPC	Model Predictive Control
NLP	Nonlinear Programming
NMPC	Nonlinear Model Predictive Control
OCP	Optimal Control Problem
ODE	Ordinary Differential Equation
PID	Proportional Integral Derivative
PMP	Pontryagins Maximum principle
QP	Quadratic Programming
RHC	Receding Horizon Control
SQP	Sequential Quadratic Programming

Chapter 1

Introduction

The recent interest in subsea processing is mainly driven by the advantages of reduced pumping cost, water management cost and reduced environmental impact due to water re-injection (Moraes et al., 2013). Also, the subsea processing makes it possible to use smaller or no platforms at all, due to the fact that the equipment used for processing is placed on the seabed rather than topside (Albuquerque et al., 2013).

The majority of oil wells produce a mixture of mainly gas, oil and water. Conventionally, to separate this mixture, a series of separators is used placed starting with a rough separation of gas-liquid mixture in three phase gravity separators to separate the mixtures into oil, water and the gas phase. The produced water that has been separated can be used for re-injection into the reservoir, which helps to increase the pressure and ultimately the recovery. With this advantage, many subsea separators have been implemented around the world. In addition to the increased production, the companies need to comply with strict requirements concerning the disposal of produced waste water. There are many regulations for the maximum allowable concentration of oil in the water that could be discharged into the sea which are set in terms of daily maximum and monthly average (Prescott, 2012). For example, the maximum limit for hydrocarbon discharge in North Sea is set at 30ppm (Durdevic et al., 2017).

Three phase horizontal gravity separators are considered in this thesis. The separation is based on density difference between the single phases as well as gravitational forces. A proper control scheme allows to separate the mixtures based on maximization of oil production, minimization of oil in the produced water or to control the levels at their desired setpoints.

For the last two decades, predictive control has been found to be a very good controller design scheme mainly due to the high performance controllers that can be used for multivariable proceses and also that the constraints in the process are handled in a straightforward and organized way (Nunes, 2001).

According to Stokke et al. (1994), a study was carried out on Statoil's Statfjord A crude oil production platform, where Model Predictive Control (MPC) was applied to control the water, oil and the system pressure. In this process, due to the higher production levels and large variations in the oil levels, experienced operators had to adjust several PI controllers simultaneously. When the PI controllers are implemented in a system, it is inappropriate to manipulate more than one PI controller at a time. Therefore, in order to solve this problem, a Model Predictive Control layer is placed on top of the PI control loops, so that it is possible to optimally and dynamically control the setpoints of the PI controllers.

There has not been much work done on Model Predictive Control applied to the gravity separator system. Nunes (2001) used a polynomial operator technique to study the stability and performance of predictive control applied to a gravity separator system. Here, the nonlinear system was linearized around a steady state and proves that the linear model predictive control is a good control method for oil -water-gas separators. Stokke et al. (1994) studied both MPC and LQG (Linear Quadratic Gaussian) to control the levels in an oil-water-gas separator train. It was found that MPC was far superior to LQG controller since the LQG controller was sensitive to noise.

The remainder of this thesis is structured as follows. Chapter 2 outlines majorly the description of the process, introduction to optimization and optimal control, methods to solve optimal control problems and the control structure hierarchy in chemical plants. Chapter 3 presents the control oriented modeling approach. Chapter 4 outlines the theory of Model Predictive Control (MPC). Chapter 5 describes the simulation results and chapter 6 concludes the thesis with an outlook on future work.

Chapter 2

Background

2.1 Particle movement through the fluid

To understand the separation process, it is important to understand the principle of emulsions. An emulsion is a dispersion of one liquid in another liquid, where both liquids are immiscible. The phase in which the droplets are present is the continuous phase and the droplets are termed as the dispersed phase. The main principle of gravity separation is the separation of droplets based on density differences and gravitational forces. The forces exerted on the droplets are the drag force, buoyant force and the gravitational forces as shown in Figure 2.1.

The drag force also known as the frictional force will be exerted on the droplet in the viscous fluid. This force is always in the direction opposite to the velocity of the droplet and is given by

$$F_d = \frac{C_D\, v^2\, \rho\, A_p}{2}, \tag{2.1}$$

where C_D is the drag coefficient, v is the relative velocity of the droplet with

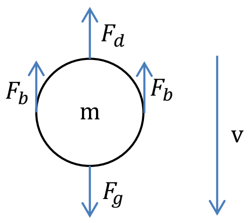


Figure 2.1: Forces acting on a droplet settling through a continuous phase.

respect to the surrounding fluid, ρ is the density of the surrounding fluid and A_p is the reference area of the droplet.

The droplet experiences an upward force which opposes the weight of the particle and is termed the buoyant force. It comes from the pressure exerted on the droplet by the fluid and is given by

$$F_b = V_p \, \rho \, g, \tag{2.2}$$

where V_p is the volume of the droplet and g is the acceleration due to gravity.

In addition to the drag and the buoyant force, the particle is exposed to the gravitational force given by

$$F_g = V_p \, \rho_p \, g, \tag{2.3}$$

where ρ_p is the density of the droplet.

The drag coefficient C_D depends on the Reynolds number and it can be defined for several flow regimes. For laminar flow, Stokes' law is applied and the drag coefficient is given by

$$C_D = \frac{24}{N_{Re}}, \tag{2.4}$$

where the Reynolds number N_{Re} is given by

$$N_{Re} = \frac{d_p\, v\, \rho}{\mu}, \tag{2.5}$$

where d_p is the diameter of the droplet and μ is the viscosity of the medium in which the droplet is present.

At equilibrium, gravitational force on the particle is equal to the drag force and the buoyant force. Examining equations (2.1), (2.2), (2.3) for the forces acting on the particle, yields the terminal velocity of the particle in the laminar flow, assuming Stokes' law valid, and is given by

$$v = g d_p^2 \frac{(\rho_p - \rho)}{18\mu}. \tag{2.6}$$

2.2 Horizontal Gravity Separator Description

In this thesis, a horizontal gravity separator is considered for bulk separation of gas, oil and water. Many different separator types are used in industry, differing in shape and internal devices. Thereby, the three phase horizontal gravity separator as presented in Figure 2.2 is one of the most common types being employed (Mendes et al., 2012).

Figure 2.2 shows volumetric liquid and the gas inflows $q_{L,in}$ and $q_{G,in}$ respectively. The liquid which is a mixture of oil and water enters the separation chamber and separates out into the continuous water and the oil layers based on the difference in density and the gravitational forces as introduced in section 2.1. The oil flows over a weir plate to the oil chamber where the valve u_O is manipulated to extract the oil from the separator vessel. Also, the valves u_W and u_G are used to control the outflows of water and gas from the separator. The control of the levels h_W, h_L and the system pressure p is done using the valves u_W, u_O and u_G respectively.

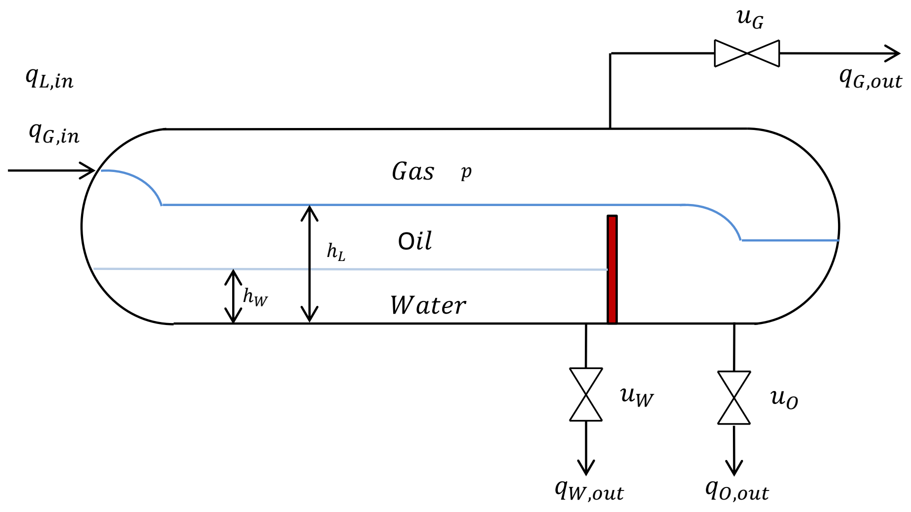


Figure 2.2: Schematic of the three phase horizontal gravity separator.

2.3 Introduction to Optimization

Mathematical optimization is an important tool for the analysis of physical system. This tool has been extensively used for the selection of best elements from a set of available elements. To analyze the system of study, an objective must be identified which could be the maximization of a company's profits or minimization of an energy consumption in a chemical plant. In either of the cases, the objective is largely determined by certain characteristics of the system namely the variables or unknowns. The aim of optimization is to find those unknown values or variables that optimize the objective. Usually, the variables are constrained or restricted to some bounds. The process of deciding the objective of the given problem, along with variables and bounds on the variables is termed modeling of the given system.

After the formulation of the model, an appropriate optimization algorithm has to be chosen to solve the problem of interest.

2.3.1 Mathematical Formulation

Optimization is the maximization or minimization of an objective function subject to variables which are constrained by bounds. Optimization models represent the problem choices as decision variables and attempt to find the variables that maximize or minimize the objective function value subject to constraints on variables. The optimization problem can be formulated as follows:

$$\min_{x} f(x) \tag{2.7a}$$

$$\text{subject to} \quad g_i(x) = 0 \qquad i = 1, ..., m \tag{2.7b}$$

$$h_j(x) \leq 0 \qquad j = 1, ..., p \tag{2.7c}$$

$$x_{min} \leq x \leq x_{max} \tag{2.7d}$$

where $f(x)$ is the objective function, $g(x)$ are the set of equality constraints and $h(x)$ are the set of inequality constraints, x are the decision variables constrained by lower and upper bounds x_{min} and x_{max} respectively.

If the goal is to minimize the objective function, then the minimization problem can be written as

$$\min_{x} f(x) \Leftrightarrow \max_{x} -f(x)$$

Now the minimization problem given by (2.7) can be formulated as the maximization problem

$$\max_{x} - f(x) \tag{2.8a}$$

$$\text{subject to} \quad g_i(x) = 0 \qquad i = 1, ..., m \tag{2.8b}$$

$$h_i(x) \leq 0 \qquad i = 1, ..., p \tag{2.8c}$$

$$x_{min} \leq x \leq x_{max} \tag{2.8d}$$

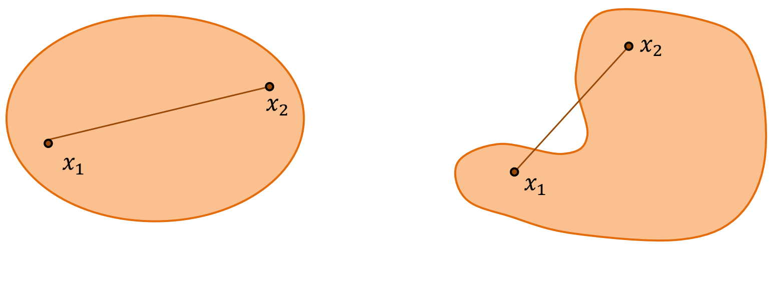
Optimization problems can be of many classes comprehensively presented in the book by Nocedal and Wright (2006). In this thesis, a brief introduction to a few optimization problem types will be discussed.

Optimization problems formulated in the general form (2.8) can be classified based on the linearity, nonlinearity, convexity of the objective function and constraints. It also depends on how large or small the number of variables in the optimization problem.

Constrained and unconstrained optimization: In case of unconstrained optimization problems, the general form (2.8) does not have any equality or inequality constraints, meaning Equations (2.8b) - (2.8d) are absent. These problems consider maximizing or minimizing the objective function value with no restriction on the decision variable values. However, most of the problems in engineering applications are constrained.

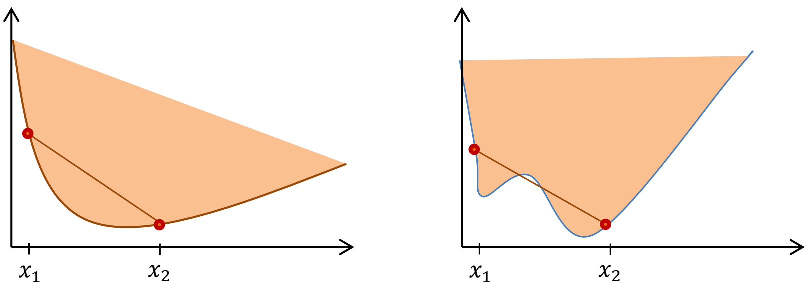
Local and global optimization: Many optimization problems attempt to find a local solution, which is a point at which the objective function value is smaller than the neighbouring points. A Global solution is the point at which the objective function value is smaller than all feasible points.

Convex and non-convex optimization: The optimization problem is said to be convex if the objective function is a convex function, all the equality constraints are linear and inequality constraints are concave. A set is defined as a convex set if a line segment joining any two points in that set lies within that set as shown in Figure 2.3a. The function which is defined in an interval is said to be convex if the straight line segment joining any two points in that interval is as shown in Figure 2.4a.



(a) A convex set **(b)** A nonconvex set

Figure 2.3: The figure on the left shows convex set and the figure to the right indicates nonconvex set



(a) A convex function **(b)** A nonconvex function

Figure 2.4: The figure on the left shows a convex function and the figure to the right indicates nonconvex function

In addition to the above optimization problem types, some specific optimization problems will be discussed for the completeness of the report.

Linear Programming (LP): If the objective function f(x) defined by (2.8) is linear and all the constraints are linear, then the formulation is said to be a linear programming problem. LP problems are always convex since the objective function is linear and the constraints are linear with the feasible set being convex. The LP problem is defined by

$$\min_{x \in R^n} c^T x \tag{2.9a}$$

$$\text{subject to} \quad g_i(x) = a_i^T x - b_i = 0, \qquad i = 1, ..., m \tag{2.9b}$$

$$h_j(x) = a_j^T x - b_j \leq 0 \qquad j = 1, ..., p \tag{2.9c}$$

Quadratic Programming (QP): The mathematical optimization problem is said to be quadratic programming if the objective function f(x) is quadratic and the constraints are linear. The convexity of a QP problem depends on the positive semi-definiteness of the matrix Q in (2.10a). The QP problem is defined by

$$\min_{x \in R^n} x^T Q x + c^T x \tag{2.10a}$$

$$\text{subject to} \quad g_i(x) = a_i^T x - b_i = 0, \qquad i = 1, ..., m \tag{2.10b}$$

$$h_j(x) = a_j^T x - b_j \leq 0 \qquad j = 1, ..., p \tag{2.10c}$$

Nonlinear Programming (NLP): A general optimization problem is to select n decision variables $x_1, x_2, x_3, .., x_n$ from a feasible region such that a given objective function is minimized or maximized. The problem becomes a nonlinear programming problem if any of the objective or constraints are defined by nonlinear functions of variables. A general nonlinear minimization problem is given by

$$\min_{x,u} f(x) \tag{2.11a}$$

$$\text{subject to} \quad f_i(x) \leq 0, \qquad i = 1, 2, ..., m \tag{2.11b}$$

$$h_j(x) = 0, \qquad j = 1, 2, ..., p \tag{2.11c}$$

The most promising approaches to solve nonlinear optimization problems are active-set SQP (Sequential quadratic programming) methods and the interior-point methods.

SQP is one of the most widely used methods to solve nonlinear programming problems. It is an iterative procedure, where at every major iteration, a QP sub problem is solved. The solution of the QP problem is used to construct a new iterate. The construction of new iterates is done such that the series of iterates converge to a local minimum.

Interior point methods solve the problem (2.11) by applying Newtons method to a series of modified KKT (Karush–Kuhn–Tucker) conditions. In the hierarchy of convex optimization algorithms, interior point methods form the third level. The simplest or the first level includes problems whose KKT conditions are linear equations that can be solved analytically. Next level is Newtons method, which reduces the problem into a sequence of equality constrained quadratic problems. Interior point methods solve the optimization problem by reducing it to a sequence of linear equality constrained problems. Barrier method and primal-dual interior point method are frequently used interior point techniques. The goal of interior point algorithms is to approximately formulate the inequality constrained problem as an equality constrained problem where Newtons method can be used.

There are several software packages that implement nonlinear interior-point methods such as LOQO, KNITRO/DIRECT, IPOPT, BARNLP, whereas active set packages are MINOS, SNOPT, KNITRO/ACTIVE etc (Nocedal and Wright, 2006).

2.4 Optimal Control Problem

An optimal control problem is a constrained optimization problem with a dynamic system as one of its constraint. There are three parts in an optimal control problem formulation namely an objective function, constraint functions and a process model. The general optimal control problem for a class of dynamic optimization problems can be formulated as

$$\min_{x,u} J = E(x(t_f), b, t_f) + \int_{t_0}^{t_f} L(x(t), u(t), b, t)dt \tag{2.12a}$$

$$\text{subject to} \quad \dot{x} - f(x(t), u(t), b, t) = 0, \tag{2.12b}$$

$$g(x, u, b, t) = 0, \tag{2.12c}$$

$$x_0 = x(t_0), \tag{2.12d}$$

$$h(x, u, b, t) \leq 0 \qquad \forall t \in [t_0, t_f] \tag{2.12e}$$

The objective function also known as performance index represents the mathematical expression that has to be either maximized or minimized. The constraint function and the plant models both determine a search domain for the optimization procedure.

The objective function given in (2.12a) is in Bolza form where $L(\cdot)$ is known as the Lagrange form and $E(\cdot)$ is known as the Mayer form. Here, t represents the time variable, $x(t) \in \mathbb{R}^{n_x}$ and $u(t) \in \mathbb{R}^{n_u}$ denote a vector of state variables and control decision variables respectively which are going to be optimized. Parameters are the time independent decision variables denoted by $b \in \mathbb{R}^{n_b}$.

The process model given by (2.12b) and (2.12c) is also known as system equation constraints which represent an additional set of equality constraints and gives a mathematical representation of phenomena taking place in the observed system. (2.12b), (2.12c) represent ODE models and algebraic equations respectively. Both set of equations combined present a set of differential algebraic equations (DAEs).

Different equality and inequality constraints may bound the values of control and state variables to safeguard the environment, to maintain certain setpoints in the control loop etc. Equation (2.12e) is termed path constraint, which is used to determine a particular value of the state or decision variables at specific time instants. They are also known as the terminal constraints, when for instance state at the final time t_f is restricted to a particular value.

2.4.1 Methods to solve optimal control problem

There are three fundamental approaches to solve a constrained optimal control problem as presented in the form (2.12a). These methods are outlined as follows.
1) Dynamic Programming, Hamilton- Jacobi- Bellman (HJB)
2) Indirect methods, Calculus of variations, Euler Lagrange differential equation, Pontryagin's Maximum principle (PMP)
3) Direct methods which are based on parameterizing optimial control problem into a finite dimensional nonlinear programming problem (Kaya, 2007; Geiger, 2009).

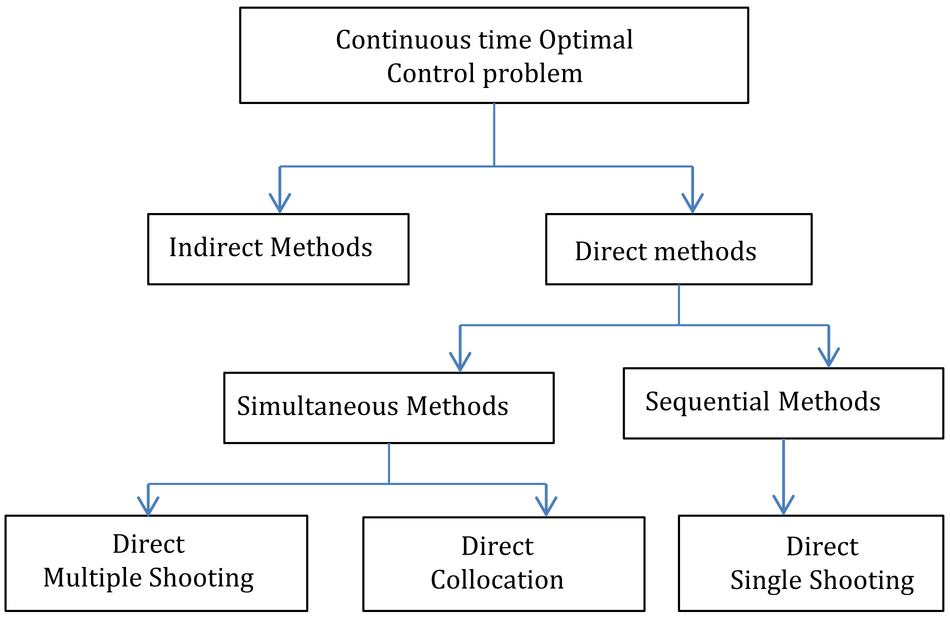


Figure 2.5: Classification of methods to solve optimal control problem

Indirect Method

Indirect methods make use of optimal conditions of the infinite optimal control problem to obtain a boundary value problem. These optimality conditions are discretized and then the boundary value problem is solved. Hence they are termed ”optimize and then discretize” (Diehl et al., 2006).

Direct Method

Contrary to the indirect methods, direct methods can be used to solve the optimal control problems without the need to derive necessary conditions for optimality. The principle of direct method is to discretize the original infinite dimensional optimal control problem into a nonlinear program (NLP) with finite dimensions. Therefore these methods are also termed ”discretize and then optimize” methods. The finite dimensional NLP is solved numerically using standard solution algorithms. The two well known approaches within this group of methods are sequential and simultaneous approaches. Direct single shooting is a class of sequential methods while direct multiple shooting and direct collocation represents the simultaneous methods.

The following methods give an overview of the direct methods to solve the infinite dimensional optimal control problems. Direct methods can be used both for ODE and DAE models. For the sake of simplicity in understanding, the optimal control problem subject to an ODE model is briefly explained. However, this thesis is based on a system of semi-explicit differential algebraic equations (DAE) as will be seen in section 3.3, and the direct method used to solve this is described in section 4.4.

Direct Single Shooting (DSS)

This is the basic direct method for solving optimal control problems. Here the model simulation and optimization are done sequentially. It is a control parametrization method where the control is approximated and the control trajectory is parametrized using a piecewise smooth approximation, typically piecewise linear functions and the ODE or the DAE constraints (2.12b) are solved using standard numerical integration solvers. The parametrization of the control trajectory is done in the time horizon $[t_0, t_f]$, where the horizon is divided into set of gridpoints as $t_0 = 0 < t_1 < .... < t_N = t_f$.

$$u(t) = q_i \qquad \text{for } t \in [t_i, t_{i+1}], \qquad i = 0, 1, ..., N-1 \tag{2.13}$$

In general, the control parametrization is represented as $u(t; q)$, where the term after the semicolon indicate the dependence of the discretized control parameters on every subinterval. Using numerical integration, the states are obtained as a function of several control parameters and they are denoted as $x(t; q)$. Finally, an NLP solver is used to obtain the optimal control sequence. This is called the sequential approach for solving the optimal control problem since the ODE/DAE solvers integrate the differential equations and the NLP solver finds optimal control parameters sequentially.

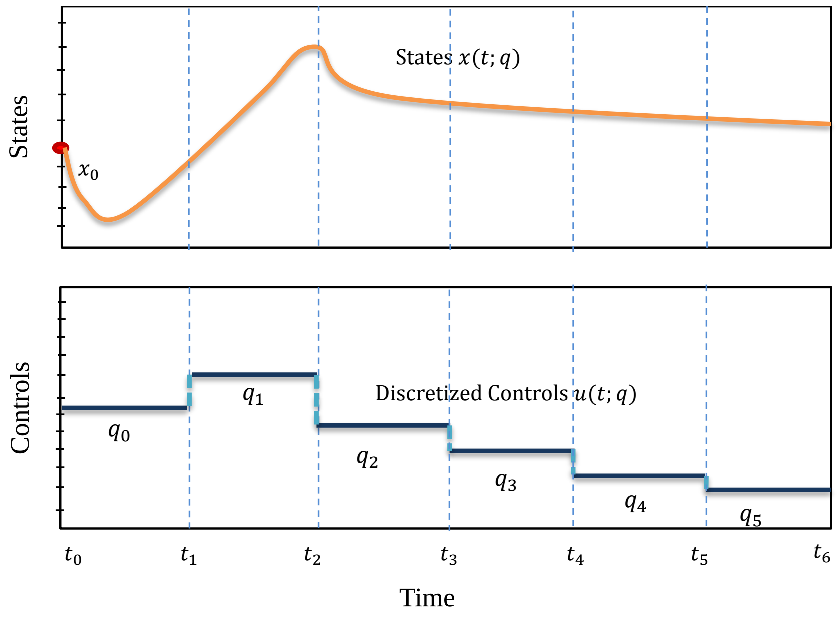


Figure 2.6: Schematic of Direct Single Shooting.

Direct Multiple Shooting (DMS)

In a direct multiple shooting method, the time interval $[t_0, t_f]$ is divided into several subintervals as presented earlier and a direct shooting method is used over each subinterval. On every subinterval the control trajectory is piecewise discretized as in (2.13) and the ODE (2.14a) is solved numerically on each interval independently, starting with an initial value (2.14b) at every gridpoint to obtain multiple trajectory segments as shown in Figure 2.7.

$$\dot{x}_i(t) - f(x_i(t), q_i) = 0, \tag{2.14a}$$

$$x_i(t_i) = s_i \tag{2.14b}$$

Solving the above set of initial value problems (2.14) at every gridpoint, trajectory segments equal to the number of subintervals are obtained and they are denoted as $x_i(t; s_i, q_i)$.

The set of control parametrization coefficients and the initial values of state beginning at each subinterval as denoted by v remain unknown in the optimization problem and can be solved using NLP methods like Sequential Quadratic Programming (SQP) methods or Interior Point methods as described in section 2.3.1.

$$v = [s_0, q_0, s_1, q_1, ..., s_5, q_5, s_6]$$

In order to enforce continuity of the state trajectories, additional constraints are added at the interface of each subinterval where it is desired to drive the difference between $x(t_i^-)$ and $x(t_i^+)$ to zero.

Numerical integration of the IVP over the time interval $[t_i, t_{i+1}]$ provides the state x at t_{i+1} denoted by $x_i(t_{i+1}; s_i, q_i)$ as shown in Figure 2.7. The trajectory is physically meaningful when the shooting gaps at every subintervals are closed as shown in Figure 2.7 meaning

$$x_i(t_{i+1}; s_i, q_i) - s_{i+1} = 0$$

The variables s_i and q_i indicate that the trajectory segments depend on the initial

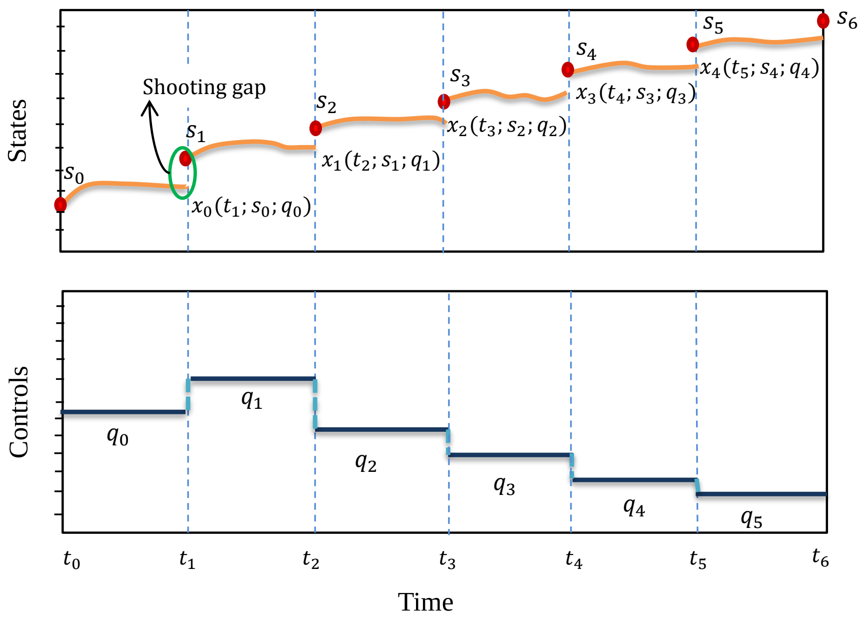


Figure 2.7: Schematic of Direct Multiple Shooting. Note the discontinuities of states in the intermediate stage of optimization, where the equality constraints are not yet satisfied.

values at the intervals and the corresponding control inputs. The shooting gaps are introduced as additional constraints in the optimization problem.

After the convergence of the NLP solver, the shooting gap constraints are fulfilled as indicated in Figure 2.8.

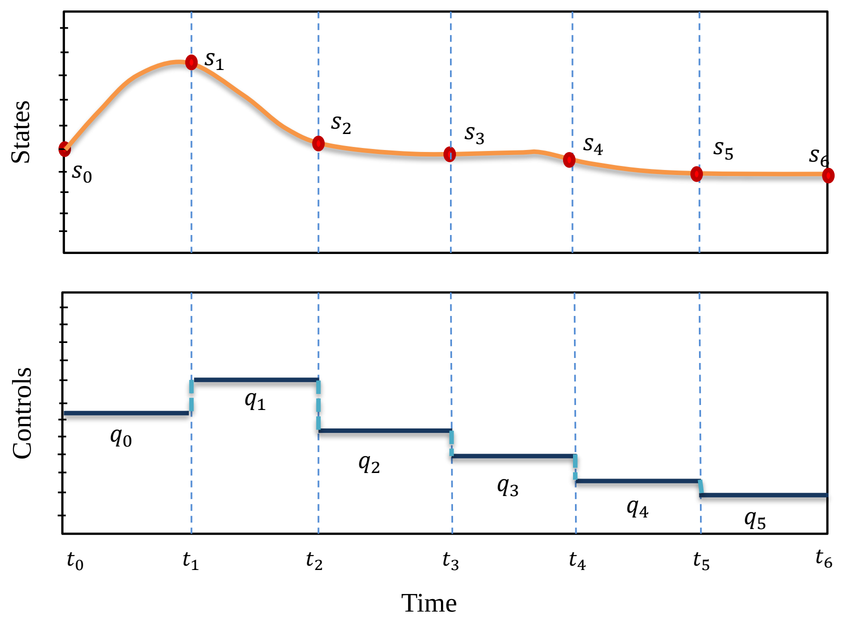


Figure 2.8: Schematic of Direct Multiple Shooting where the shooting gap constraints are satisfied after the convergence of the NLP solver (Kirches, 2010)

Direct Collocation

A direct collocation method is a state and control parametrization method where the states are approximated by cubic polynomials and the controls are expressed as piecewise linear function. The discretization is performed on a grid of N intervals with c intermediate collocation points in each interval. For example, in interval $[t_0, t_1]$ three intermediate collocation points are indicated as shown in Figure 2.9. There are different schemes to choose the collocation points depending on factors like stability and order of convergence (Magnusson and Åkesson, 2015).

$$\dot{x}(t) = f(x(t), u(t)), \tag{2.15a}$$

$$x(t_0) = x_0 \tag{2.15b}$$

An infinite ODE system is approximated using a collocation scheme, for instance the Lagrange collocation method, to obtain a polynomial approximation on each interval.

At every collocation point, the state derivative of the approximated Lagrange polynomial is evaluated and is denoted as $\dot{x_c}$. The value of the system dynamics (2.15a) is also computed at the collocation points $f(x_c, u(t))$. The main idea is to minimize the error between the state derivative from the system dynamics and the derivative of polynomial approximation at the collocation points as shown in Figure 2.9.

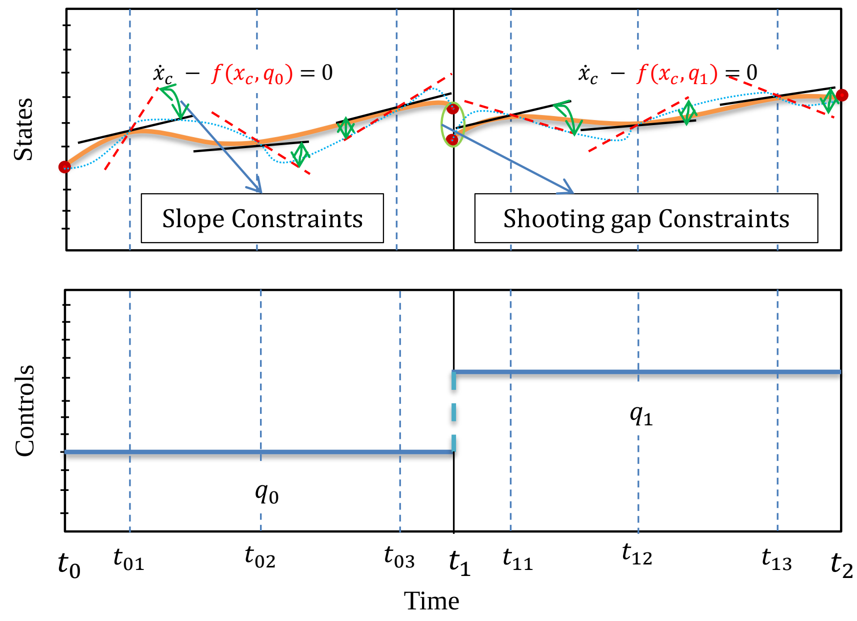


Figure 2.9: Schematic of the Direct Collocation method. The state derivative of the approximated polynomial (black line) and the value of the system dynamics (red dotted line). The polynomial approximation (orange curve) and the true solution of the system dynamics (blue dotted curve). The control input is piecewise constant over the interval $[t_k, t_{k+1}]$

The equality constraints $\dot{x_c} - f(x_c, u(t)) = 0$ are introduced into the optimization algorithm as slope constraints. In addition to this, shooting gap constraints as discussed before for the multiple shooting method are also introduced into the optimization problem. Figure 2.9 illustrates the concept of the direct collocation method.

However, the optimal control presented above are open loop techniques which do not consider measurement noise or errors in the model. Figure 2.10 illustrates open loop control. To take the disturbances affecting the process into account, there has to be a closed loop system as shown in Figure 2.11, where the estimated states or the actual measurements are fed back to the optimizer. This is known as Model Predictive Control (MPC) or receding horizon control and is described in detail in Section 4.1.

2.4.2 Open loop vs. closed loop optimal control

In open loop optimal control, an optimal control law which is of the form $u^{opt}(t) = \omega(t; t_0, x_0)$ obtained for a specific initial value of the state $x(t_0) = x_0$, for $t \in [t_0, t_f]$.

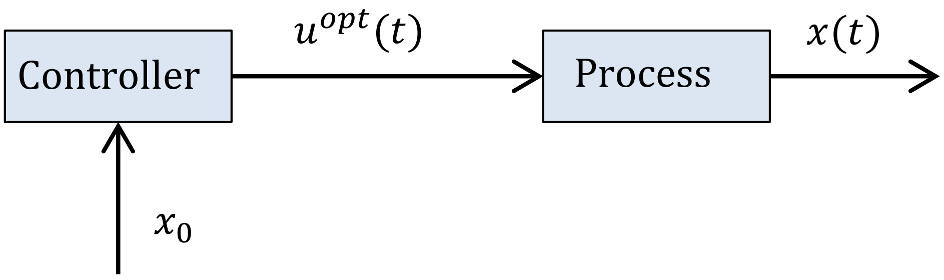


Figure 2.10: Schematic of open loop optimal control

In case of closed loop optimal control, a feedback control law is of the form $u^{opt}(t) = \omega(t; x(t))$, wherein the optimal control actions $u^{opt}(t)$ are calculated corresponding to the current state $x(t)$, for $t \in [t_0, t_f]$.

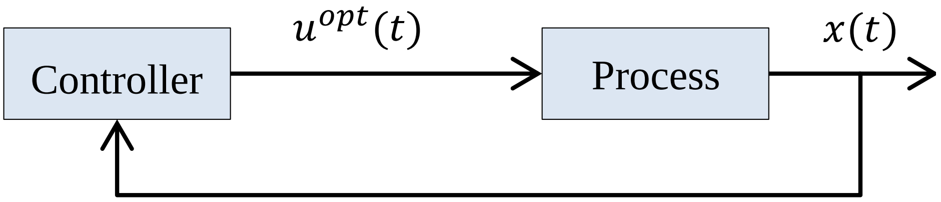


Figure 2.11: Schematic of closed loop optimal control

2.5 Software Package

In this thesis, the optimization software package CasADi (**C**omputer **a**lgebra **s**ystem for **A**utomatic **D**ifferentiation) is used to solve Optimal Control Problems (OCP) and nonlinear model predictive control (NMPC) for a gravity separator system. CasADi solves the OCPs using direct and indirect methods both for ODE and DAE systems. Details on the installation and the syntax of CasADi can be found in the user guide by Andersson (2013).

CasADi is a tool used to efficiently compute the derivatives using automatic differentiation and is specifically developed for dynamic optimization problems. It makes use of two different representations for symbolic expressions. The two most fundamental classes are the SX and the MX symbolics. SX is the scalar representation which involves the unary or binary elementary operations. MX is the matrix representation where the elementary operations are multiple-input and multiple-output. The MX symbolic representation is more efficient for high level operations like matrix multiplication and function calls. On the contrary, the SX representation provides faster computations by reducing the additional symbolical simplifications (Magnusson and Åkesson, 2015). The mixing of these two representations makes it possible to implement a code that is both computationally fast and generic since the SX expressions have much lower overhead per operation. Thus the SX expressions are considered for use in low level operations, for example while declaring the variables, while the MX expressions should be used for constraint function of an NLP. However, it is important to note that performing any other operation like multiplication of the two symbolic representations for instance cannot be done.

2.6 Plantwide Control by Timescale Separation

The term ”plantwide control” is used in the area of process control and it is defined as the selection of the control structure (Larsson and Skogestad, 2000). It is very important to have the right control selection since the chemical or the process plants are largely complex. As seen in Figure 2.12, the layers constituting scheduling, site-wide optimization and local optimization deal with economic optimization whereas the layers supervisory control and regulatory control are concerned with setpoint tracking defined by the layer above. The two main objectives for control is the longer term economics and short term stability. Based on these two objectives, the control layer is divided into a *”slow” supervisory (economic) layer* and a *”fast” regulatory (stabilization) layer*.

The objectives of the supervisory layer is to perform advanced economic control actions (meaning to control CV_1 at their desired setpoints and to provide setpoints CV_2^{SP} to the regulatory layer) and avoid saturation of the regulatory layer. The implementation of the supervisory control is done either by using model predictive

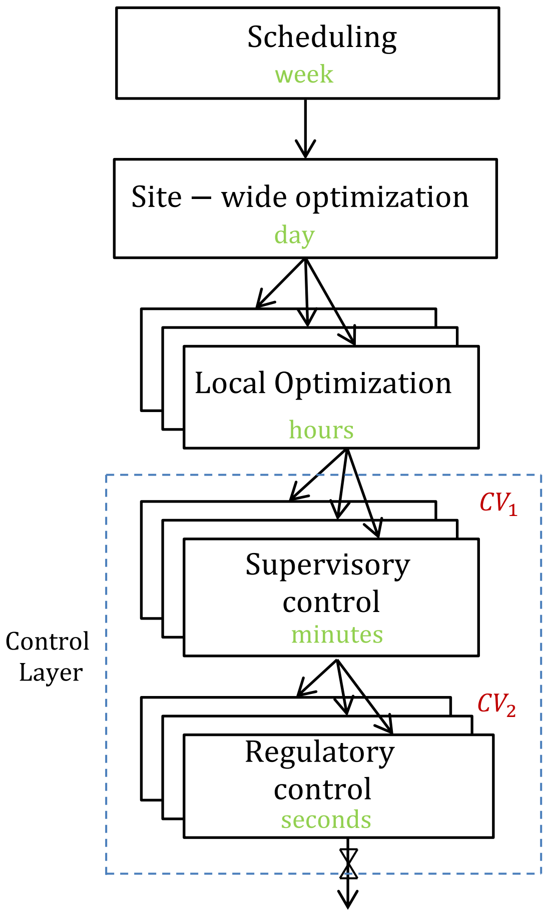


Figure 2.12: Typical control hierarchy in a chemical plant (Larsson and Skogestad, 2000).

control (MPC) or by making use of advanced control based on simpler elements like cascade control, selectors, ratio control, split range control, decoupler and valve position control. The advantage of having these simple elements is that the implementation can take place in existing basic control structure. However, it is not easily understood by the engineer who has not designed it. Hence, model based solutions (MPC) are usually preferred since the design can be easily modified.

The main objective of the regulatory layer is to keep the plant or the process at the steady state, by using simple single loop PID controllers. This layer tracks the setpoints given by the supervisory layer (MPC).

Chapter 3

Modeling of the process

The models presented in this chapter are based upon the work of Backi and Skogestad (2017).

3.1 Assumptions

- Absence of dense - packed layer in the separator which means that the droplets will either rise or fall from the interface to disperse in the continuous phases.
- Plug flow model for both the oil and the water phase.
- Absence of coalescence and breakage of the droplets.
- *Stokes' law* is applied for the rising and the settling velocity of the water and oil droplets.
- There are neither water nor oil droplets dispersed in the gas phase and no gas droplets dispersed in the water and oil phases respectively.

3.2 Mathematical Models Description

This section describes the models for the horizontal gravity separator. The simplified schematic of the gravity separator is illustrated in Figure 3.1. The liquid enters the separator with volumetric flow rate $q_{L,in}$ and the gas flow rate into the separator is $q_{G,in}$. The separator is divided into two sections, namely the separation section and an oil chamber. The incoming liquid is a mixture of water and oil and the volume fraction of water in the liquid is termed water cut α, whereas the volume fraction of oil is termed oil cut $\beta = 1 - \alpha$. As the liquid enters the separator, some of the water droplets disperse into the bulk oil phase which is denoted by ϕ_{wo}. It follows that out of inflowing water, the fraction of water entering the water continuous phase is $\phi_{ww} = 1 - \phi_{wo}$. Similarly, some of the oil droplets will disperse into the bulk water phase which is denoted by ϕ_{ow}. Therefore, the fraction of the inflowing oil entering the continuous oil phase is $\phi_{oo} = 1 - \phi_{ow}$. The oil rich phase, flows over the weir into the oil chamber and the flow is controlled by the valve u_O. Similarly, the water rich phase is released before the weir and the flow is manipulated using valve u_W.

The objective is to control the levels of water, oil and the system pressure and to calculate the amount of dispersed phase in the bulk phase outlets. The variables in the model are classified as state variables $x = [h_L \; h_W \; p]$, manipulated variables $u = [q_{W,out} \; q_{O,out} \; q_{G,out}]$ and the disturbance variable $d = [q_{L,in} \; q_{G,in}]$.

3.2.1 Inventory Models

In this section, non-linear dynamic models are derived representing the change in water level, total liquid level, which includes the oil and the water level, as well as the system pressure. Figure 3.1 illustrates the simplified schematic of the separation process indicating the water level, the total liquid level, the inlet and the outlet sections.

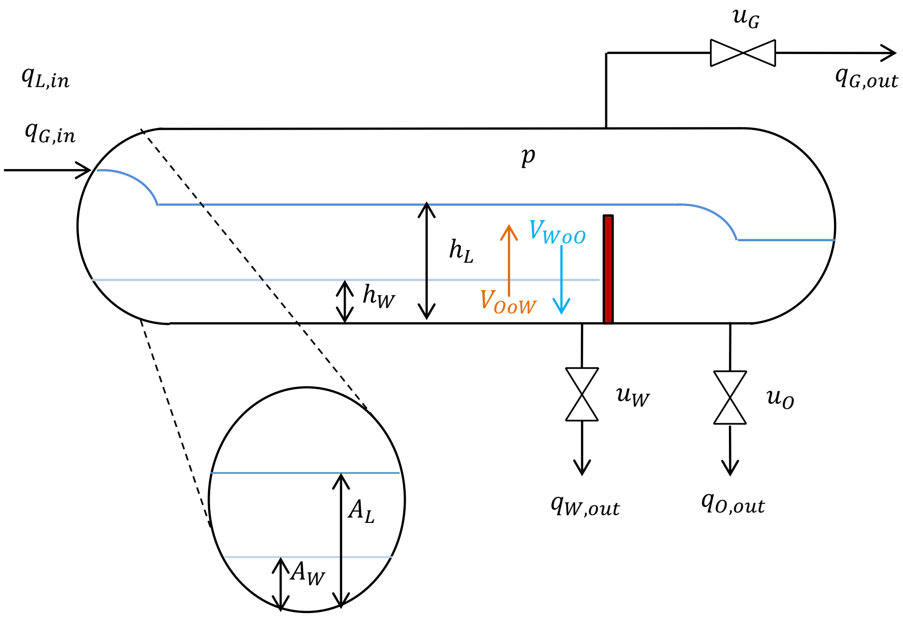


Figure 3.1: Schematic of the three phase horizontal gravity separator indicating heights of inventory.

Rate of change of water level

The rate of change of water volume is given as

$$\frac{dV_W}{dt} = q_{W,in} - q_{W,out} - \frac{V_{OoW}}{t_h^{water}} + \frac{V_{WoO}}{t_h^{oil}}. \tag{3.1}$$

The inflowing water contains dispersed oil droplets, and hence $q_{W,in}$ is given by

$$q_{W,in} = q_{L,in}(\alpha\phi_{ww} + \beta\phi_{ow}).$$

Hence, the rate of change of water volume is

$$\frac{dV_W}{dt} = q_{L,in}(\alpha\phi_{ww} + \beta\phi_{ow}) - q_{W,out} - \frac{V_{OoW}}{t_h^{water}} + \frac{V_{WoO}}{t_h^{oil}}.$$

where V_{OoW} denotes the total volume of oil droplets leaving the water phase into their continuous phase (subscript OoW : **O**il **o**ut of **W**ater), V_{WoO} denotes the total volume of water droplets leaving the oil phase and entering their continuous phase (subscript WoO: **W**ater **o**ut of **O**il) as shown in Figure 3.1, t_h^{water} and t_h^{oil} denotes the horizontal residence times of water and oil phase respectively.

The volume of water contained in the separator is $V_W = A_W L$, where L is the length of the separator, A_W is the cross sectional area of the circular segment containing water with water height h_W as indicated in Figure 3.1.

Therefore the change in area is given by

$$\frac{dA_W}{dt} = \frac{1}{L}\frac{dV_W}{dt}.$$

The cross sectional area of the circular segment filled with water A_W is calculated as

$$A_W = r^2\Big[\cos^{-1}\Big(\frac{r - h_W}{r}\Big) - (r - h_W)\sqrt{2rh_W - h_W^2}\Big]. \qquad (3.2)$$

In order to obtain the differential equation for the height of water level, the time derivative of (3.2) is given by

$$\frac{dA_W}{dt} = 2\sqrt{2rh_W - h_W^2}\frac{dh_W}{dt}. \qquad (3.3)$$

The differential equation for the height of water level is

$$\frac{dh_W}{dt} = \frac{1}{2L\sqrt{2rh_W - h_W^2}} \frac{dV_W}{dt}. \tag{3.4}$$

Rate of change of total liquid level The rate of change of liquid volume in the separator is given by

$$\frac{dV_L}{dt} = q_{L,in} - q_{L,out}. \tag{3.5}$$

The total outflow is the amount of oil and water leaving the separator. Therefore the equation is rewritten as

$$\frac{dV_L}{dt} = q_{L,in} - q_{W,out} - q_{O,out}.$$

The volume of the segment in which the liquid is contained is given by $V_L = A_L L$ where A_L is the cross sectional area of the cylindrical segment containing total liquid.

The rate of change of liquid cross sectional area is

$$\frac{dA_L}{dt} = \frac{1}{L}(q_{L,in} - q_{W,out} - q_{O,out}).$$

The cross sectional area of the circular segment containing the total liquid A_L is calculated as

$$A_L = r^2 \Big[\cos^{-1}\Big(\frac{r - h_L}{r}\Big) - (r - h_L)\sqrt{2rh_L - h_L^2} \Big]. \tag{3.6}$$

In order to obtain the rate of change of liquid level, (3.6) is differentiated with time,

$$\frac{dA_L}{dt} = 2\sqrt{2rh_L - h_L^2}\frac{dh_L}{dt}. \tag{3.7}$$

Therefore, the differential equation for height of the liquid is

$$\frac{dh_L}{dt} = \frac{1}{2L\sqrt{2rh_L - h_L^2}}\frac{dV_L}{dt}. \tag{3.8}$$

Rate of change of system pressure:

As mentioned previously in Section 3.1, the gas phase of the separator is modeled by taking the ideal gas law into account. The ideal gas law is given by

$$\bar{p}V_G = n_G RT, \tag{3.9}$$

and $V_G = V_{Sep} - V_L$, where V_{Sep} is the total volume of the separator and V_G is the volume of the separator containing the gas phase both measured in cubic metres, $\bar{p}$ is the system pressure measured in Pascal, n_G is the number of gas moles, R is the universal gas constant, T is the temperature measured in Kelvin.

The ideal gas law given by (3.9) is differentiated, yielding

$$V_G\frac{d\bar{p}}{dt} + \bar{p}\frac{dV_G}{dt} = RT\frac{dn_G}{dt} + n_G R\frac{dT}{dt}.$$

Assuming isothermal compression, the above equation reduces to

$$V_G\frac{d\bar{p}}{dt} + \bar{p}\frac{dV_G}{dt} = RT\frac{dn_G}{dt}.$$

Since the volume of the gas phase in the separator is inversely proportional to the total liquid phase volume, volume change of the gas phase is given by

$$\frac{dV_G}{dt} = -\frac{dV_L}{dt} = q_{L,in} - q_{W,out} - q_{O,out}. \tag{3.10}$$

The component balance for the gas phase is given by

$$\frac{dn_G}{dt} = -\frac{\rho_G}{M_G}(q_{G,in} - q_{G,out}). \tag{3.11}$$

Using (3.10) and (3.11) the gas pressure dynamics is written as

$$V_G \frac{d\bar{p}}{dt} = RT\frac{\rho_G}{M_G}(q_{G,in} - q_{G,out}) + \bar{p}(q_{L,in} - q_{L,out}), \tag{3.12}$$

and $\bar{p} = p \cdot 10^5$, where p is measured in bars.

The scaled form of (3.12) is thus given by

$$\frac{dp}{dt} = \Big(\frac{RT}{V_G}\frac{\rho_G}{M_G}(q_{G,in} - q_{G,out}) + (\frac{p * 10^5}{V_G})(q_{L,in} - q_{L,out})\Big) \cdot 10^{-5}. \tag{3.13}$$

3.2.2 Droplet size distribution

As stated in section 3.1, it is assumed that there is absence of coalescence or breakage of the droplets in the separator. Therefore a fixed droplet size distribution is assumed. Droplet sizes of 10 classes were defined as shown in Table 3.1. It is observed from the Figure 3.2, that the droplets are approximately normally distributed.

Table 3.1: Droplet size distribution.

Droplet size class	Droplet size μm	Number of droplets
1	50	1e8
2	100	5e8
3	150	1e9
4	200	5e9
5	250	1e10
6	300	1e10
7	350	5e9
8	400	1e9
9	450	5e8
10	500	1e8

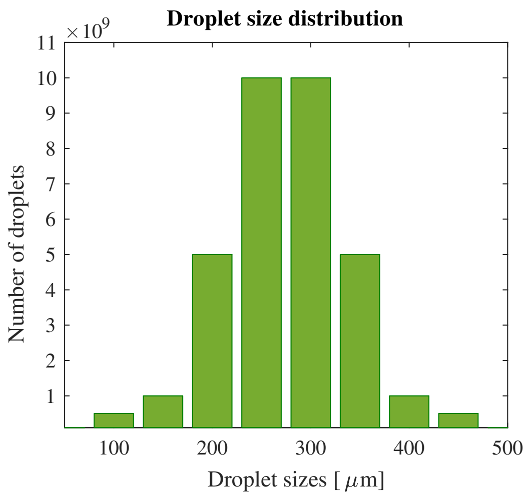


Figure 3.2: Droplet size distribution for 10 droplet classes

3.2.3 Horizontal Velocities

As the mixture enters the gravity separator, it is divided into the water continuous phase and the oil continuous phase, each of these phases containing dispersed oil and water droplets, respectively. As the continuous phase moves in the horizontal direction towards the outlet, the dispersed droplets move relative to their continuous phase. Therefore, the horizontal velocity of the droplets is assumed to be equal to the continuous phase in which they are present.

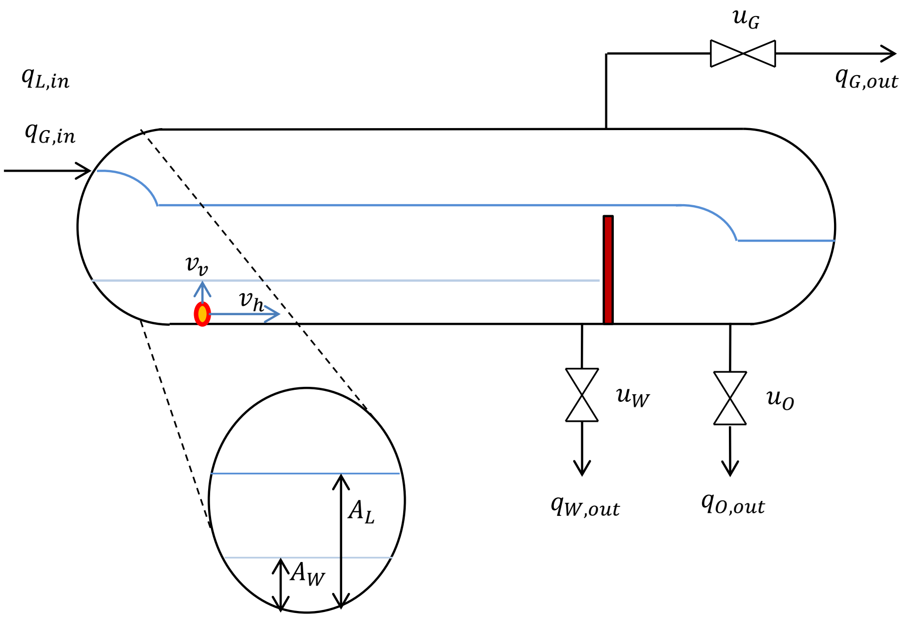


Figure 3.3: Schematic of the three phase gravity separator indicating velocity components on the droplets.

Horizontal velocity of water droplets:

$$v_h^{water} = \frac{q_{O,in}}{A_O}, \tag{3.14}$$

where $q_{O,in}$ is the flow rate of the oil into the oil continuous phase and A_O is the area of the circular segment containing the oil phase in which the water droplets are being dispersed.

The area of the segment containing oil is obtained using (3.2) and (3.6) and is given by $A_O = A_L - A_W$.

The total flow rate of the oil $q_{O,in}$ is the sum of the oil entering the oil continuous phase and the oil dispersed into the water phase

$$q_{O,in} = q_{L,in}\big(\beta\phi_{oo} + \alpha\phi_{wo}\big).$$

Horizontal velocity of oil droplets:

$$v_h^{oil} = \frac{q_{W,in}}{A_W}, \tag{3.15}$$

where $q_{W,in}$ is the total flow rate of the water into the water continuous phase and the water dispersed into the oil phase and A_W is the area of the circular segment containing the water phase in which the oil droplets are being dispersed.

The area of the segment containing water A_W is given in (3.2).

The flow rate of liquid into the water continuous phase $q_{W,in}$ is the amount of water entering the water phase and the amount of dispersed oil entering the water phase

$$q_{W,in} = q_{L,in}\big(\alpha\phi_{ww} + \beta\phi_{ow}\big).$$

3.2.4 Vertical Velocities

The terminal velocity for the dispersed particles is given by *Stokes' Law*

$$v_v = \frac{g d_p^2 (\rho_p - \rho)}{18\mu}, \tag{3.16}$$

where d_p is the diameter of the particle for every droplet class as mentioned in section 3.2.2, ρ_p is the density of the particle, ρ is the density of the continuous phase and μ is the viscosity of the continuous phase. The particles can be either water or oil droplets dispersed in their continuous phases.

Stokes' Law assumes that the particles are spherical in shape, the flow of the droplet is laminar and that the droplet movement is not hindered by the other droplets or the surface of the separator.

3.2.5 Horizontal and vertical residence time

In order to understand if the dispersed phase is being separated or is carried along with the bulk phase, it is important to understand the horizontal and the vertical residence times. The horizontal residence time for the droplets in their respective bulk phase can be defined as the amount of time the droplets spend in the separator before reaching the outlets.

The horizontal residence time is given by the equation:

$$t_h = \frac{L}{v_h^{oil,water}}. \tag{3.17}$$

The dispersed particles are considered to be separated from the continuous phase, if they rise or settle vertically to reach the interface well within the time it takes for the dispersed phase to move horizontally along the length of the separator.

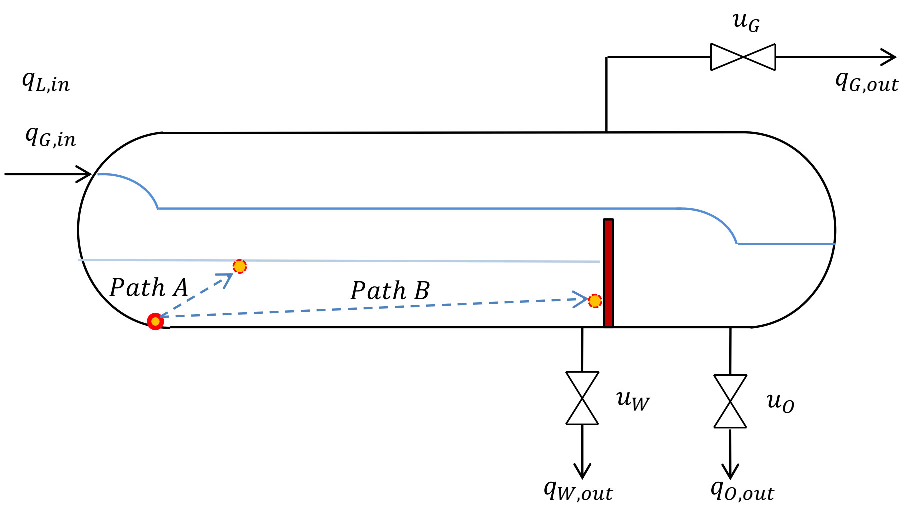


Figure 3.4: Illustration of the horizontal and the vertical residence times.

The vertical residence time is derived from the terminal velocity

$$t_v = \frac{h_{W,O}}{v_v}, \tag{3.18}$$

where $h_{W,O}$ is the level of water and oil layer ($h_O = h_L - h_W$), and v_v is the vertical velocity of the droplets (see section 3.2.1 and 3.2.4).

Comparing these residence time comparisons, the number of droplets entering their respective homogeneous phases can be calculated. Consequently, the total volume of the droplets leaving the bulk phases can be calculated. It is understood that if the time taken by the droplets to reach the interface is less than the horizontal residence time, the droplets will reach their homogeneous phase before reaching the outlet. This is illustrated by path A in Figure 3.4, and if the time taken by the droplets to reach their outlet is lesser than the vertical residence time, the droplets will be present at their outlet and is illustrated by path B in Figure 3.4.

3.2.6 Separation Efficiency

In section 3.2.5, the time taken by the droplets to reach their native phase or remain dispersed in their continuous phase was described. Based on this, the number and volume of oil droplets entering their native phase or remaining in the continuous water phase was estimated using a code scripted in MATLAB (see Appendix E.3) and similar calculations were made for the water droplets (see Appendix E.4). Separation efficiency is an important factor in order to measure the performance of separators. Separation efficiency of oil is defined as the fraction of initially dispersed oil being removed from the water phase and is given by

$$\eta_O = 1 - \frac{\text{Volume of oil droplets present in water outlet}}{\text{Initial volume of oil dispersed}}. \tag{3.19}$$

Similarly, the separation efficiency of water is defined as the fraction of initially dispersed water being removed from the oil phase and is given by

$$\eta_W = 1 - \frac{\text{Volume of water droplets present in the oil outlet}}{\text{Initial volume of water dispersed}}. \tag{3.20}$$

This means an efficiency value of 98% indicates that 2% of initially dispersed volume is contained at the outlet.

3.3 Formulation of Models in the DAE form

$$\frac{dh_W}{dt} = \frac{1}{2L\sqrt{2rh_W - h_W^2}}(q_{W,in} - q_{W,out} - \frac{V_{OoW}}{t_h^{water}} + \frac{V_{WoO}}{t_h^{oil}}) \tag{3.21a}$$

$$\frac{dh_L}{dt} = \frac{1}{2L\sqrt{2rh_L - h_L^2}}(q_{L,in} - q_{W,out} - q_{O,out}) \tag{3.21b}$$

$$\frac{dp}{dt} = \Big(\frac{RT}{V_G}\frac{\rho_G}{M_G}(q_{G,in} - q_{G,out}) + (\frac{p * 10^5}{V_G})(q_{L,in} - q_{L,out})\Big) \cdot 10^{-5} \tag{3.21c}$$

In section 3.2.5, we understood that if the vertical residence time of the droplets is less than the horizontal residence time, then the droplets will reach the oil water interface before they reach the outlet and if the vertical residence time of the droplets is greater than the horizontal residence time, then fractions of the single droplets size classes will remain in the bulk as dispersed phase.

Oil droplets in the water phase:

The system switches between these two models and since there are 10 droplet classes, the system takes the form

If ($t_h^{oil} \geq t_v^{oil}(i)$) [Oil droplets move to the interface]

$$l_p = h_W, \tag{3.22a}$$

$$V_{OoW}(i) = n(i)V(i) \tag{3.22b}$$

Else ($t_h^{oil} < t_v^{oil}(i)$) [Oil droplets remain dispersed]

$$l_p = t_h^{oil} v_v^{oil}(i), \tag{3.23a}$$

$$V_{OoW}(i) = \frac{l_p}{h_W} n(i)V(i) \tag{3.23b}$$

Water droplets in the oil phase:

If ($t_h^{water} \geq t_v^{water}(i)$) [Water droplets move to the interface]

$$l_p = h_O, \tag{3.24a}$$

$$V_{WoO}(i) = n(i)V(i) \tag{3.24b}$$

Else ($t_h^{water} < t_v^{water}(i)$) [Water droplets remain dispersed]

$$l_p = t_h^{water} v_v^{water}(i), \tag{3.25a}$$

$$V_{WoO}(i) = \frac{l_p}{h_O} n(i)V(i) \tag{3.25b}$$

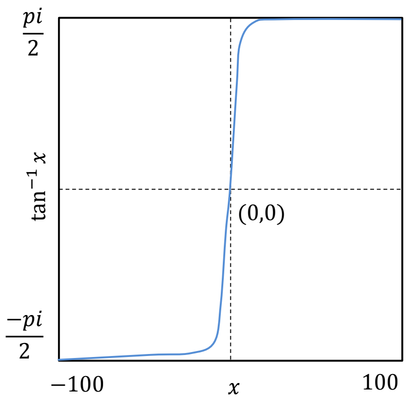
where $i \in$ [1,..,10], $n(i)$ is the number of particles in each particle class and $V(i)$ is the volume of particles in each particle class, $h_O = h_L - h_W$, $V_{OoW}(i)$ is the volume of oil leaving the water phase for particle class i and $V_{WoO}(i)$ is the volume of water leaving the oil phase for particle class i and l_p is the position of the droplets at the end of the separator.

These models (3.21), (3.22), (3.23), (3.24) and (3.25) are collectively termed hybrid DAEs (Najafi and Nikoukhah, 2006) or discontinuous DAEs (Powell et al., 2016; Agrawal et al., 2003).

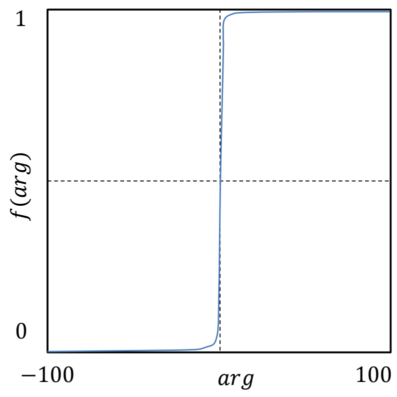
The if/else logic in the model is represented using greater than or equal to ($\geq$) and a less than or equal to ($\leq$) operators. The if/else formulation is presented as a set of continuous algebraic equations, in such a way that only binary solutions (0 or 1) are obtained for certain variables at the solution.

The logical expressions could be introduced into the optimization problems using mixed integer programming and solved using mixed integer programming algorithms. However, combining these algorithms and introducing them into the CasADi environment seemed challenging. Also, converting dynamic optimization problems into MINLP (Mixed Integer Nonlinear Programming) problems does not seem like an easy task (Powell et al., 2016). Therefore, the variables which activate the change in the dynamics of the process should be formulated such that it will have reasonable solutions at either zero or one. This kind of formulation allows logical disjunctions in the dynamic optimization problems to be solved without making use of mixed integer programming.

Therefore, logical statements are embedded as sets of algebraic equations thus maintaining mathematical continuity. There are many operators like sigmoid function or arctan functions that can be used for this purpose. In this thesis, arctan functions are used to express the functions which are discontinuous in nature.



(a) Plot of $f(x) = arctan(x)$ function where function value ranges from $[-\pi/2, \pi/2]$



(b) Plot of $f(arg) = \frac{\arctan(b \cdot arg)+\frac{\pi}{2}}{\pi}$ where the function value lies in $[0, 1]$ and b is the stiffness factor

Figure 3.5: Plot of arctan function on the left in the range $[-\pi/2, \pi/2]$ converted to a function on the right that takes binary values.

Therefore, the argument function which is the difference in horizontal and vertical residence times from Equations (3.22) and (3.23) is written as

$$arg_{oil} = t_h^{oil} - t_v^{oil}(i) \tag{3.26}$$

and the argument function from Equations (3.24), (3.25) is written as

$$arg_{water} = t_h^{water} - t_v^{water}(i) \tag{3.27}$$

Since, only the binary values are necessary to represent the if/else statements, the function is rewritten as Equation (3.28) such that either 0 or 1 is obtained for the variables at that solution. The value for b in f(arg) can be adjusted to make the curve smooth or stiff.

$$f(arg) = \frac{\arctan(b \cdot arg) + \frac{\pi}{2}}{\pi} \tag{3.28}$$

where arg stands for either arg_{water} or arg_{oil}.

Hence the algebraic equation is written in the form $f_i(z_i) = 0$ where z_i is the algebraic state given by the particle position l_p for 10 particle classes $i = [1, ..., 10]$. Depending on the argument function (3.28), if it is greater than or equal to zero, one set of 'if' statement is executed and if it is less than zero, the other set of 'if' statement will be executed causing $f(x)$ to take binary values. This can therefore be written as continuous set of algebraic equations

$$f_i(z_{oil,i}) = z_{oil,i} - (f(arg_{oil})h_W) + (1 - f(arg_{oil}))t_h^{oil}v_v^{oil}(i) \tag{3.29}$$

$$f_i(z_{water,i}) = z_{water,i} - (f(arg_{water})h_O) + (1 - f(arg_{water}))t_h^{water} v_v^{water}(i) \tag{3.30}$$

In this way, the volume of oil leaving the water phase is calculated for the individual particle class and then summed up to obtain the total oil volume leaving the water phase.

$$V_{OoW}(i) = \frac{z_{oil,i}}{h_W} n(i)V(i) \tag{3.31}$$

Therefore the total volume of oil leaving the water phase is given by

$$V_{OoW} = \sum V_{OoW}(i) \tag{3.32}$$

Similarly, the volume of water leaving the oil phase is calculated for the individual particle class and then summed up to obtain the total water volume leaving the oil phase.

$$V_{WoO}(i) = \frac{z_{water,i}}{h_O} n(i)V(i) \tag{3.33}$$

and the total volume of water leaving the oil phase is given by

$$V_{WoO} = \sum V_{WoO}(i) \tag{3.34}$$

The differential equations (3.21) and the set of algebraic equations (3.29), (3.30) for gravity separator are collectively termed semi-explicit DAE (Differential Algebraic Equations)

All the models discussed above have been implemented as separate script files (see Appendix E.1 - E.4) and the plant model is solved using ode15s solver.

Chapter 4

Model Predictive Control Theory

4.1 Introduction

Model Predictive Control (MPC) is a control strategy that calculates control inputs by solving constrained optimal control problem over a finite time horizon.

Currently, model predictive control also referred to as receding horizon control (RHC) is the most attractive feedback strategy particularly for linear processes subject to constraints (Findeisen and Allgöwer, 2002). MPC uses a plant model to predict the output trajectories. Many real systems are nonlinear in nature. In these cases, if the plant is approximated by a Linear Time Invariant (LTI) model, it is insufficient to accurately capture the nonlinearities in the model.

4.2 MPC Elements

There are three main stages which constitute the method of model predictive control strategy. First, the future values of the output variables are predicted based on the process model and current measurements. Second, the optimization problem is solved using a suitable algorithm. Finally, for a particular time instant, the manipulated variables are applied to the process which minimizes the cost or the objective function. This procedure is repeated at subsequent intervals for the entire prediction horizon.

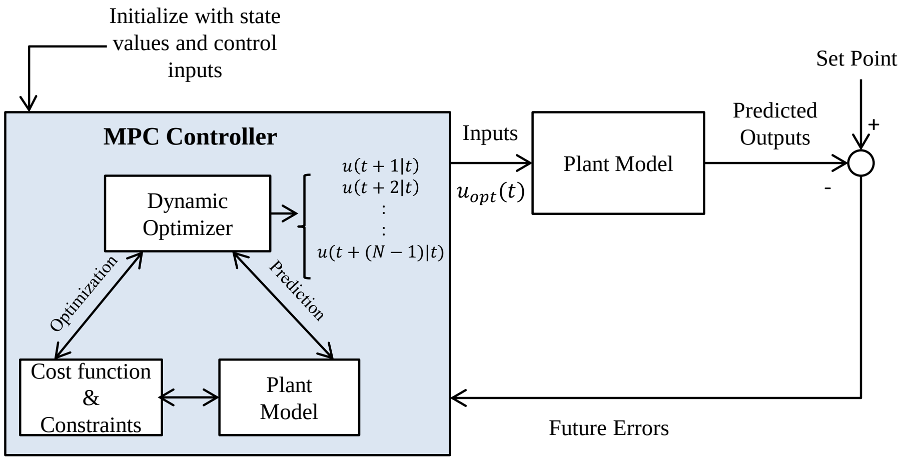


Figure 4.1: Schematic of basic MPC structure

The elements of the Model Predictive Controller are shown is Figure 4.1. Plant model in the figure represents the models of the gravity separator system. In the MPC controller mechanism, the model predicts the process output in the future and the optimization calculates the sequence of optimal control inputs by optimizing a determined set of reference criteria. The criterion, also known as the objective or cost function can take the form of a quadratic function of the errors between the predicted outputs and the setpoint or the reference value. These errors will be assigned a weighting factor depending on the relative importance of the manipulated variables, control variables or the constraint violations.

In short, the MPC consists of

1. At a given sampling instant i, given the initial values of the control inputs and the states, the system output for the entire prediction horizon N is obtained $y((i+k)|i)$ along with future control signals $u((i+k)|i)$.
2. Optimize the determined set of reference criterion such that the process is kept close to the setpoint
3. The first control signal calculated is applied to the plant $u(i|i)$, while the others are rejected
4. For the subsequent sampling instants, step 1 is repeated with the new value of the state obtained from the plant.

This is illustrated in Figure 4.2 where the system is simulated for a total time of t_f seconds. Along the prediction horizon T_p, the optimal control sequence is calculated for N sampling instants, that is $N = \frac{T_p}{h}$ where h is the sampling period. Assuming that the sampling period is 1 second each, the optimization problem is solved at every sampling instant and only the first part of the optimal control sequence is applied to the plant denoted by u_{opt}. Towards the end of the simulation, a set of optimal control inputs is obtained $u_{opt} = [u_0, u_1, u_2, ..., u_{t_f-1}]$.

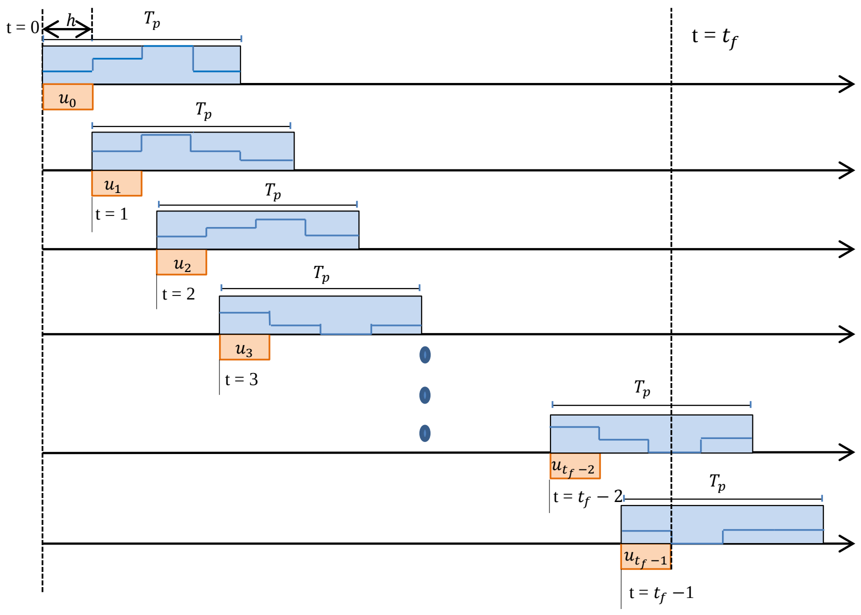


Figure 4.2: Receding horizon strategy for a total simulation time of t_f seconds

In the next section, the formulation of NMPC problem and also the detailed method to solve the optimization problem will be described.

4.3 Formulation of NMPC Optimization Problem

Figure 4.3 illustrates the basic principle of Model Predictive Control. Based on the measurements obtained at time t, the controller predicts the trajectory of the system dynamics over the prediction horizon T_p and determines the control input over the control horizon T_c such that the cost function or the performance criteria is optimized. For simplicity, the prediction horizon T_p is taken to be equal to the control horizon T_c. The time difference h between the measurements can vary, however it is assumed that the measurements take place between fixed sampling instants h. At time $t + h$, a new measurement is obtained and by making use of it, the entire procedure of prediction and optimization is repeated to obtain a new optimal control input and the prediction horizon is moved forward.

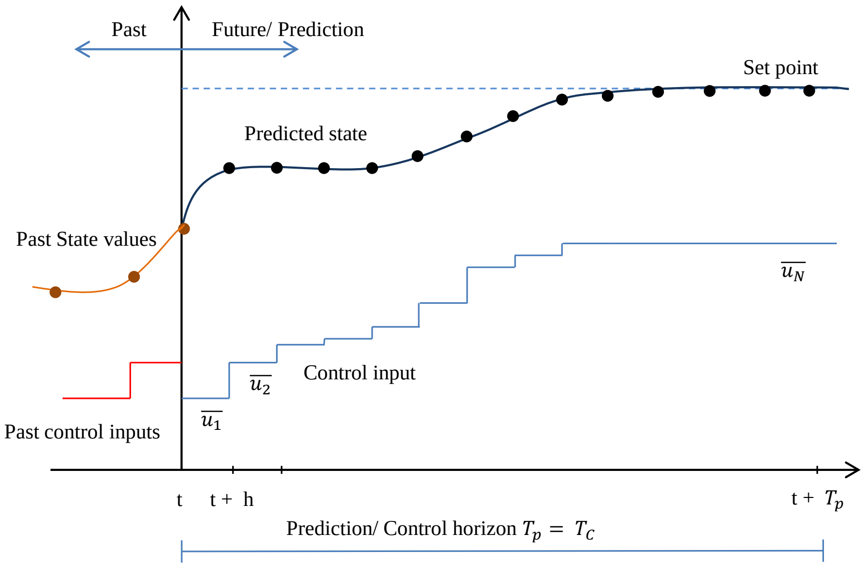


Figure 4.3: Illustration of the predictive control algorithm (Findeisen and Allgöwer, 2002)

As seen from section 3.3, the gravity separator system is modelled as a semi-explicit index-1 DAE (differential algebraic equation) of the form

$$\dot{x} = f(x, z, u, b), \tag{4.1a}$$

$$0 = g(x, z, u, b) \tag{4.1b}$$

where x is the vector of differential states, z is the vector of algebraic states, the vector of control inputs is given by u, and b are the time invariant parameters.

We assume that the state of the system x is measurable. In NMPC, the control input applied to the process in the interval $[t \;\; t+h]$ can be obtained by the repeated solution of finite horizon optimal control problem of the form

$$\min_{x,z,u} \int_{t_0}^{t_f} L(t, x(t), z(t), u(t), b)dt + \sum_{k=0}^{N-1} \Delta u_k^T R \Delta u_k \tag{4.2a}$$

$$\text{subject to} \quad \dot{x}(t) = f(t, x(t), z(t), u(t), b), \tag{4.2b}$$

$$0 = g(t, x(t), z(t), u(t), b), \tag{4.2c}$$

$$x(t_0) = x_0, \tag{4.2d}$$

$$x^{low} \leq x(t) \leq x^{high}, \tag{4.2e}$$

$$z^{low} \leq z(t) \leq z^{high}, \tag{4.2f}$$

$$u^{low} \leq u(t) \leq u^{high}, \tag{4.2g}$$

$$-\Delta u^{high} \leq \Delta u(t) \leq \Delta u^{high} \tag{4.2h}$$

The objective function is the typical optimal control Lagrange term as discussed in section 2.4 together with a quadratic penalty on Δu_k where R is the diagonal weighting matrix. The optimization variables are the discrete control inputs u, and the trajectories x and z.

The Lagrange term introduced in the objective function is in its continuous form and has to be discretized using the direct collocation algorithm. Whereas, the control inputs are introduced within the direct collocation algorithm as discrete variables. Therefore, in the above formulation (4.2a) the continuous Lagrange term is augmented with the differences of the piecewise constant control inputs, Δu which are discretized with N sampling instants in the time interval $[t_0 \;\; t_f]$.

4.4 Direct Collocation Method to solve NMPC Optimization Problem

Section 2.4.1 briefly described the methods used in general to solve the optimal control problem. Among the direct methods seen earlier, direct collocation method is employed in this thesis to solve the optimal control problem (4.7b). The basic idea is to discretize the differential equations and transcribe the optimization problem into a finite dimensional NLP, which is later solved by IPOPT. The third order direct collocation gives a polynomial approximation of (4.1) as shown in Figure 4.4.

The optimization time horizon is divided into N elements such that each interval is in $[t_k \;\; t_{k+1}]$ where $k \in [0, ..., N-1]$. The trajectories x and z are approximated using low order polynomials in the interval $[t_k \;\; t_{k+1}]$ known as collocation polynomials. The collocation polynomials are formulated by choosing K collocation points and these are kept same for all elements. In this thesis, Lagrange interpolating polynomials are being used as the collocation polynomials given by

$$P_{k,i}(t) = \prod_{j=0, j\neq i}^{K} \frac{t - t_{k,j}}{t_{k,i} - t_{k,j}} \tag{4.3}$$

with the property

$$P_{k,i}(t_{k,l}) = \begin{cases} 1 & \text{if } l = i \\ 0 & otherwise \end{cases}$$

Let $t_{k,i}$ be the collocation points in the interval $[t_k \;\; t_{k+1}]$ and $i \in [0, .., K]$, where $(x_{k,i} \;\; z_{k,i} \;\; u_{k,i})$ is the simplified notation of $(x(t_{k,i}) \;\; z(t_{k,i}) \;\; u(t_{k,i}))$. The collocation polynomials are then formulated by interpolating the values $(x_{k,i} \;\; z_{k,i})$ in that time interval. The interpolation yields

$$x(x_k, t) = \sum_{i=0}^{K} x_{k,i} \; P_{k,i}(t) \tag{4.4a}$$

$$z(z_k, t) = \sum_{i=1}^{K} z_{k,i} \; P_{k,i}(t) \tag{4.4b}$$

where $x_{k,i}$ are the parameters and $P_{k,i}$ are the Lagrange polynomials. Additionally, the degrees of freedom (DOF) for the differential states is $K + 1$ whereas it is K for the algebraic states. The reason for an additional DOF in case of the differential states is to ensure continuity so that the shooting gaps are closed. Algebraic states can be discontinuous and therefore do not require an additional DOF.

In order to ensure integration accuracy, Radau collocation points are used. The Radau collocation points are obtained from Radau quadrature, which is similar to the Gaussian quadrature. However, one collocation point is clearly defined at one end of the time interval, rather than having all points completely inside the time interval like the Legendre collocation points (Powell et al., 2016).

The control inputs are assumed to be piecewise constant in the interval $[t_k \;\; t_{k+1}]$ and given by

$$u(t) = u_k \tag{4.5}$$

The polynomials for $\dot{x}$ are obtained by differentiating the polynomials for x

$$\dot{x}_{k,i} = \frac{1}{(t_{k+1} - t_k)} \sum_{r=0}^{K} \dot{P}_{k,l}(t_{k,i}) \; x_{k,r} = \frac{1}{(t_{k+1} - t_k)} \sum_{r=0}^{K} \alpha_{r,i} \; x_{k,r} \tag{4.6}$$

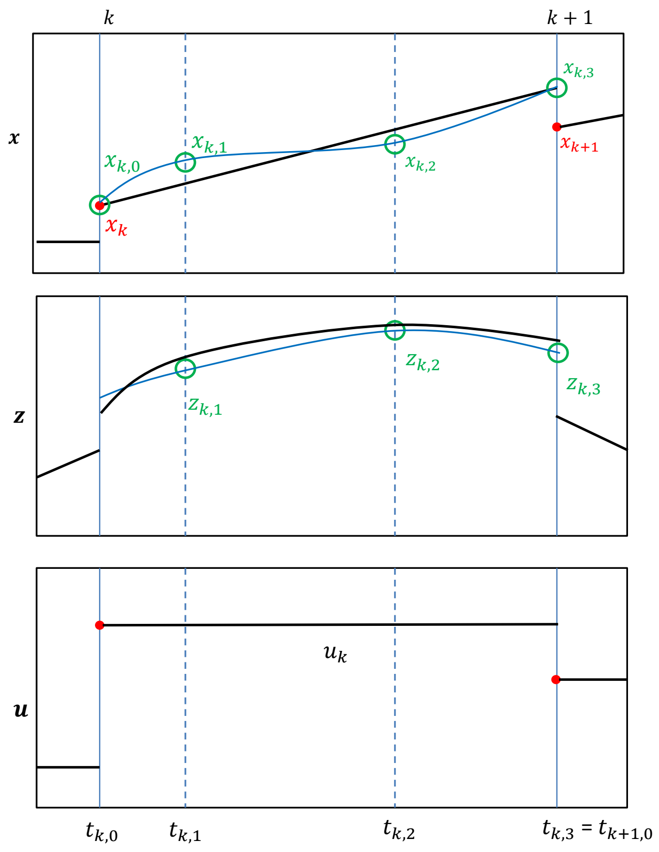


Figure 4.4: Schematic representation of third order direct collocation in the single sampling interval $[t_k \quad t_{k+1}]$. The control input is piecewise constant in the interval $[t_k \quad t_{k+1}]$.

4.5 Finite Dimensional NLP

The variables in the transcribed NLP are chosen such that it contains the variable values at all collocation points $\dot{x}_{k,i}, x_{k,i}, z_{k,i}$, the control input values u_k the initial value of the state $x_{k,0}$ and the parameters b. The transformation results in the NLP

$$\min_{x_{k,i},z_{k,i},u_k} \sum_{k=0}^{N-1}\sum_{i=0}^{K}\omega_i L(t_{k,i},x_{k,i},z_{k,i},u_k,b) + \sum_{k=0}^{N-1}\Delta u_k^T R\Delta u_k \tag{4.7a}$$

$$\text{subject to}\quad \dot{x}_{k,i} - f(t_{k,i},x_{k,i},z_{k,i},u_k,b) = 0, \tag{4.7b}$$

$$0 = g(t_{k,i},x_{k,i},z_{k,i},u_{k,i},b), \tag{4.7c}$$

$$x_{1,0} = x_0, \tag{4.7d}$$

$$x^{low} \leq x_{k,i} \leq x^{high} \tag{4.7e}$$

$$z^{low} \leq z_{k,i} \leq z^{high} \tag{4.7f}$$

$$u^{low} \leq u_{k,i} \leq u^{high} \tag{4.7g}$$

$$-\Delta u^{high} \leq \Delta u_k \leq \Delta u^{high} \tag{4.7h}$$

$$0 = x_{k,K} - x_{k+1,0} \tag{4.7i}$$

$$\dot{x}_{k,i} = \frac{1}{(t_{k+1}-t_k)}\sum_{r=0}^{K}\alpha_{r,i}\, x_{k,r} \tag{4.7j}$$

The Lagrange term in (4.2a), which is in the continuous form is transcribed into discrete form in (4.7a) using Gauss quadrature within each element such that the integral is approximated by a summation using quadrature weights ω_i (see Appendix B). The states need to be continuous with respect to time and hence the shooting gap constraint or the continuity constraint is introduced given by (4.7i). Equation (4.7j) is introduced to specify the relation between $\dot{x}$ and x. The weights $\alpha_{r,i}$ are the result of differentiating Lagrange polynomials for the states.

4.6 Continuous Logic in an NMPC problem

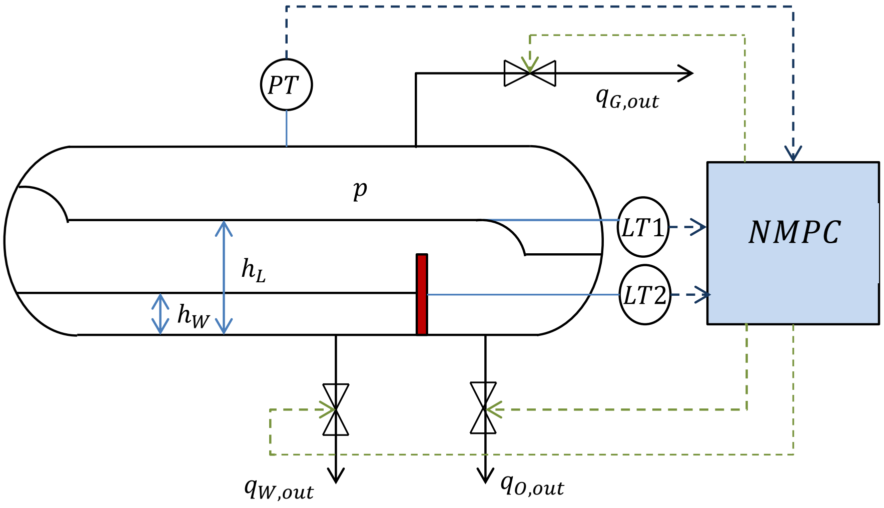


Figure 4.5: Control structure for a gravity separator

The MPC problem attempts to minimize the deviations from the reference value for the water level h_W, liquid level h_L and sytem pressure p, without making large control moves. Therefore, a quadratic performance index is employed, where the deviations are squared at the end of each time interval, and are weighted accordingly, and are finally summed up to produce an objective function to be minimized. This generates an optimization problem which is subject to system models and the constraints.

As discussed in section 2.6, the MPC provides setpoints to the regulatory layer (PI controllers), and this layer manipulates the control valves to keep the plant at steady state values. However, in this thesis the regulatory layer is not considered, and the NMPC is directly used to manipulate the control valves using the optimal control inputs obtained after solving the optimization problem.

Chapter 5

Simulation Results and Discussion

In this thesis, the dynamic optimization problem is transformed into a discrete NLP problem using direct collocation method where radau collocation points are used. This problem has been implemented in CasADi using MATLAB as the modeling environment. Once the NLP problem is created, it is solved using the IPOPT plugin, which is an open source primal- dual interior point method incorporated in CasADi installations (Andersson, 2013). At every iteration, the first value of the obtained optimal input is sent to the plant model. The plant model has been implemented in MATLAB as separate m-files or script files. From the plant model, the states computed are sent to the optimizer and these values will be the starting point for the next iteration. The simulation results in this section are described based on the Gullfaks-A production rates (Laleh et al., 2013; Backi and Skogestad, 2017). The simulation parameters are presented in Appendix A.

As it has been discussed in section 3.3, the models are expressed as semi-explicit DAEs, described by the differential equations and two sets of algebraic equations each for the oil droplets dispersed in the continuous water phase and the water droplets dispersed in the continuous oil phase. However, while implementing both sets of algebraic equations in the CasADi environment, an error message was given by the IPOPT plugin ”Converged to a point of local infeasibility. Problem may be infeasible”. Since, the main interest was to minimize the total amount of oil present in the water phase, simulations were carried out using the differential equations 3.21 and considering only the algebraic equations 3.29 which represent the oil droplets present in the water phase, therefore discarding the other set of algebraic equations that represent the water droplets in the oil phase 3.30.

Table 5.1: Setpoint values for the controlled variables for different cases

Parameters	Values [Case 1, III, 1V]	Values [Case II]
h_W^{SP}	1.2	1.2, 1.4, 1.6, 1.8
h_L^{SP}	2.5	2.5
p^{SP}	68.7	68.7

Case I: Comparative study of the effect of noise on the system control behavior

In real processes, the controller has to handle noisy measurements. It is important that the controller should not be very sensitive to the noise, because there is a possibility that the control valves will be subjected to continuous wear and tear and can be worn out easily. Therefore, a random noise was added to the states both of high order magnitude and low order magnitude to understand the effect of noise on the system behavior. The NMPC formulation for this case is written as follows, and the setpoint values are displayed in Table 5.1

$$
\begin{aligned}
\min_{q_{W,out},q_{O,out},q_{G,out}} \quad & \sum_{k=0}^{N-1}(h_W(k)-h_W^{SP})^2+(h_L(k)-h_L^{SP})^2+(p(k)-p^{SP})^2 \\
+ \quad & \sum_{k=0}^{N-1}(q_{W,out}(k)-q_{W,out}(k-1))^2 \\
+ \quad & \sum_{k=0}^{N-1}(q_{O,out}(k)-q_{O,out}(k-1))^2 \\
+ \quad & \sum_{k=0}^{N-1}(q_{G,out}(k)-q_{G,out}(k-1))^2 && \text{(5.1a)} \\
\text{Subject to} \quad & (3.21), && \text{(5.1b)} \\
& (3.29), && \text{(5.1c)} \\
& 0.9 \le h_W \le 1.9, && \text{(5.1d)} \\
& 2.2 \le h_L \le 3.3, && \text{(5.1e)} \\
& 50 \le p \le 100, && \text{(5.1f)} \\
& -0.05 \le q_{W,out}(k)-q_{W,out}(k-1) \le 0.05, && \text{(5.1g)} \\
& -0.05 \le q_{O,out}(k)-q_{O,out}(k-1) \le 0.05, && \text{(5.1h)} \\
& -0.05 \le q_{G,out}(k)-q_{G,out}(k-1) \le 0.05 && \text{(5.1i)} \\
& h_W(0),\ h_L(0),\ p(0) \ \text{ given} && \text{(5.1j)} \\
& q_{W,out}(-1),\ q_{O,out}(-1),\ q_{G,out}(-1) \ \text{ given} && \text{(5.1k)}
\end{aligned}
$$

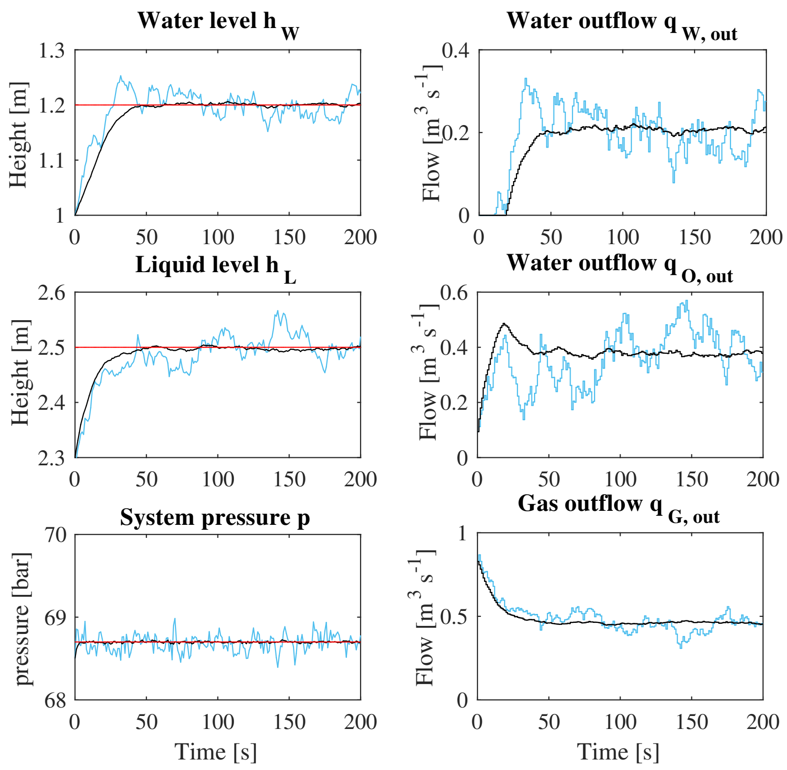


Figure 5.1: Right: Manipulated variables for low magnitude noise addition (black) and high magnitude noise addition (blue) Left: State variables for low magnitude noise addition (black) and high magnitude noise addition (blue) and the setpoint values (red)

For the purpose of generating noise, `randn` function from MATLAB toolbox, which draws a random scalar from a standard normal distribution is used. In Figure 5.1, a low magnitude measurement noise is added to the states, which is of the order of 10^{-4} on the water and the liquid level, and 10^{-3} on the system pressure. Additionally, a high measurement noise is added to the states, which is of the order of 10^{-3} on water level and the liquid level, and 10^{-2} on the system pressure. It is observed that the controller is very sensitive to increase in measurement noise thereby causing the manipulated variables to be noisy as well.

In this thesis, all simulations for case II, case III and case IV have been carried out for low magnitude measurement noise.

CASE II: Setpoint Tracking

The objective is to track setpoints, where the output has to converge to a given reference value as close as possible while fulfilling the constraints. Here, the water level h_W, total liquid level h_L and the system pressure p are considered for tracking. The NMPC formulation for this case is given by (5.1), where the setpoints for water, liquid levels and the system pressure are displayed in Table 5.1.

In Figure 5.2, the plots on the left hand side show the variables to be controlled or the state variables h_W, h_L and p. The three plots on the right hand side show their corresponding manipulated variables $q_{W,out}$, $q_{O,out}$ and $q_{G,out}$. A series of stepwise setpoint changes were employed to the water level. Setpoint changes of +0.2 m were given at every 200 s of simulation time, with the initial water level of 1 m. It is observed from the Figure that the controlled variables are kept at their set-point values by the NMPC controllers where the set-point values are displayed in red.

It is also seen that, when a step is given to the water level at every 200 s, the controller tries to close the control valve, therefore the water outflow drops to zero. Ideally, there should be a gradual drop in water outflow, since the rate of change of flow constraint has been implemented in the code (see Appendix E.5, line). However, there is a sudden fall to zero at the point where the step change in level is introduced, which means that the rate of change of flow constraint is violated at those time instants. Whereas, the implementation of the rate of change constraints work well at those time instants where no step change is introduced. Similar observation is made for the outlet oil flow rate, where there is a rise in the oil outflow which is as expected, however the rise is not gradual where the step

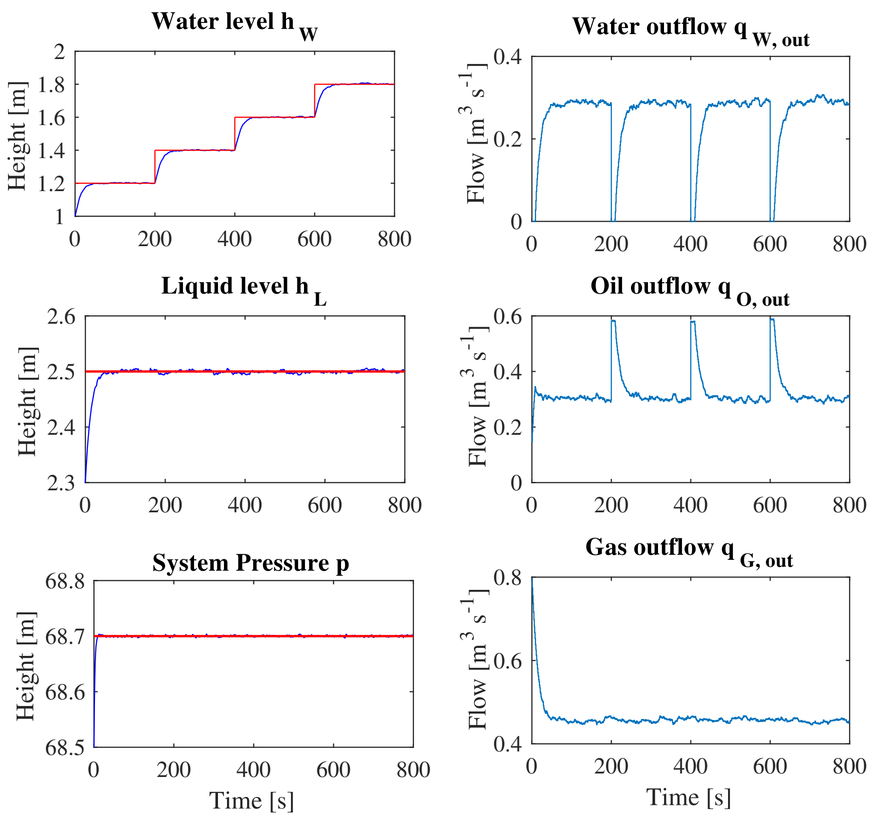


Figure 5.2: Setpoint tracking of gravity separator controlled with NMPC. State variables h_W, h_L and p on the left with their respective manipulated variables $q_{W,out}$, $q_{O,out}$ and $q_{G,out}$ on the right

change in the water level is given, therefore violating the constraints (5.1h) at those time instants (see Appendix D). In this formulation, equal weights were given to both the states as well as the change in manipulated variables. The reason being that it was equally important to keep the states at their reference values, and also to avoid the violation of control move constraints.

CASE III: Disturbance Rejection

As it has been discussed in chapter 3, the separator levels and the system pressure are often disturbed by varying inflow rate into the separator system. Usually, the separator is disturbed by the liquid inflow $q_{L,in}$ and the gas inflow $q_{G,in}$. The optimization problem given by (5.1) is solved and the aim of the controller is to maintain the levels and the system pressure at their desired reference values, (see Table 5.1) despite the disturbances due to the inflows to the system.

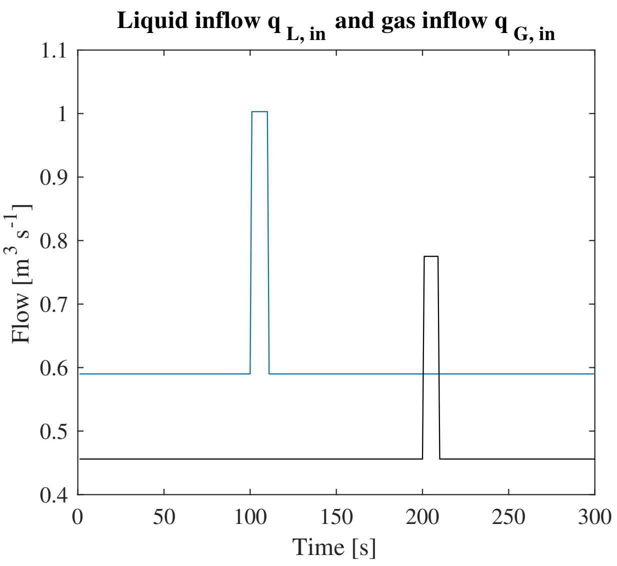


Figure 5.3: Pulse disturbance in liquid inflow $q_{L,in}$ (blue) and disturbance in gas inflow $q_{G,in}$ (black)

Figure 5.4 shows the response of the levels and system pressure of the gravity separator subject to pulse disturbances in the liquid inflow rate $q_{L,in}$ and the gas inflow rate $q_{G,in}$ one at a time. At t = 100 s, a sudden increase in liquid inflow rate is applied, therefore increasing from the steady state value $q_{L,in}$ = $0.59\,\mathrm{m^3\,s^{-1}}$ to $1.003\,\mathrm{m^3\,s^{-1}}$ and then at t = 110 s back to the initial value as shown in the Figure 5.3. It can be seen that, with the sudden increase in the liquid inflow, the water

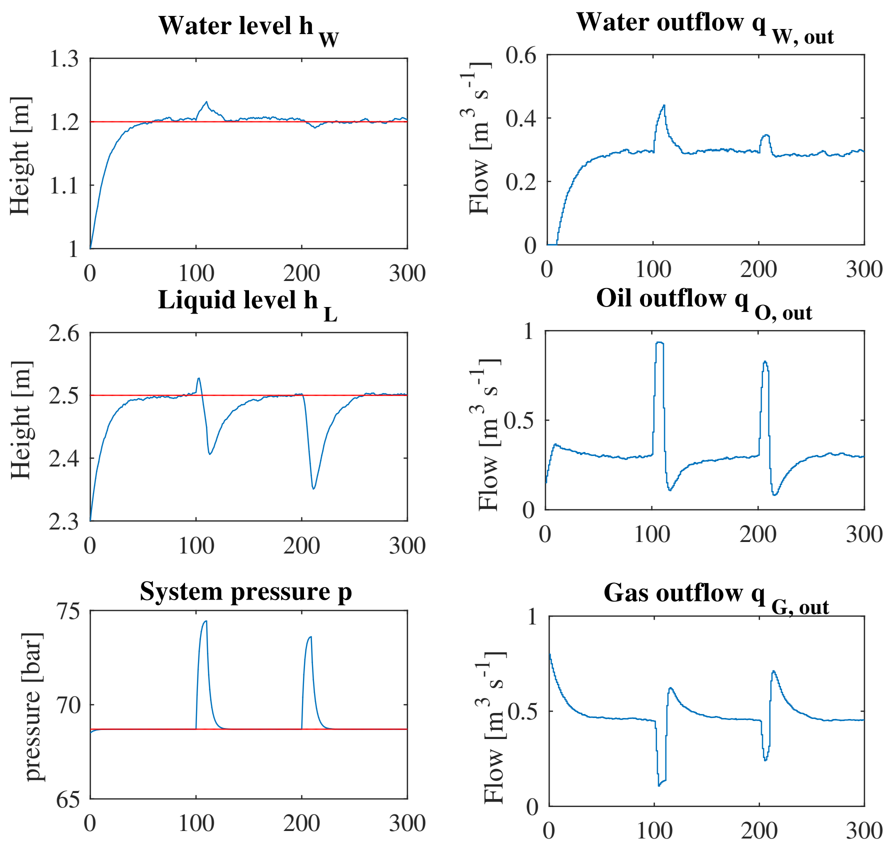


Figure 5.4: Disturbance rejection when there is sudden increase in inflow liquid $q_{L,in}$ and inflow gas $q_{G,in}$. Left: h_W, h_L and p, and the dotted red line indicates the reference values. Right: Manipulated variables $q_{W,out}$, $q_{O,out}$ and $q_{G,out}$

level and the liquid level increases, thereby increasing the pressure of the system. The controller tries to flush the accumulated liquid out of the system to restore the levels to their reference values.

Similarly, a pulse disturbance in the gas inflow rate is given to the system. At t = 200 s, a sudden increase in gas inflow rate is applied, therefore increasing from the steady state value $q_{G,in}$ = $0.456\,\mathrm{m^3\,s^{-1}}$ to $0.775\,\mathrm{m^3\,s^{-1}}$ and then at t = 210 s back to the initial value as shown in the Figure 5.3. It can be observed that the disturbance in $q_{G,in}$ results in abrupt increase in the system pressure. The controller starts flushing more gas out of the system to bring back the pressure to normal level. It can be seen that other manipulated variables change appropriately to bring back water and liquid levels back to the reference values.

CASE IV: Maximize the volume of oil entering its homogeneous phase (V_{OoW})

In this case, the objective is to maximize the total volume of oil entering its native (mother) phase, and obtain the optimal value of the water level while satisfying the constraints. It is desired to control the water level in the range given by (5.2d). Therefore, the objective function does not include tracking for the water level since maximizing V_{OoW} and tracking of the water level to the desired setpoint cannot be done simultaneously. Since the goal is to minimize the objective function while maximizing V_{OoW}, this term is preceded with a minus sign in the objective function. Additionally, because it is very important to maximize V_{OoW}, a higher weight was assigned to this term in comparison to the weights on state variables and the control moves. The NMPC in this case is formulated as follows.

$$
\begin{aligned}
\min_{q_{W,out}, q_{O,out}, q_{G,out}} \quad & \sum_{k=0}^{N-1} 100(h_L(k) - h_L^{SP})^2 + 100(p(k) - p^{SP})^2 \\
+ \quad & \sum_{k=0}^{N-1} 100(q_{W,out}(k) - q_{W,out}(k-1))^2 \\
+ \quad & \sum_{k=0}^{N-1} 100(q_{O,out}(k) - q_{O,out}(k-1))^2 \\
+ \quad & \sum_{k=0}^{N-1} 100(q_{G,out}(k) - q_{G,out}(k-1))^2 \\
- \quad & \sum_{k=0}^{N-1} 1000(V_{OoW}(k))^2 && (5.2a) \\
\text{Subject to} \quad & (3.21), && (5.2b) \\
& (3.29), && (5.2c) \\
& 0.9 \leq h_W \leq 2.2, && (5.2d) \\
& 2.2 \leq h_L \leq 3.3, && (5.2e) \\
& 50 \leq p \leq 100, && (5.2f) \\
& -0.05 \leq q_{W,out}(k) - q_{W,out}(k-1) \leq 0.05, && (5.2g) \\
& -0.05 \leq q_{O,out}(k) - q_{O,out}(k-1) \leq 0.05, && (5.2h) \\
& -0.05 \leq q_{G,out}(k) - q_{G,out}(k-1) \leq 0.05 && (5.2i) \\
& h_L(0),\ p(0),\ q_{W,out}(-1),\ q_{O,out}(-1),\ q_{G,out}(-1) \ \text{given} && (5.2j)
\end{aligned}
$$

In Figure 5.5, the two plots on the top show the state variables h_L, p and the plots on the bottom show the manipulated variables $q_{O,out}$, $q_{G,out}$. It is seen from the plots of the controlled variables, that the NMPC is able to keep them at their desired reference values, (see Table 5.1). It can be observed from the oil outflow plot, that there are a few drops in the outlet flow rate thus violating the constraints 5.2h. The reason is that, while solving the above optimization problem, IPOPT gives a message ”Maximum number of iterations reached” at those points which means that the solver could not find the optimal control input at those time instants (see Appendix D, Figure D.3). Similar observation is also made in Figure 5.6.

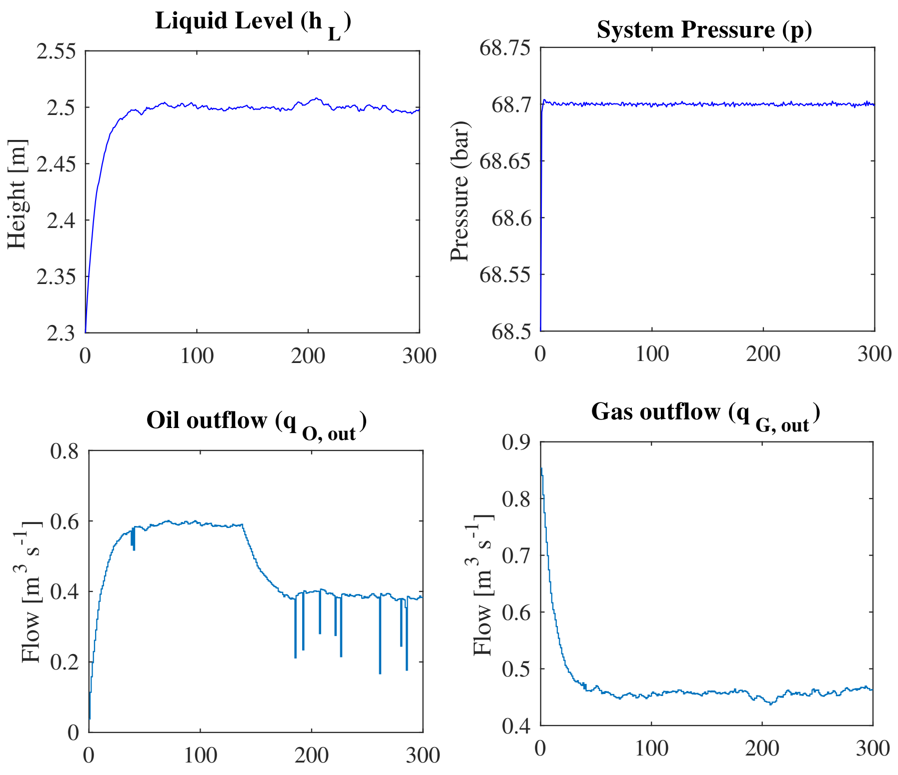


Figure 5.5: Controlled variables h_L and p on top with their respective manipulated variables $q_{O,out}$ and $q_{G,out}$ on the bottom

In Figure 5.6, the plot on top shows the water level h_W, which starts at an initial value of 1 m, and stabilizes at 2 m. The red dotted lines indicate the upper and lower bounds given by (5.2d). It is observed from the Figure that the controlled variable h_W is kept within its bounds and stabilizes at an optimal value. It can be seen that the optimal value for the level of water in order to maximize the amount of oil leaving the water phase and entering its continuous phase V_{OoW} is at 2 m.

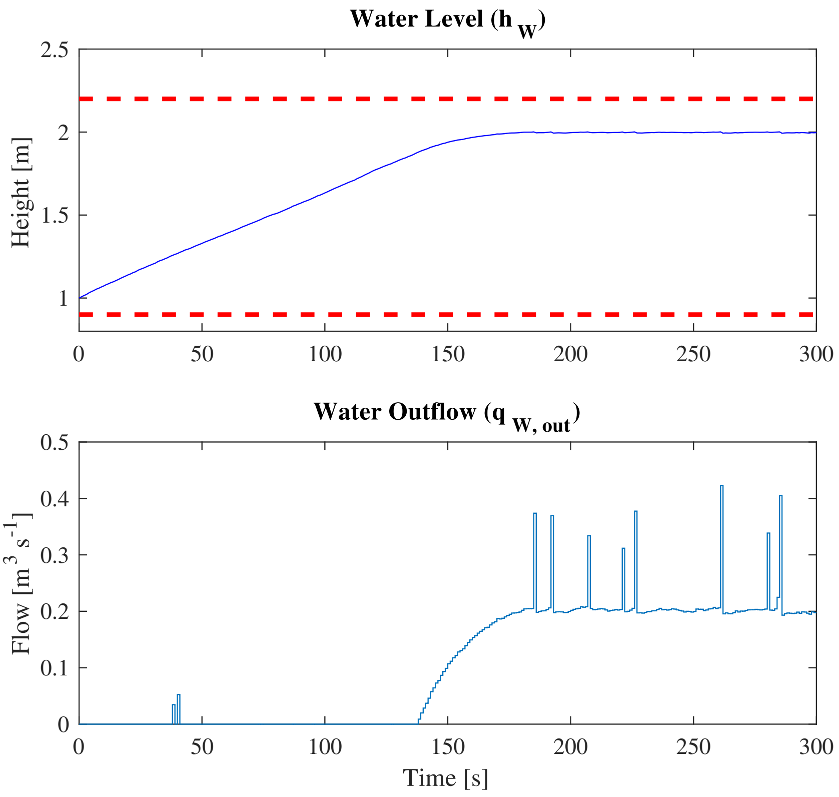


Figure 5.6: Top: Controlled variable h_W indicated in (blue) and the bounds on h_W in dotted red. Below: Manipulated variable $q_{W,out}$

In order to investigate the optimal water level, the total volume of oil at the water outlet, efficiency of oil removal from the water phase as well as the horizontal and the vertical residence times were analyzed. The lower bound on the water level is chosen to be 0.9 m, while the upper bound is set to 2.2 m. In order to investigate the optimal water level, four different water levels are considered within the given bounds.

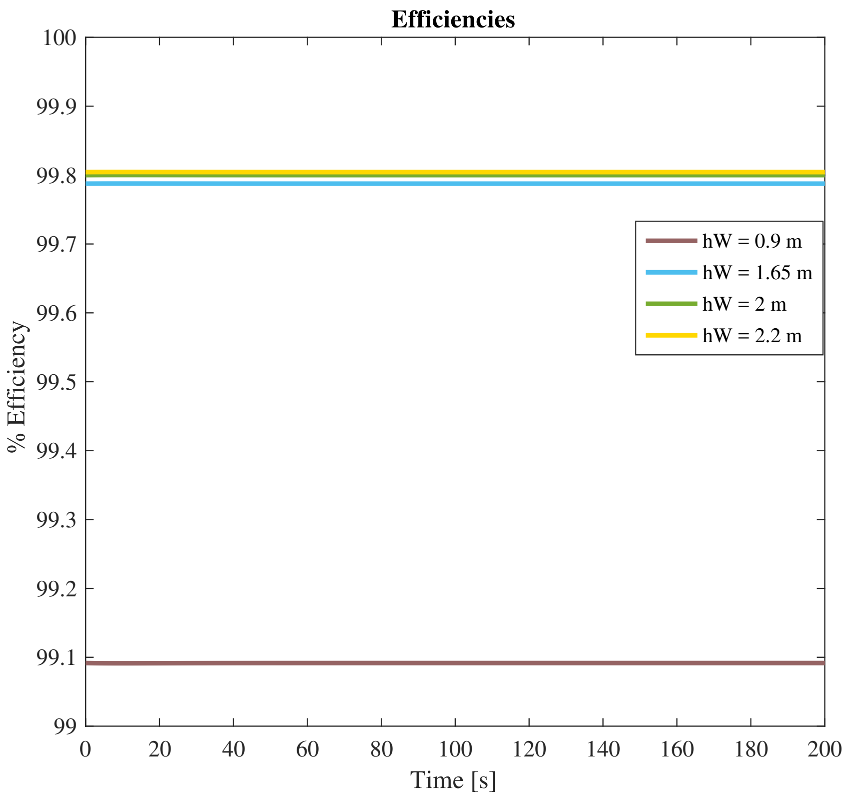


Figure 5.7: Efficiencies of oil removal from the water phase for different water levels

From Figure 5.7, it can be observed that at lower water levels the efficiency of oil removed from the water phase is smaller, whilst it increases with an increase in the water level. It can be noticed that the efficiency of oil removal for a water level greater than 2 m is almost constant at around 99.8%.

This can also be observed from the oil droplet distribution at the water outlet, which is demonstrated in the following Figures 5.8 - 5.11.

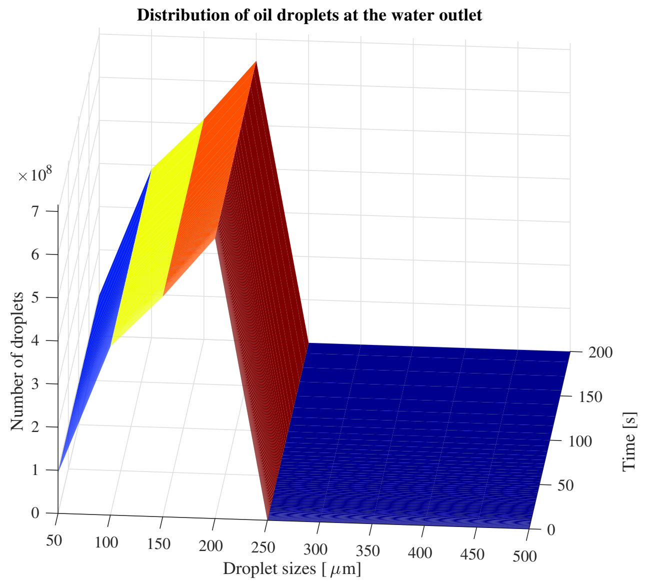


Figure 5.8: Distribution of oil droplets at the water outlet for water level = 0.9 m

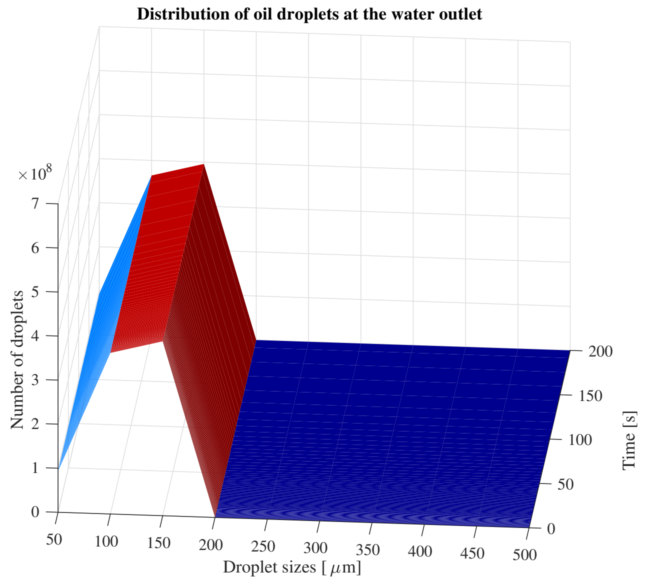


Figure 5.9: Distribution of oil droplets at the water outlet for water level = 1.65 m

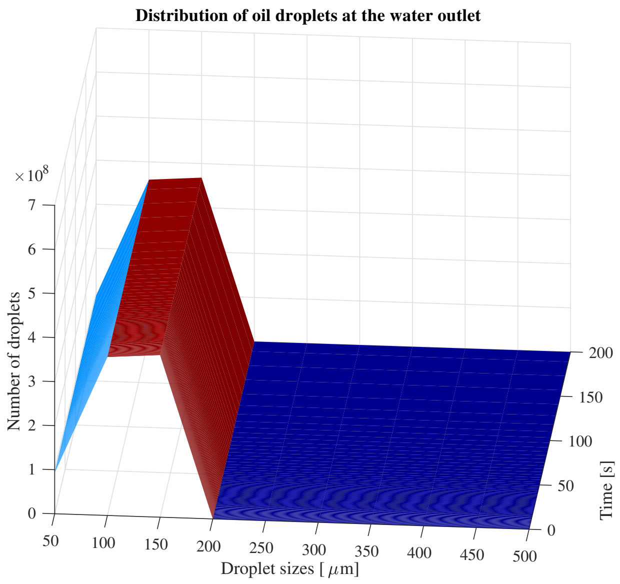


Figure 5.10: Distribution of oil droplets at the water outlet for water level = 2 m

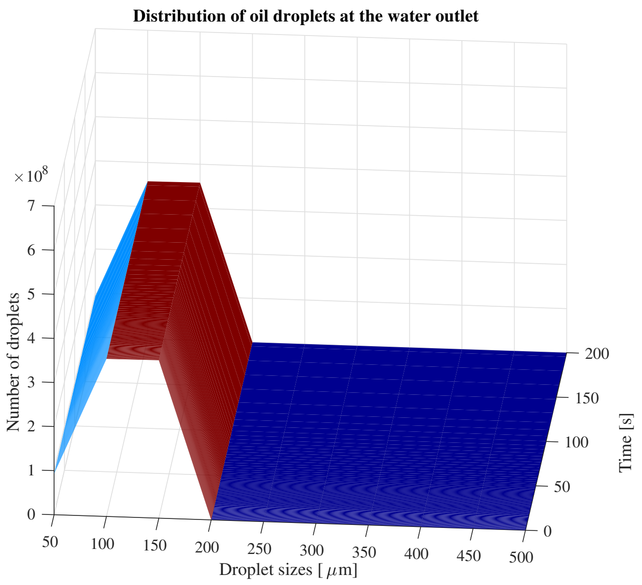


Figure 5.11: Distribution of oil droplets at the water outlet for water level = 2.2 m

Figures 5.8 - 5.11 show the amount of oil droplets present at the water outlet for the corresponding droplet size and time for different water levels, respectively. From Figure 5.8, it can be seen that for a water level of 0.9 m, the cut-off oil droplet size (minimum size above which the dispersed droplets present at the water outlet is zero) is observed to be 250 µm. This means that all oil droplets larger than 250 µm leave out of the water phase into their mother phase. With the increase in water level, the cut-off oil droplet size decreases (200 µm), meaning that more particle classes leave the water phase and enter their mother phase. Therefore, also the total volume of oil droplets leaving the water phase (V_{OoW}) will be higher. This is due to the fact that if the water level is smaller, the horizontal velocity of the droplets is higher. Hence, the droplets need less time to reach the outlet in comparison to the time it takes to reach their oil-water interphase (see Appendix C).

It can be observed from Figures 5.10 and 5.11 that for levels of water above 2 m the oil distribution at the water outlet remains almost nearly the same.

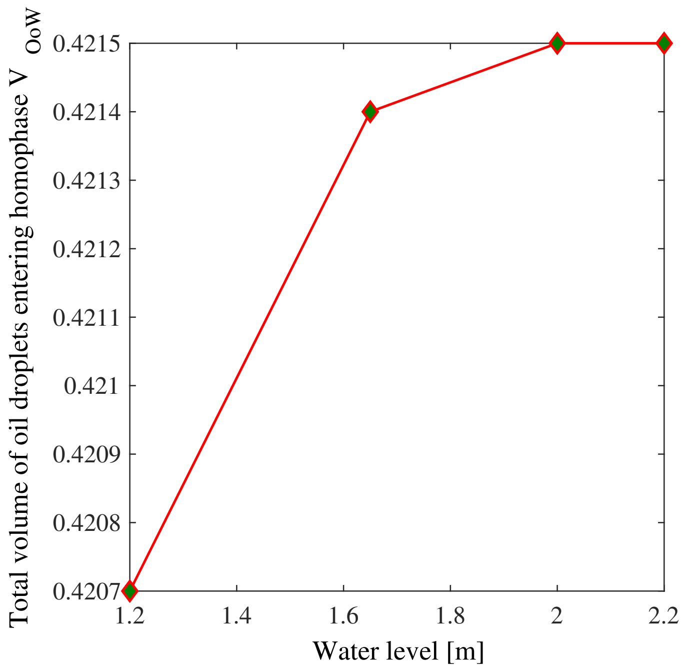


Figure 5.12: Total volume of oil droplets entering their continuous phase V_{OoW} for the respective water levels.

From Figure 5.12 it can be seen that with the increase in water level, the total volume of oil droplets leaving the water phase and entering the oil phase is numerically almost the same for water levels above 2 m. This explains the optimal level of the water obtained.

Chapter 6

Conclusion

In this thesis, a Nonlinear Model Predictive Control (NMPC) strategy is proposed to control the levels and the system pressure of a three phase gravity separator. The NMPC optimization problem was formulated for the three phase system and solved using the Direct collocation approach (Simultaneous method) using CasADi within the MATLAB programming environment. Nonlinear dynamic models were derived for the water level, liquid level and the pressure of the gas in the system based on inflow and outflow dynamics. The equations were implemented in MATLAB as separate script files and were solved using ode15s solver. The models were formulated as a semi-explicit DAE system and the discontinuous algebraic equations were embedded into the optimizing controller as continuous algebraic equations using $\arctan$ functions.

The simulations were conducted for several cases in order to analyze the performance of the NMPC controller on the system behavior. Firstly, the system behavior was studied to analyze the performance of the controller to sensitivity in measurement noise. It was observed that the controller was sensitive to the large magnitude in measurement noise. This could lead to the wear and tear of the control valves very easily. Therefore for the successive cases, the system performance was studied for a low magnitude noise added to the measurements. Secondly, the controller was studied for the tracking of the setpoint using NMPC scheme. It was seen that the states converged to their given setpoints while satisfying the constraints. However, at the time instants, where the step change in water level was introduced, constraint violation of the rate of change in manipulated variables was observed. Thirdly, the controller was studied for the case, where sudden disturbances in liquid and gas inflows to the gravity separator system were introduced. It was observed that the controller was able to reject the disturbances and regulate the output at the desired setpoint without any offset. Finally, a case was investigated to maximize the total volume of oil entering its native phase, and thereby to obtain the optimal water level. It was noticed that the optimal level of water was at 2 m and within the given bounds. This was because, at water level greater than 2 m, the efficiency of oil removal from the water phase was almost constant. Also, the distribution of oil droplets at the water outlet remained almost the same. Additionally, the numerical values from the simulation show that the total volume of oil entering its continuous phase at a water level greater than 2 m is almost constant.

From the simulations, it is observed that the controller is effectively able to track the setpoints, reject disturbances for a low magnitude of measurement noise since the controller is sensitive to high magnitude measurement noise thereby reacting positively to changes in the parameters. Results also agree with the simulations, and it can be concluded that the Nonlinear Model Predictive Controller is a good control method for oil-water-gas separators.

Future Work

The fluid dynamics inside the separators involve a lot of complexity, and hence it is necessary to study the phenomena like droplet breakage and coalescence. In actuality, there exists an initial distribution of droplet size at the inlet of the separator, and the size distribution changes inside the separator due to nonuniform flow patterns and factors such as coalescence and droplet breakage. This evolution of the droplet size distribution can be estimated by the use of Population Balance Equations (PBE) and incorporating these models can further improve the exactness of estimation.

For the absence of droplet breakage and coalescence, only Stokes law is sufficient to determine the terminal settling velocity of the droplets which has been considered for the purpose of modeling. However, while considering the coalescence and the break up of droplets which exists commonly in the separators, the velocity is hindered and the Stokes law for settling will need to take into account the correction factor. Introducing PBEs within the CasADi environment seems a bit challenging and also the simulations might take a lot of computation time.

Bibliography

Agrawal, J., Moudgalya, K. M., Pani, A. K., June 2003. Sliding motion of discontinuous dynamical systems described by differential algebraic equations. Science Direct 36 (2), 735–740.

Albuquerque, F., Riberio, O., Morais, M., May 2013. Subsea processing systems: Future vision. Offshore Technology Conference.

Andersson, J., October 2013. A General-Purpose Software Framework for Dynamic Optimization. PhD thesis, Arenberg Doctoral School, KU Leuven, Department of Electrical Engineering (ESAT/SCD) and Optimization in Engineering Center, Kasteelpark Arenberg 10, 3001-Heverlee, Belgium.

Backi, C. J., Skogestad, S., 2017. A simple dynamic gravity separator model for separation efficiency evaluation incorporating level and pressure control, proceedings of the 2017 American Control Conference.

Diehl, M., Bock, H. G., Diedam, H., Wieber, P.-B., 2006. Fast Direct Multiple Shooting Algorithms for Optimal Robot Control. In: Fast Motions in Biomechanics and Robotics, Optimization and Feedback Control, September, 2005.

Vol. 340 of Lecture Notes in Control and Information Sciences. Springer, Heidelberg, Allemagne, pp. 65–93.

Durdevic, P., Pedersen, S., Yang, Z., 2017. Challenges in modelling and control of offshore de-oiling hydrocyclone systems. Journal of Physics.

Findeisen, R., Allgöwer, F., 2002. An introduction to nonlinear model predictive control. In Proceedings of the 21st Benelux Meeting on Systems and Control, 119–141Denmark.

Geiger, B., August 2009. Unmanned Aerial Vehicle Trajectory Planning with Direct Methods. Master's thesis, The Pennsylvania State University.

Kaya, A. S., December 2007. On the Optimal Operation of the Open Plate Reactor. Master's thesis, Dalarna University.

Kirches, C., October 2010. Fast Numerical Methods for Mixed-Integer Nonlinear Model-Predictive Control. Ph.D. thesis, Heidelberg Graduate School of Mathematical and Computational Methods for the Sciences.

Laleh, A. P., Svrcek, W. Y., Monnery., W. D., February 2013. Computational fluid dynamics-based study of an oilfield separator–part II: An optimum design. Society of Petroleum Engineers 2, 52 – 59.

Larsson, T., Skogestad, S., 2000. Plantwide control: A review and a new design procedure. Vol. 21 of 4. Modeling, Identification and Control, doi:10.4173/mic.2000.4.2, pp. 209–240.

Magnusson, F., Åkesson, J., June 2015. Dynamic optimization in jmodelica.org. processes 4 (3), 471–496.

Mendes, P. R. C., Normey-Rico, J., Plucenio, A., Carvalho, R. L., July 2012. Disturbance estimator based nonlinear mpc of a three phase separator. Symposium on Advanced Control of Chemical Processes.

Moraes, C., da Silva, F., Oliveira, D., October 2013. Subsea versus topside processing - conventional and new technologies. Offshore Technology Conference.

Najafi, M., Nikoukhah, R., September 2006. Modeling and simulation of differential equations in scicos. The Modelica Association, 177–185.

Nocedal, J., Wright, S. J., 2006. Numerical Optimization, 2nd Edition. Springer Science+Business Media, LLC.

Nunes, G., 2001. Design and analysis of multivariable predictive control applied to an oil water gas separator: A polynomial approach. Master's thesis, University of Florida.

Powell, K. M., Eaton, A. N., Hedengren, J. D., Edgar, T. F., March 2016. A continuous formulation for logical decisions in differential algebraic systems using mathematical programs with complementarity constraints. Processes.

Prescott, C. N., January 2012. Subsea Separation and Processing of Oil, Gas & Produced Water Past, Present and Future Why We Need It Now. Fluor Offshore Solutions, USA, 3rd Edition.

Stokke, S., Strand, S., Sjong, D., 1994. Model Predictive Control of a Gas-Oil-Water Separator Train. Springer, Methods of Model Based Process Control, Ch. 5, pp. 701–715.

Appendices

Appendix A

Simulation Parameters

Table A.1: Simulation Parameters (Backi and Skogestad, 2017; Laleh et al., 2013).

Parameter	Description	Value
M_G	Molar mass of the gas	0.01604 $kg\,mol^{-1}$
R	Universal gas constant	8.314 $kg\,m^2\,s^{-2}\,mol^{-1}\,K^{-1}$
T	Temperature	328.5 K
g	Gravitational acceleration	9.8 $m\,s^{-2}$
$q_{G,in}$	Volumetric gas inflow	0.456 $m^3\,s^{-1}$
$q_{L,in}$	Volumetric liquid inflow	0.59 $m^3\,s^{-1}$
μ_O	Viscosity of oil	0.001 $kg\,(m\,s)^{-1}$
μ_W	Viscosity of water	0.0005 $kg\,(m\,s)^{-1}$
ρ_G	Density of gas	49.7 $kg\,m^{-3}$
ρ_O	Density of oil	831.5 $kg\,m^{-3}$
ρ_W	Density of water	1030 $kg\,m^{-3}$

Appendix B

Quadrature coefficients

The aim is to evaluate the integral $I = \int_a^b f(x)\,dx$. Assuming that the values of the function $f(x)$ is known at $n+1$ points; $x_0, ..., x_n \in [a, b]$. With the function values $f(x_0), ..., f(x_n)$, the Lagrange interpolating polynomial is written in the form

$$L_n(x) = \sum_{i=0}^{n} f(x_i) P_i(x), \tag{B.1}$$

where

$$P_i(x) = \prod_{j=0, j\neq i}^{n} \frac{x - x_j}{x_i - x_j}, \qquad 0 \leq i \leq n \tag{B.2}$$

The integral $I = \int_a^b f(x)\,dx$ is then approximated by the integral of $L_n(x)$:

$$\int_a^b f(x)\,dx \approx \int_a^b L_n(x)\,dx = \sum_{i=0}^{n} f(x_i) \int_a^b P_i(x)\,dx = \sum_{i=0}^{n} \omega_i f(x_i) \qquad \text{(B.3)}$$

This way the integral is converted to a summation

$$\int_a^b f(x)\,dx = \sum_{i=0}^{n} \omega_i f(x_i)$$

Therefore the quadrature coefficients ω_i in Equation B.3 is expressed by the equation given below

$$\omega_i = \int_a^b P_i(x)\,dx. \qquad \text{(B.4)}$$

Appendix C

Residence time of droplet classes

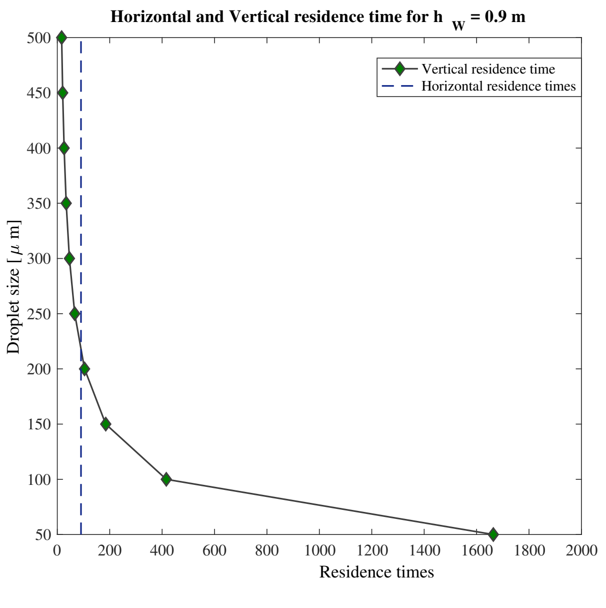


Figure C.1: Horizontal and vertical residence times of oil droplets in the water phase for water level = 0.9 m. The horizontal residence time = 90.45 s.

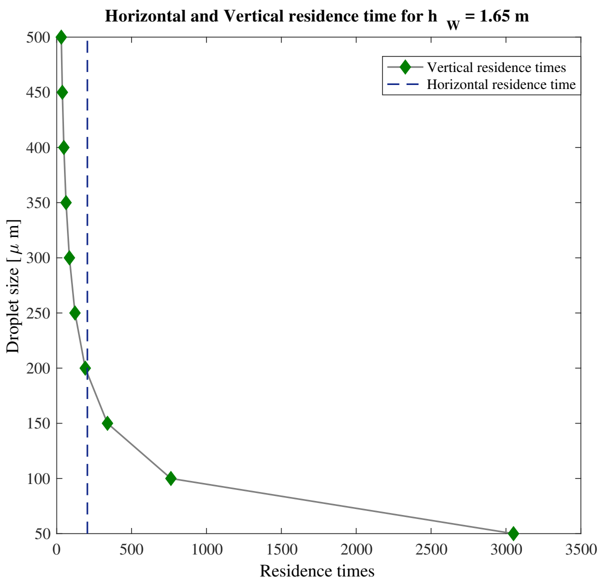


Figure C.2: Horizontal and vertical residence times of oil droplets in the water phase for water level = 1.65 m. The horizontal residence time = 204.715 s.

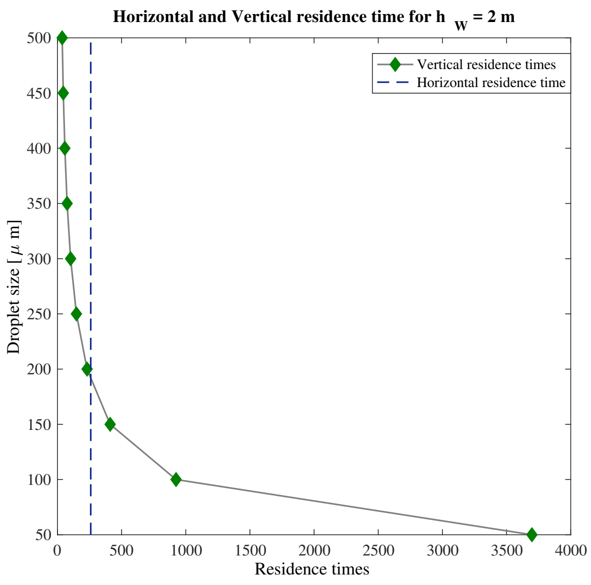


Figure C.3: Horizontal and vertical residence times of oil droplets in the water phase for water level = 2 m. The horizontal residence time = 259.58 s.

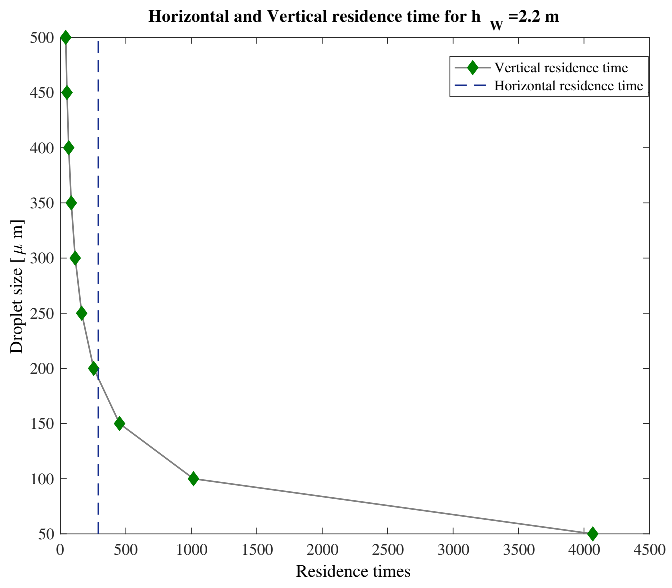


Figure C.4: Horizontal and vertical residence times of oil droplets in the water phase for water level = 2.2 m. The horizontal residence time = 289.96 s.

The above figures indicate the horizontal (blue dotted line) and vertical residence times (green diamond marked) of droplets in the water phase. It is understood that, if the horizontal residence time is greater than the vertical residence times, the droplets will leave to their continuous phase.

It can be observed from Figure C.1 that all droplets above 250 μm leave to their continuous phase for a water level of 0.9 m. With the increase in water level, the cut-off droplet size is 200 μm as can be seen from Figures C.2 - C.4 meaning all droplets above the cut-off droplet size enter their mother phase.

Appendix D

Rate of change constraints

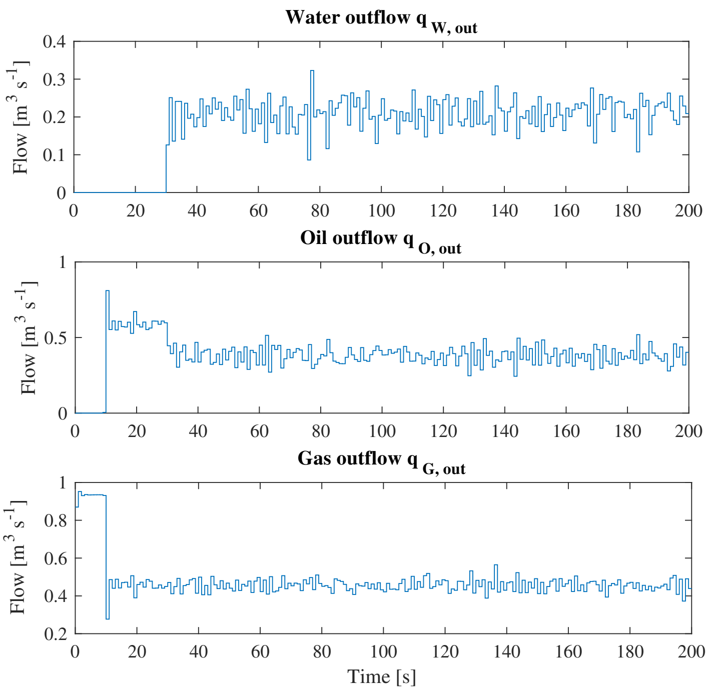


Figure D.1: The control inputs without the inclusion of rate of change constraints

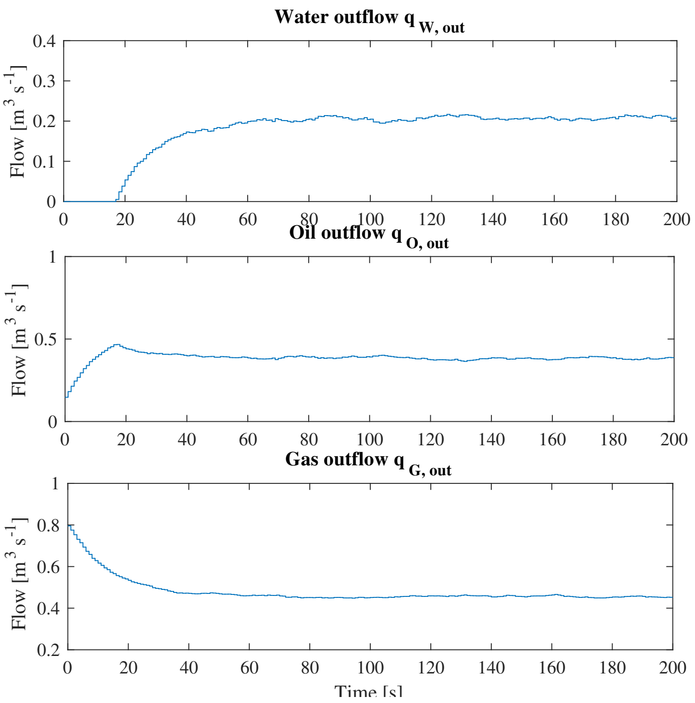


Figure D.2: The control inputs with the inclusion of rate of change constraints

Without the implementation of the rate of change constraint $\Delta u^T R \Delta u$ the plots for the control inputs is as shown in Figure D.1, and with the implementation of the above mentioned constraint, the plot as shown in Figure D.2 is obtained. The inclusion of this constraint in the optimization problem penalizes the control moves ($\Delta u_t = u_t - u_{t-1}$), thereby reduces the wear and tear of valves.

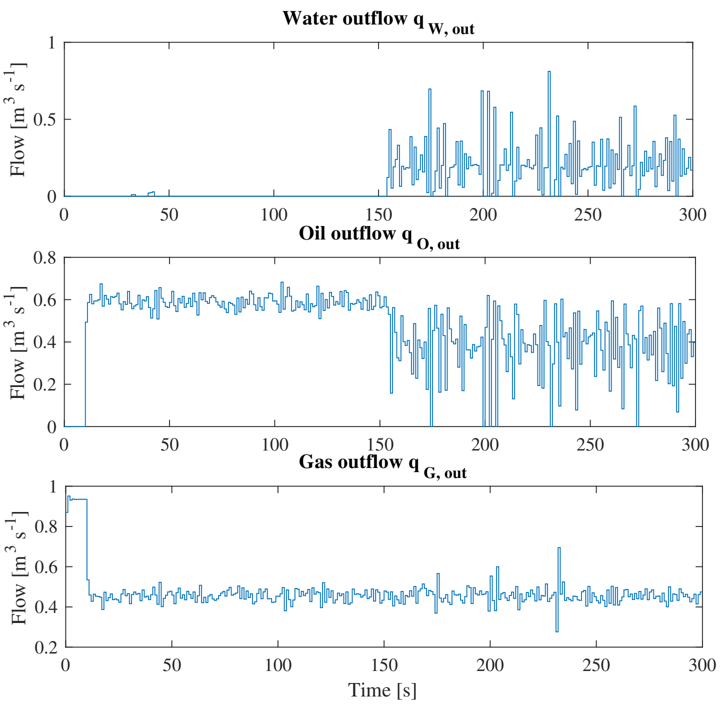


Figure D.3: The control inputs without the inclusion of rate of change constraints

Without the implementation of the rate of change constraint $\Delta u^T R \Delta u$ the plots for the control inputs is as shown in Figure D.3.

Appendix E

MATLAB CODE

E.1 velocity_inflows.m

```
function [vel_oil,vel_water,qWin] = velocity_inflows(qLin,hL,hW,
    alpha,phi_wo,phi_ow,radius)

abs_wo = alpha*phi_wo;                  % Fraction of water going into
     the continuous oil phase (subscript wo -> water to oil)
abs_ww = alpha*(1-phi_wo);              % Fraction of water going into
     the continuous water phase (subscript ww -> water to water)
abs_ow = (1-alpha)*phi_ow;              % Fraction of oil going into
    the continuous water phase (subscript ow -> oil to water)
abs_oo = (1-alpha)*(1-phi_ow);          % Fraction of oil going into
    the continuous oil phase (subscript oo -> oil to oil)

flow_fraction_w = abs_ww + abs_ow;  % Split factor describing the
    amount of oil and water going into the water phase
flow_fraction_o = abs_oo + abs_wo;  % Split factor describing the
    amount of oil and water going into the oil phase
```

```

AL = radius^2/2*(2*acos((radius-hL)/radius)-sin(2*acos((radius-hL)
    /radius)));    % Cross sectional area of liquid
AW = radius^2/2*(2*acos((radius-hW)/radius)-sin(2*acos((radius-hW)
    /radius)));    % Cross sectional area of water
AO = AL-AW;                                                   % Cross
    sectional area of oil

 vel_oil = flow_fraction_o*qLin/AO;       % Velocity of the
    continuous oil phase
 vel_water = flow_fraction_w*qLin/AW;     % Velocity of the
    continuous water phase

qWin = qLin*flow_fraction_w;           % Fraction of liquid inflow
    going into to water phase
```

E.2 gravity_ode.m

```
 function dydt = gravity_ode(t,y, parameters_plant)

% ASSIGN EMPTY VECTOR
 dydt = zeros(3,1);

% DEFINE THE PASSING PARAMETERS
 qLin= parameters_plant(1);
 alpha = parameters_plant(2);
 phi_wo = parameters_plant(3) ;
 phi_ow = parameters_plant(4);
 radius = parameters_plant(5);
L = parameters_plant(6) ;
R = parameters_plant(7);
Temp = parameters_plant(8) ;
```

```
density_gas = parameters_plant(9);
MG = parameters_plant(10);
qGin = parameters_plant(11);
qWout = parameters_plant(12);
qOout = parameters_plant(13);
qGout = parameters_plant(14);
hW = parameters_plant(15);
hL = parameters_plant(16);
p = parameters_plant(17);

% CALLING THE FUNCTIONS
[vel_oil ,vel_water ,qWin] = velocity_inflows(qLin ,hL ,hW, alpha ,
    phi_wo ,phi_ow ,radius);   % horizontal velocity , inflows
    calculation model
[~, VOoW_vector, ~]  = oildroplets_in_water(vel_water , hW, L);
                                 % oildroplets_in_Water calculation
    model
[~, VWoO_vector, ~]  = waterdroplets_in_oil(vel_oil , hL, hW, L);
                              % waterdroplets_in_oil calculation model



% DEFINING THE VARIABLES
time_oil = L/vel_oil;                % used in dhW/dt model
time_water = L/vel_water;            % used in dhW/dt model
constant = R*Temp*density_gas/MG; % used in dp/dt model

% DIFFERENTIAL EQUATIONS

%dhW/dt
dydt(1) = (qWin- qWout + sum(VWoO_vector)/time_oil- sum(
    VOoW_vector)/time_water)/(2*L*sqrt(2*radius*y(1) - y(1)^2));

%dhL/dt
```

```
dydt(2) = (qLin - qOout-qWout)/(2*L*sqrt(2*radius*y(2) - y(2)^2));

%dp/dt
dydt(3) = 1e-5* 1/(pi*radius^2*L-radius^2/2*(2*acos((radius-y(2))/
    radius)-sin(2*acos((radius-y(2))/radius)))*L) * (constant*(
    qGin-qGout) + 1e5*y(3)*(qLin-qOout-qWout));
```

E.3 oildroplets_in_water.m

```
function [noil_new, Voil_new, lpp_oil_new] = oildroplets_in_water(
    v_horizontal, h_water, n, L)

 % v_horizontal is the horizontal velocity of oil droplets =
     horizontal
 % velocity of the water phase

 % h_water is the height of the water.
 % n is the vector of initial oil particles size distribution.
 % L is the length of the Gravity Separator
 % d is the vector of diameter of all the oil particles.
 % s is the vertical velocity of the droplets.
 % V is the volume of oil droplets

 d = zeros(1,10)';
 V = zeros(1,10)';
 s = zeros(1,10)';
 lpp_oil_new = zeros(1,10)';
 noil_new = zeros(1,10)';
 Voil_new = zeros(1,10)';
 density_oil = 831.5;
 density_water = 1030;
 dyn_visc_water = 0.5e-3;
 for i = 1:10
 d(i) = 50*10^(-6)*i;
V(i) = 4/3*pi*(d(i)/2)^3;
 s(i) = abs(((d(i))^2)*9.81*(density_oil-density_water)/(18*
    dyn_visc_water));

 if (L/v_horizontal)>= ( h_water/s(i))

 noil_new(i) = 0;           % number of oil droplets present in the
     water phase
 Voil_new(i) = n(i)* V(i); % volume of oil droplets leaving the
    water phase
 lpp_oil_new(i)= h_water;  % position of the oil particles
else
 lpp_oil_new(i)= (L/v_horizontal)*s(i);
 noil_new(i) = (1-(lpp_oil_new(i)/h_water))* n(i);
Voil_new(i) = n(i) * V(i)*(lpp_oil_new(i)/h_water);
 end
end
```

E.4 waterdroplets_in_oil.m

```
function [nwater_new, Vwater_new,lpp_water_new] =
    waterdroplets_in_oil(v_horizontal, h_liquid, h_water, n, L)

% v_horizontal is the horizontal velocity of water droplets =
    horizontal
% velocity of the bulk oil phase

% h_water is the height of the water.
% n is the vector of initial water particles size distribution.
% L is the length of the Gravity Separator
% d is the vector of diameter of all the water particles.
% s is the vertical velocity of the droplets.
% V is the volume of water droplets

h_oil = h_liquid - h_water;
d = zeros(1,10)';
V = zeros(1,10)';
s = zeros(1,10)';
lpp_water_new = zeros(1,10)';
nwater_new = zeros(1,10)';
Vwater_new = zeros(1,10)';
density_oil = 831.5;
density_water = 1030;
dyn_visc_oil =1e-3;

for i = 1:10
d(i) = 50*10^(-6)*i;
V(i) = 4/3*pi*(d(i)/2)^3;
s(i) = abs(((d(i))^2)*9.81*(density_oil-density_water)/(18*
    dyn_visc_oil));

    if (L/v_horizontal)>= (h_oil/s(i))
        nwater_new(i) = 0;          % number of water droplets
            present
                                    % in the oil phase
        Vwater_new(i) = n(i)* V(i); % volume of water droplets
            leaving
                                    % the oil phase
        lpp_water_new(i)= h_water;  % position of the water
            particles
    else
        lpp_water_new(i)= (L/v_horizontal)*s(i);
        nwater_new(i) = (1-(lpp_water_new(i)/h_oil))* n(i);
        Vwater_new(i) = n(i) * V(i)*(lpp_water_new(i)/h_oil);
    end
end
```

E.5 NMPC.m

```
clear all

addpath('F:\NTNU')
import casadi.*
% Running using only oil droplets in water algebraic equations,
    here
% optimal solution is found

% TIME HORIZON
T = 20;

% DECLARE VARIABLES
qLin= 0.59  ;                % Inlet liquid flow rate
alpha = 0.135;               % Water cut
phi_wo = 0.3;                % Fraction of incoming water entering
    the oil phase
phi_ow = 0.3;                % Fraction of incoming oil entering the
    water phase
radius = 1.65;               % Radius of the gravity separator
L = 10 ;                     % Length of gravity separator
R = 8.314;                   % Universal gas constant
Temp = 328.5  ;              % Temperature
density_gas = 49.7;          % Density of gas
MG = 0.01604;                % Molecular weight of gas
qGin = 0.456;                % Inlet gas flowrate
```

```

phi_ww = 1-phi_wo ;        % fraction of water entering the water
    phase
phi_oo = 1-phi_ow ;        % fraction of oil entering the oil phase
V_sep = 3.14*radius*radius*L; % Total separator volume

density_oil = 831.5;
density_water = 1030;
dyn_visc_water = 0.5e-3;
dyn_visc_oil =1e-3;

% INITIAL VALUES FOR THE INTEGRATORS OF THE DYNAMIC STATES
hWinit = 1;                % Initial value for the water level [m]
hLinit = 2.3;              % Initial value for the liquid level [m]
pinit = 68.5;              % Initial pressure [bar]

% PARTICLE DIAMETERS
d1 = 50*10^(-6);
d2 = 100*10^(-6);
d3 = 150*10^(-6);
d4 = 200*10^(-6);
d5 = 250*10^(-6);
d6 = 300*10^(-6);
d7 = 350*10^(-6);
d8 = 400*10^(-6);
d9 = 450*10^(-6);
d10= 500*10^(-6);

% VOLUME OF EACH PARTICLE CLASS
V1 = 4/3*pi*(d1/2)^3;
V2 = 4/3*pi*(d2/2)^3;
V3 = 4/3*pi*(d3/2)^3;
V4 = 4/3*pi*(d4/2)^3;
```

```
V5 = 4/3*pi*(d5/2)^3;
V6 = 4/3*pi*(d6/2)^3;
V7 = 4/3*pi*(d7/2)^3;
V8 = 4/3*pi*(d8/2)^3;
V9 = 4/3*pi*(d9/2)^3;
V10 = 4/3*pi*(d10/2)^3;

V = [V1 V2 V3 V4 V5 V6 V7 V8 V9 V10];

% TOTAL PARTICLE DISTRIBUTION
n = [1e8 5e8 1e9 5e9 1e10 1e10 5e9 1e9 5e8 1e8];

% TOTAL INITIAL VOLUE DISTRIBUTION OF ALL PARTICLES
init_V = sum(n .* V);

% SETTLING VELOCITY OF OIL PARTICLE CLASS IN THE WATER PHASE
so_1 = abs(((d1)^2)*9.81*(density_oil-density_water)/(18*
    dyn_visc_water));
so_2 = abs(((d2)^2)*9.81*(density_oil-density_water)/(18*
    dyn_visc_water));
so_3 = abs(((d3)^2)*9.81*(density_oil-density_water)/(18*
    dyn_visc_water));
so_4 = abs(((d4)^2)*9.81*(density_oil-density_water)/(18*
    dyn_visc_water));
so_5 = abs(((d5)^2)*9.81*(density_oil-density_water)/(18*
    dyn_visc_water));
so_6=  abs(((d6)^2)*9.81*(density_oil-density_water)/(18*
    dyn_visc_water));
so_7 = abs(((d7)^2)*9.81*(density_oil-density_water)/(18*
    dyn_visc_water));
so_8 = abs(((d8)^2)*9.81*(density_oil-density_water)/(18*
    dyn_visc_water));
```

```
so_9 = abs(((d9)^2)*9.81*(density_oil-density_water)/(18*
    dyn_visc_water));
so_10 = abs(((d10)^2)*9.81*(density_oil-density_water)/(18*
    dyn_visc_water));

% DECLARE MODEL VARIABLES
% State variables
x1 = MX.sym('x1');          % Water level
x2 = MX.sym('x2');          % Liquid level
x3 = MX.sym('x3');          % System Pressure

% Control variables
u1 = MX.sym('u1');          % Outlet Water flowrate
u2 = MX.sym('u2');          % Outlet Oil flowrate
u3 = MX.sym('u3');          % Outlet gas flowrate

%Algebraic variables (particle position of oil in the water phase)
lpp1 = MX.sym('lpp1');
lpp2 = MX.sym('lpp2');
lpp3 = MX.sym('lpp3');
lpp4 = MX.sym('lpp4');
lpp5 = MX.sym('lpp5');
lpp6 = MX.sym('lpp6');
lpp7 = MX.sym('lpp7');
lpp8 = MX.sym('lpp8');
lpp9 = MX.sym('lpp9');
lpp10 = MX.sym('lpp10');

% CONCATENATE VARIABLES
x = [x1;x2;x3];             % state variables
u = [u1; u2; u3];           % control variables
lpp = [lpp1; lpp2; lpp3; lpp4; lpp5; lpp6; lpp7; lpp8; lpp9; lpp10
     ]; % particle position alg. variables
```

```

% DECLARE VARIABLES
flow_fraction_w =  (alpha*(1-phi_wo)) + ((1-alpha)*phi_ow);
AW = (radius^2*acos((radius-x1)/radius)) - ((radius - x1)*sqrt(2*
    radius*x1 - x1*x1));
vel_water = (flow_fraction_w*qLin)/AW;
time_water = L/ vel_water ;

A_L = (radius^2*acos((radius-x2)/radius)) - ((radius - x2)*sqrt(2*
    radius*x2 - x2*x2));
AO = A_L-AW;
flow_fraction_o = ((1-alpha)*(1-phi_ow)) + (alpha*phi_wo);
vel_oil = (flow_fraction_o*qLin)/AO;
time_oil = L/vel_oil;

% DEFINE ARGUMENT FUNCTION(residence time difference of oil
    droplets)
arg1 = (L/vel_water)-(x1/so_1);
arg2 = (L/vel_water)-(x1/so_2);
arg3 = (L/vel_water)-(x1/so_3);
arg4 = (L/vel_water)-(x1/so_4);
arg5 = (L/vel_water)-(x1/so_5);
arg6 = (L/vel_water)-(x1/so_6);
arg7 = (L/vel_water)-(x1/so_7);
arg8 = (L/vel_water)-(x1/so_8);
arg9 = (L/vel_water)-(x1/so_9);
arg10 = (L/vel_water)-(x1/so_10);

% USE ARCTAN FUNCTION TO EXPRESS CONTINUOUS SYSTEM
arctan_func1 = (atan((arg1).*1000*pi)+pi/2)./pi;
arctan_func2 = (atan((arg2).*1000*pi)+pi/2)./pi;
arctan_func3 = (atan((arg3).*1000*pi)+pi/2)./pi;
arctan_func4 = (atan((arg4).*1000*pi)+pi/2)./pi;
```

```
arctan_func5 = (atan((arg5).*1000*pi)+pi/2)./pi;
arctan_func6 = (atan((arg6).*1000*pi)+pi/2)./pi;
arctan_func7 = (atan((arg7).*1000*pi)+pi/2)./pi;
arctan_func8 = (atan((arg8).*1000*pi)+pi/2)./pi;
arctan_func9 = (atan((arg9).*1000*pi)+pi/2)./pi;
arctan_func10 = (atan((arg10).*1000*pi)+pi/2)./pi;

% DEFINING ALGEBRAIC EQUATIONS
f_z1 = lpp1 - ((arctan_func1)*x1 + (1-arctan_func1)*(L*so_1./
    vel_water));
f_z2 = lpp2 - ((arctan_func2)*x1 + (1-arctan_func2)*(L*so_2./
    vel_water));
f_z3 = lpp3 - ((arctan_func3)*x1 + (1-arctan_func3)*(L*so_3./
    vel_water));
f_z4 = lpp4 - ((arctan_func4)*x1 + (1-arctan_func4)*(L*so_4./
    vel_water));
f_z5 = lpp5 - ((arctan_func5)*x1 + (1-arctan_func5)*(L*so_5./
    vel_water));
f_z6 = lpp6 - ((arctan_func6)*x1 + (1-arctan_func6)*(L*so_6./
    vel_water));
f_z7 = lpp7 - ((arctan_func7)*x1 + (1-arctan_func7)*(L*so_7./
    vel_water));
f_z8 = lpp8 - ((arctan_func8)*x1 + (1-arctan_func8)*(L*so_8./
    vel_water));
f_z9 = lpp9 - ((arctan_func9)*x1 + (1-arctan_func9)*(L*so_9./
    vel_water));
f_z10 = lpp10 - ((arctan_func10)*x1 + (1-arctan_func10)*(L*so_10./
    vel_water));

% VOLUME OF OIL DROPLETS ENTERING THEIR NATIVE PHASE
VOoW1 = (lpp1./x1)*n(1)* V1;
VOoW2 = (lpp2./x1)*n(2)* V2;
VOoW3 = (lpp3./x1)*n(3)* V3;
```

```
VOoW4 = (lpp4./x1)*n(4)* V4;
VOoW5 = (lpp5./x1)*n(5)* V5;
VOoW6 = (lpp6./x1)*n(6)* V6;
VOoW7 = (lpp7./x1)*n(7)* V7;
VOoW8 = (lpp8./x1)*n(8)* V8;
VOoW9 = (lpp9./x1)*n(9)* V9;
VOoW10 =(lpp10./x1)*n(10)* V10;

VOoW = (VOoW1+ VOoW2 + VOoW3 + VOoW4 + VOoW5 + VOoW6 + VOoW7 +
    VOoW8 +  VOoW9 + VOoW10);

% MPC DIFFERENTIAL EQUATIONS

 xdot1 = (qLin*(alpha*phi_ww+(1-alpha)*phi_ow)-u1 -(VOoW/time_water
    ))/(2*L*sqrt(2*radius*x1 - x1^2));
 xdot2 = (qLin - u2 -u1)/(2*L*sqrt(2*radius*x2 - x2^2));
    VG = V_sep - A_L*L;
 xdot3 = (1/VG)*(((((R*Temp*density_gas)/(MG))*(qGin - u3))+(x3
    *10^5)*(qLin-u1-u2)))*10^-5;

% CONCATENATE EQUATIONS TO FORM SEMI EXPLICIT DAE SYSTEM
 diff = [xdot1; xdot2; xdot3];   % Differential eqns
 alg = [f_z1; f_z2; f_z3; f_z4; f_z5; f_z6; f_z7; f_z8; f_z9; f_z10
    ];% Particle position alg expression

% MPC OBJECTIVE TERM

L1 = (x1 - 1.2)^2 + (x2 - 2.5 )^2 + (x3 - 68.7)^2; % CASE 1
%L1 = 100* (x2 - 2.5)^2 + 100*(x3 - 68.7)^2 - 1000*(VOoW)^2;   %
    CASE 3

% CONTINUOUS TIME DYNAMICS
```

```
f = Function('f',{x,lpp,u},{diff,alg,L1},{'x','z','p'},{'xdot','zj
    ','qj'});

%% ———DIRECT COLLOCATION SCHEME———%%

% DEGREE OF INTERPOLATING POLYNOMIAL
d = 3;

% GET COLLOCATION POINTS
tau_root = [0 collocation_points(d, 'radau')];

% COEFFICIENTS OF THE COLLOCATION EQUATION
C = zeros(d+1,d+1);

% COEFFICIENTS OF THE CONTINUITY EQUATION
D = zeros(d+1, 1);

% COEFFICIENTS OF THE QUADRATURE FUNCTION
B = zeros(d+1, 1);

% CONSTRUCT POLYNOMIAL BASIS
for j=1:d+1
  % CONSTRUCT LAGRANGE POLYNOMIALS TO GET THE POLYNOMIAL BASIS AT
      THE COLLOCATION POINT
  coeff = 1;
  for r=1:d+1
    if r ~= j
      coeff = conv(coeff, [1, -tau_root(r)]);
      coeff = coeff / (tau_root(j)-tau_root(r));
    end
  end

  % EVALUATE POLYNOMIAL AT THE FINAL TIME TO GET THE COEFFICIENTS
      OF CONTINUITY EQUATION
  D(j) = polyval(coeff, 1.0);

  % EVALUATE TIME DERIVATIVE OF THE POLYNOMIAL AT ALL COLLOCATION
      POINTS TO GET COEFFICIENTS OF CONTINUITY EQUATION
  pder = polyder(coeff);
  for r=1:d+1
    C(j,r) = polyval(pder, tau_root(r));
  end

  % EVALUATE INTEGRAL OF THE POLYNOMIAL TO GET COEFFICIENTS OF THE
      QUADRATURE FUNCTION
  pint = polyint(coeff);
  B(j) = polyval(pint, 1.0);
end

% CONTROL DISCRETIZATION
N = 10;                         % Number of control intervals
h = T/N;                        % Size of each control interval

% START WITH AN EMPTY NLP
w ={};
w0 = [];
lbw = [];
ubw = [];
J = 0;
g={};
lbg = [];
ubg = [];

% "LIFT" INITIAL CONDITIONS
X0 = MX.sym('X0', 3);
```

```
Z0 = MX.sym('ZO', 10);
w = {w{:}, X0, Z0};
lbw = [lbw; 1; 2.3; 68.5; zeros(10,1)]; % Assign initial
    conditions to lower bound
ubw = [ubw; 1; 2.3; 68.5; zeros(10,1)]; % Assign initial
    conditions to upper bound
w0 =  [w0;  1; 2.3;  68.5;  zeros(10,1)];% Initial conditions
    value


%% FORMULATE THE NLP (DIRECT COLLOCATION)
Xk = X0;
for k = 0: N-1
    % NEW NLP VARIABLE FOR EACH CONTROL
    Uk = MX.sym(['U_' num2str(k)],3);
    w = {w{:}, Uk};
    lbw = [lbw; 0;0;0];
    ubw = [ubw; 2 ; 2; 5];
    w0 =  [w0; 0; 0; 0];

    % STATE AT COLLOCATION POINTS
    Xkj = {}; Zkj = {};
    for j=1:d
        Xkj{j} = MX.sym(['X_' num2str(k) '_' num2str(j)], 3);
        Zkj{j} = MX.sym(['Z_' num2str(k) '_' num2str(j)], 10);
        w = {w{:}, Xkj{j},Zkj{j}};
        lbw = [lbw; 0.9; 1.9; 50; zeros(10,1)];
        ubw = [ubw; 2.2; 2*radius; 100; 2*ones(10,1)];
        w0 = [w0; 1; 2.3; 68.5; zeros(10,1)];
    end

    % LOOP OVER COLLOCATION POINTS
    Xk_end = D(1)*Xk; %where D is the continuity eqn coeff
    for j=1:d
      % EXPRESSION FOR STATE DERIVATIVE AT COLLOCATION POINT(
          polynomial approx)
      xp = C(1,j+1)*Xk; % C is the derivative at the collocation
          points
      for r=1:d
          xp = xp + C(r+1,j+1)*Xkj{r};
      end
      % APPEND COLLOCATION EQUATIONS
      [fj,zj,qj] = f(Xkj{j},Zkj{j},Uk);        % fj = dxdt & qj =
          objective function value, zj = solution of alg
          expression
      g = {g{:}, h*fj - xp, zj};               % slope constraints
      lbg = [lbg; zeros(3,1); zeros(10,1)];
      ubg = [ubg; zeros(3,1); zeros(10,1)];

      % ADD CONTRIBUTION TO THE END STATE
      Xk_end = Xk_end + D(j+1)*Xkj{j};

      % ADD CONTRIBUTION TO THE QUADRATURE FUNCTION
      J = J + B(j+1)*qj*h;

    end
    % NEW NLP VARIABLE FOR STATE AT THE END OF THE EQUATION
    Xk = MX.sym(['X_' num2str(k+1)], 3); % this value is updated
    w = {w{:}, Xk};
    lbw = [lbw; 0.9 ; 1.9; 50];
    ubw = [ubw; 2.2; 2*radius; 100];
    w0 =  [w0 ; 1; 2.3; 68.5];

    % ADD EQUALITY CONSTRAINT
    g = {g{:}, Xk_end-Xk};          % shooting gap constraint
    lbg = [lbg; 0; 0 ;0];
    ubg = [ubg; 0; 0 ;0];

    % ADD CONTROL MOVES AND PENALTY TERM IN THE OBJECTIVE FUNCTION
    if k>0
    % ADD EQUALITY CONSTRAINT
    g = {g{:}, Uk - Uk_prev};
    lbg = [lbg; -0.05; -0.05 ; -0.05];
    ubg = [ubg; 0.05; 0.05 ; 0.05];
    % INCLUDE THE PENALTY TERM IN OBJECTIVE FUNCTION WITH WEIGHT
        FACTOR
     J = J + ((Uk(1) - Uk_prev(1)).^2) + ((Uk(2) - Uk_prev(2)).^2)
         + ((Uk(3) - Uk_prev(3)).^2);
    end

     Uk_prev = Uk;
end
%%

% CREATE AN NLP SOLVER
prob = struct('f', J, 'x', vertcat(w{:}), 'g', vertcat(g{:}));
options.ipopt.print_level = 0; % options does not allow to print
    at every iteration
options.print_time = false;
solver = nlpsol('solver', 'ipopt', prob, options);

% TOTAL SIMULATION TIME
tsim =300;

% DEFINE EMPTY MATRICES
qWout_vec = zeros(1,tsim);
qOout_vec = zeros(1,tsim);
qGout_vec = zeros(1,tsim);

```

```
position_oilparicles_vec = zeros(tsim,10);
volume_oilparticles_vec = zeros(tsim,10);

% PASSING INITIAL VALUES TO PLANT MODEL
hW = hWinit;
hL = hLinit;
p = pinit;

for sim = 1: tsim
    % SOLVE THE NLP
    sol = solver('x0', w0, 'lbx', lbw, 'ubx', ubw,'lbg', lbg, 'ubg
        ', ubg);
    w_opt = full(sol.x);
    sim                         % helps to know the current
        iteration

    % SOLUTION
    x1_opt = w_opt(56:6+13*d:end);
    x2_opt = w_opt(57:6+13*d:end);
    x3_opt = w_opt(58:6+13*d:end);

    u1_opt = w_opt(14:6+13*d:end);
    u2_opt = w_opt(15:6+13*d:end);
    u3_opt = w_opt(16:6+13*d:end);

    % FIRST VALUE OF THE OPTIMIZER SENT TO PLANT MODEL
    qWout= u1_opt(1);
    qOout= u2_opt(1);
    qGout= u3_opt(1);

    % SET EMPTY MATRICES TO STORE THE OPTIMUM VALUES
    qWout_vec(sim) = qWout;
    qOout_vec(sim) = qOout;
    qGout_vec(sim) = qGout;

    % CONCATENATE THE OPTIMUM INPUTS
    control_inputs = [qWout;qOout;qGout];
    control_inputs_all(:,sim) = [control_inputs];

    % SIMULATE PLANT MODEL
    parameters_plant = [qLin, alpha, phi_wo, phi_ow, radius, L, R,
        Temp, density_gas, MG, qGin, qWout, qOout, qGout, hW, hL,
        p];
    [t,y]=ode15s(@gravity_ode,[sim-1 sim],[hW, hL, p],[],
        parameters_plant);

    % EXTRACT EACH FINAL ELEMENT OF THE STATES
    hW = y(end,1);
    hL = y(end,2);
    p  = y(end,3);

    % ADDING NOISE TO THE STATES
    hW =  hW +  1e-3*randn(1);
    hL =  hL +  1e-3*randn(1);
    p  =  p  +  1e-2*randn(1);

    [vel_oil,vel_water,qWin] = velocity_inflows(qLin,hL,hW,alpha,
        phi_wo,phi_ow,radius);
    [~, VOoW_vector, lpp_oil_new]  = oildroplets_in_water(
        vel_water, hW, L);

    position_oilparicles_vec(sim,:) = [lpp_oil_new]';
    volume_oilparticles_vec(sim,:) = [VOoW_vector]';

    % STATES PERTURBED BY NOISE
    states = [hW; hL; p];
    states_all(:,sim) = [states];

%% SET NEW INITIAL VALUES FOR THE NEXT ITERATION
lbw = [];
ubw = [];
w0 = [];

% "LIFT" INITIAL CONDITIONS
lbw = [lbw; states; lpp_oil_new'];
ubw = [ubw; states; lpp_oil_new'];
w0 =  [w0;  states; lpp_oil_new'];

    for k= 1: N
        lbw = [lbw; 0; 0; 0];
        ubw = [ubw; 2 ; 2; 5];
        w0 =  [w0; control_inputs];

        % STATE AT COLLOCATION POINTS
        for j=1:d
            lbw = [lbw; 0.9; 1.9; 50; zeros(10,1)];
            ubw = [ubw; 2.2; 2*radius; 100; 2*ones(10,1)];
            w0 =  [w0; states; lpp_oil_new'];
        end

        % NEW NLP VARIABLE FOR STATES AT THE END OF THE INTERVAL
        lbw = [lbw; 0.9 ; 1.9; 50];
        ubw = [ubw; 2.2 ; 2*radius; 100];
        w0 =  [w0 ; states];

    end

end
```

```
tgrid = 0:1:tsim;
level_W1  = [1, states_all(1,:)];
level_L1  = [2.3, states_all(2,:)];
P1 = [68.5, states_all(3,:)];

qWout1 = [control_inputs_all(1,:), nan];
qOout1 = [control_inputs_all(2,:), nan];
qGout1 = [control_inputs_all(3,:), nan];

% PLOTTING
figure;
subplot(321)
plot(tgrid, level_W1)
subplot(322)
stairs(tgrid, qWout1)
subplot(323)
plot(tgrid, level_L1)
subplot(324)
stairs(tgrid, qOout1)
subplot(325)
plot(tgrid, P1)
subplot(326)
stairs(tgrid, qGout1)
```