Dynamic Optimisation of Small-size Wastewater Treatment Plants

M. Fikar

Final report of the program:
Accueil de chercheur étranger de haut niveau
Ministère de la recherche
July – August 2001

Contents

Chapter 1

Introduction

In this report we make a summary of the research activities in the framework of the program "Accueil de chercheur étranger de haut niveau" of the "Ministère de la recherche". The research subject was dynamic optimisation of small-size wastewater treatment plants. This subject has been chosen for its importance arising from both regulation fulfillment and cost aspects of the plant operation.

During the last decade, new stricter EU directives came out for nitrogen removal in wastewater treatment. Before, only an effluent standard of 20 mg/l of nitrogen in the summer period was imposed. The new EU directive is yearly averaged effluent N-total of 10mg/l for larger plants over 20,000 p.e. (population equivalents) and 15 mg/l for smaller plants.

A widely used system for biological wastewater treatment is the activated sludge process (ASP). Removal of N requires two biological processes: nitrification and denitrification. Nitrification takes place under aerobic conditions, whereas denitrification requires anoxic environment. For small-size plants, usually a single basin is used where oxygen is supplied by surface turbines in the aeration periods and non-aeration is realised by simply switching the aeration off.

The principal energy consumption (and subsequently total operation costs) is caused by aeration. Therefore, oxygen control is of great importance and lowering the total period of aeration reduces the operation costs significantly. On the other hand, it has been shown, that the oxygen control is quite flexible in influencing the N-removal (Hao and Huang, 1996).

The main challenge for control of the ASP is disturbances rejection – to avoid excessive aeration and to maximise the conversion rates of the biological processes. The main source of disturbances is the influent. Its characteristics are large diurnal variations both in the influent flow rate and composition (result of the characteristic life patterns of households). Even if the large reactor volume dampens this diurnal cycle just by dilution, there remains a significant task for active control. The other principal sources of disturbances are rain/storm events that may cause serious overloading incidents or winter time when growth rate of the biomass is severely inhibited.

The aim of the work is to determine an optimal duration of the aeration and non-aeration sequence which will minimise the operational costs as well as satisfy the constraints specified by the EU directives.

In the second chapter, a typical model of a small-size wastewater treatment plant is described. The model consists of a single aeration basin where oxygen is supplied by surface turbines. The initial model has been described as a hybrid discrete/continuous system. Such systems are traditionally more difficult to optimise and control and therefore one of the aims was to reformulate the model so that it is described as a continuous system only.

In the third chapter, the dynamic optimisation problem is formulated. Basically, as the aim is to reduce the energy consumption, mathematical definition of this aim is presented and possible difficulties in the solution are discussed.

The Chapter 4 describes in more detail the implementation issues of the optimisation, transformation of the original dynamic optimisation to a static optimisation as well as an approach of gradient calculations needed in the nonlinear programming problem.

Chapter 5 presents simulation results obtained and Chapter 6 draws conclusions and gives future perspectives.

A user manual to the dynamic optimisation package `DYNO` is given in the Appendix, together with some examples.

Nomenclature

Roman symbols

$\mathcal{A}_O$	=	oxygen transfer rate from the turbines $\left(\text{mg.L}^{-1}\text{.day}^{-1}\right)$
BOD_5	=	biochemical oxygen demand $\left(\text{mg.L}^{-1}\right)$
COD	=	chemical oxygen demand $\left(\text{mg.L}^{-1}\right)$
$\mathcal{E}$	=	energy consumption (J)
H_i	=	Hamiltonian of the i-th constraint $(-)$
J_0	=	performance index $(-)$
J_i	=	i-th constraint (cost if $i = 0$) $(-)$
k_La	=	oxygen transfer coefficient $\left(\text{h}^{-1}\right)$
N_c	=	number of cycles $(-)$
N_d	=	number of days $(-)$
$\mathcal{P}$	=	nominal power of the turbines (kWh)
$\boldsymbol{p}$	=	parameter vector $(-)$
p.e.	=	population-equivalent
Q	=	flowrate $\left(\text{m}^3\text{.day}^{-1}\right)$
S_I	=	soluble inert COD concentration $\left(\text{mg.L}^{-1}\right)$
S_{ND}	=	soluble organic nitrogen concentration $\left(\text{mg.L}^{-1}\right)$
S_{NH}	=	ammonium concentration $\left(\text{mg.L}^{-1}\right)$
S_{NO}	=	nitrate and nitrite nitrogen concentration $\left(\text{mg.L}^{-1}\right)$
S_O	=	dissolved oxygen concentration $\left(\text{mg.L}^{-1}\right)$
S_O^{sat}	=	saturated dissolved oxygen concentration $\left(\text{mg.L}^{-1}\right)$
S_S	=	readily biodegradable COD concentration $\left(\text{mg.L}^{-1}\right)$
SS	=	suspended solids $\left(\text{mg.L}^{-1}\right)$
TN	=	total nitrogen $\left(\text{mg.L}^{-1}\right)$
Δt_j	=	aeration/non-aeration time (s)
$t_{\text{min}}^{\text{off}}$	=	minimum length of air-off periods (s)
$t_{\text{max}}^{\text{off}}$	=	maximum length of air-off periods (s)
$t_{\text{min}}^{\text{on}}$	=	minimum length of air-on periods (s)
$t_{\text{max}}^{\text{on}}$	=	maximum length of air-on periods (s)
V^{rb}	=	bioreactor volume $\left(\text{m}^3\right)$
$X_{B,H}$	=	heterotrophic biomass concentration $\left(\text{mg.L}^{-1}\right)$
$X_{B,A}$	=	autotrophic biomass concentration $\left(\text{mg.L}^{-1}\right)$
X_I	=	particulate inert COD concentration $\left(\text{mg.L}^{-1}\right)$
X_{ND}	=	particulate organic nitrogen concentration $\left(\text{mg.L}^{-1}\right)$
X_S	=	slowly biodegradable COD concentration $\left(\text{mg.L}^{-1}\right)$
$\boldsymbol{u}$	=	control vector $(-)$
$\boldsymbol{x}$	=	state vector $(-)$
$\boldsymbol{y}$	=	vector of optimised variables $(-)$

Greek letters

$\boldsymbol{\lambda}$	=	vector of adjoint variables $(-)$
τ_{COD}	=	influent COD concentration variation $(-)$

τ_Q	=	influent flowrate variation $(-)$

Subscripts and superscripts

^{in}	=	influent
^{max}	=	maximum value
^{rs}	=	recycled sludge
^{w}	=	excess sludge
$\bar{\ }$	=	mean value

Chapter 2

Wastewater Treatment Plant Model

At first, we describe the original model used at the start of the work (Chachuat et al., 2001). The original model can be characterised as a hybrid discrete/continuous dynamic system. As such systems are known to be difficult to optimise, the model has been modified so that it is described as a standard continuous-time process characterised by a set of ordinary differential equations.

2.1 Process

A real small-size treatment facility is considered (15, 000 p.e.). The process consists of a unique aeration tank ($V^{rb} = 2,050\,\text{m}^3$) equipped with mechanical surface aerators (turbines) which provide oxygen ($\mathcal{P} = 30\,\text{kW}$, $k_La = 4.5\,\text{h}^{-1}$) and mix the incoming wastewater with biomass (Fig. 2.1). The settler is a cylindrical tank where the solids are either recirculated to the aeration tank ($Q^{rs} = 7,600\,\text{m}^3/\text{day}$) or extracted from the system ($Q^w = 75\,\text{m}^3/\text{day}$).

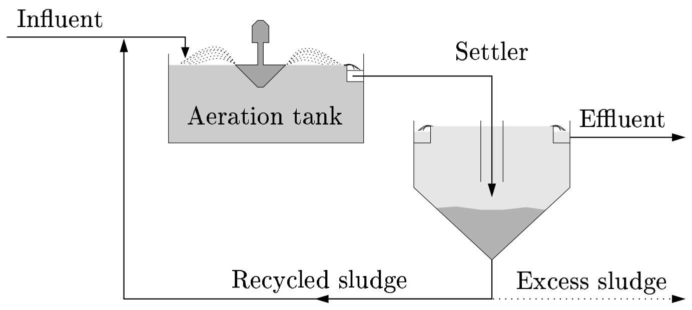

Figure 2.1: Typical small-size activated sludge treatment plant.

The influent average flow $\left(\overline{Q^{in}}\right)$ is about $3,050\,\text{m}^3/\text{day}$ and average organic $\left(\overline{COD^{in}}\right)$ and nitrogen $\left(\overline{TN^{in}}\right)$ loads are $343\,\text{mg/L}$ and $33\,\text{mg/L}$, respectively (after primary treatment). The daily variations of dry weather conditions are based on measured data from the plant. It is accounted for by defining weighting functions for both influent flowrate and organic load variations, $\tau_Q(t)$ and $\tau_{COD}(t)$, (Fig. 2.2):

- influent flowrate variation:

$$Q^{in}(t) = \tau_Q(t)\,\overline{Q^{in}} \tag{2.1}$$

$$\text{where} \quad \tau_Q(t) = \sum_{k=1}^{3}\left[a_k \cos(2k\pi t) + b_k \sin(2k\pi t)\right]$$

$$\text{with} \quad \begin{cases} a_1 = -0.32 & ; \quad b_1 = -0.18 \\ a_2 = 0.23 & ; \quad b_2 = -0.01 \\ a_3 = -0.06 & ; \quad b_3 = -0.01 \end{cases}$$

- organic load variation:

$$COD^{in}(t) = \tau_{COD}(t)\,\overline{COD^{in}}\ ,\ \forall t \tag{2.2}$$

$$\text{where} \quad \tau_{COD}(t) = \sum_{k=1}^{3}\left[c_k \cos(2k\pi t) + d_k \sin(2k\pi t)\right]$$

$$\text{with} \quad \begin{cases} c_1 = 0.24 & ; \quad d_1 = -0.20 \\ c_2 = -0.09 & ; \quad d_2 = 0.07 \\ c_3 = 0.04 & ; \quad d_3 = -0.02 \end{cases}$$

- total nitrogen load variation: since the concentration of total nitrogen in the influent exhibits small variations, it is assumed constant. Thus:

$$TN^{in}(t) = \overline{TN^{in}}\ ,\ \forall t$$

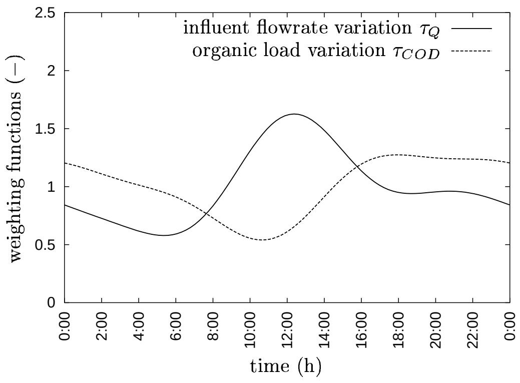

Figure 2.2: Influent flowrate and organic load variations.

The average wastewater composition is shown in Table 2.1. The fractions f are related to the state variables presented in Table 2.2 and defined as the ratio between the corresponding concentration and COD^{in} or TN^{in}.

2.2 Original Model

The application of dynamic optimisation methods requires the knowledge of a sufficiently accurate mathematical model. Among both scientists and practitioners a reasonable consensus exists upon the statement that the Activated Sludge Model No.1 (ASM 1) by Henze et al. (1987) is the most

Table 2.1: Average inlet composition

COD^{in} fractions	
f_{SI}	5%
f_{SS}	35%
f_{XI}	10%
f_{XS}	35%
f_{XBH}	15%
f_{XBA}	0%

TN^{in} fractions	
f_{SNH}	66%
f_{SNO}	0%
f_{SND}	2%
f_{XND}	32%

popular mathematical description of the biochemical processes in the reactors for N-removal. Two restrictions are however brought to the original ASM 1 model: (i) the state variable describing total alkalinity is not included, and (ii) inert particulate material from influent and from biomass decay are combined into a single variable (X_I) since they are of minor interest. The resulting biodegradation model consists of 11 state variables and 20 parameters, respectively described in Table 2.2 and Table 2.3. The kinetic and stoichiometric parameter values considered are those defined for the *simulation benchmark* (Alex et al., 1999). They can be found, for instance, on the European COST action 624 website (`http://www.ensic.u-nancy.fr/COSTWWTP`). In addition, the limitations induced on the heterotrophic biomass growth process by low concentration of ammonia in the reactor are taken into account by adding the factor $\frac{S_{NH}}{K_{NH,H}+S_{NH}}$ in the corresponding kinetic rate equation (2.4). This modification requires an additional stoichiometric parameter $K_{NH,H}$, its value is set to $0.05\,\text{g.m}^{-3}$ as suggested by Julien (1997) (Table 2.3).

The degradation rates from the ASM 1 model are incorporated into the mass balance equation for each component, assuming perfect mixing in the reactor:

$$\frac{dx_i^{br}}{dt} = \frac{Q^{in}x_i^{in} + Q^{rs}x_i^{rs} - \left(Q^{in}+Q^{rs}\right)x_i^{br}}{V^{br}} + r_i \quad , \quad i = 1, 2, \ldots, 11 \tag{2.3}$$

where $\boldsymbol{x}^{in}$, $\boldsymbol{x}^{br}$ and $\boldsymbol{x}^{rs}$ are 11-dimensional vectors related to the concentration in the influent, in the reactor and in the recycled sludge respectively ; their components are (S_I S_S X_I X_S $X_{B,H}$ $X_{B,A}$ S_{NO} S_{NH} S_{ND} X_{ND} S_O)^T. The model initial conditions are given in Table 2.4.

$\boldsymbol{r}$ is an 11-dimensional vector formed by the degradation rates of each component. To derive its form, kinetic rates ρ_j for degradation process $j = 1, \ldots, 8$ are defined as:

$$\boldsymbol{\rho} = \begin{pmatrix} \mu_H \frac{S_S^{br}}{K_S+S_S^{br}} \frac{S_{NH}^{br}}{K_{NH,H}+S_{NH}^{br}} \frac{S_O^{br}}{K_{O,H}+S_O^{br}} X_{B,H}^{br} \\ \mu_H \frac{S_S^{br}}{K_S+S_S^{br}} \frac{S_{NH}^{br}}{K_{NH,H}+S_{NH}^{br}} \frac{K_{O,H}}{K_{O,H}+S_O^{br}} \frac{S_{NO}^{br}}{K_{NO}+S_{NO}^{br}} \eta_{NOg} X_{B,H}^{br} \\ \mu_A \frac{S_{NH}^{br}}{K_{NH,A}+S_{NH}^{br}} \frac{S_O^{br}}{K_{O,A}+S_O^{br}} X_{B,A}^{br} \\ b_H X_{B,H}^{br} \\ b_A X_{B,A}^{br} \\ \kappa_h S_{ND}^{br} X_{B,H}^{br} \\ K_a \frac{X_S^{br}/X_{B,H}^{br}}{K_X+X_S^{br}/X_{B,H}^{br}} \left(\frac{S_O^{br}}{K_{O,H}+S_O^{br}} + \eta_{NO,h} \frac{K_{O,H}}{K_{O,H}+S_O^{br}} \frac{S_{NO}^{br}}{K_{NO}+S_{NO}^{br}} \right) X_{B,H}^{br} \\ K_a \frac{X_{ND}^{br}/X_{B,H}^{br}}{K_X+X_S^{br}/X_{B,H}^{br}} \left(\frac{S_O^{br}}{K_{O,H}+S_O^{br}} + \eta_{NO,h} \frac{K_{O,H}}{K_{O,H}+S_O^{br}} \frac{S_{NO}^{br}}{K_{NO}+S_{NO}^{br}} \right) X_{B,H}^{br} \end{pmatrix} \tag{2.4}$$

Table 2.2: Model state variables

1.	Inert soluble organic matter, S_I $\left[\text{gCOD.m}^{-3}\right]$
2.	Readily biodegradable substrate, S_S $\left[\text{gCOD.m}^{-3}\right]$
3.	Inert particulate organic matter and products, X_I $\left[\text{gCOD.m}^{-3}\right]$
4.	Slowly biodegradable substrate, X_S $\left[\text{gCOD.m}^{-3}\right]$
5.	Active heterotrophic biomass, $X_{B,H}$ $\left[\text{gCOD.m}^{-3}\right]$
6.	Active autotrophic biomass, $X_{B,A}$ $\left[\text{gCOD.m}^{-3}\right]$
7.	Nitrate and nitrite nitrogen, S_{NO} $\left[\text{gN.m}^{-3}\right]$
8.	Ammonium nitrogen, S_{NH} $\left[\text{gN.m}^{-3}\right]$
9.	Soluble biodegradable organic nitrogen, S_{ND} $\left[\text{gN.m}^{-3}\right]$
10.	Particulate biodegradable organic nitrogen, X_{ND} $\left[\text{gN.m}^{-3}\right]$
11.	Dissolved oxygen, S_O $\left[\text{gO}_2\text{.m}^{-3}\right]$

Table 2.3: Model parameters

	Stoichiometric parameters	
Y_H	0.67	Yield for heterotrophic biomass [−]
Y_A	0.24	Yield for autotrophic biomass [−]
fr_{XI}	0.08	Fraction of biomass leading to particulate products [−]
i_{NBM}	0.08	Mass of biomass per mass of COD in biomass $\left[\text{gN.gCOD}^{-1}\right]$
i_{NXI}	0.06	Mass of biomass per mass of COD in products from biomass decay $\left[\text{gN.gCOD}^{-1}\right]$
	Kinetic parameters	
μ_H	4.0	Maximum specific growth rate for heterotrophic biomass $\left[\text{day}^{-1}\right]$
b_H	0.3	Decay rate coefficient for heterotrophic biomass $\left[\text{day}^{-1}\right]$
K_S	10.0	Half-saturation coefficient for heterotrophic biomass $\left[\text{gCOD.m}^{-3}\right]$
$K_{NH,H}$	0.05	Ammonia half-saturation coefficient for heterotrophic biomass $\left[\text{gN.m}^{-3}\right]$
$K_{O,H}$	0.2	Oxygen half-saturation coefficient for heterotrophic biomass $\left[\text{gO}_2\text{.m}^{-3}\right]$
K_{NO}	0.5	Nitrate half-saturation coefficient for denitrifying heterotrophic biomass $\left[\text{gN.m}^{-3}\right]$
μ_A	0.5	Maximum specific growth rate for autotrophic biomass $\left[\text{day}^{-1}\right]$
b_A	0.05	Decay rate coefficient for autotrophic biomass $\left[\text{day}^{-1}\right]$
$K_{NH,A}$	1.0	Ammonia half-saturation coefficient for autotrophic biomass $\left[\text{gN.m}^{-3}\right]$
$K_{O,A}$	0.4	Oxygen half-saturation coefficient for autotrophic biomass $\left[\text{gO}_2\text{.m}^{-3}\right]$
η_{NOg}	0.8	Correction factor for μ_H under anoxic conditions [−]
η_{NOh}	0.8	Correction factor for hydrolysis under anoxic conditions [−]
κ_h	3.0	Maximum specific hydrolysis rate $\left[\text{day}^{-1}\right]$
K_X	0.1	Half-saturation coefficient for hydrolysis of slowly biodegradable substrate [−]
κ_a	0.05	Ammonification rate $\left[\text{m}^3\text{.gCOD}^{-1}\text{.day}^{-1}\right]$

Table 2.4: Initial concentrations in the aeration tank

	S_I	S_S	X_I	X_S	$X_{B,H}$	$X_{B,A}$	S_{NO}	S_{NH}	S_{ND}	X_{ND}	S_O
Conc. (mg/L)	17.98	2.27	2120.15	79.55	2238.65	115.18	0.02	9.70	0.14	6.29	0.00

The apparent reaction rates r_i for components $i = 1, \ldots, 11$ are then given by:

$$\boldsymbol{r} = \begin{pmatrix} 0 \\ -\frac{1}{Y_H}\left(\rho_1 + \rho_2\right) + \rho_7 \\ fr_{XI}\left(\rho_4 + \rho_5\right) \\ \left(1 - fr_{XI}\right)\left(\rho_4 + \rho_5\right) - \rho_7 \\ \rho_1 + \rho_2 - \rho_4 \\ \rho_3 - \rho_5 \\ -\frac{1-Y_H}{2.86\,Y_H}\rho_2 + \frac{1}{Y_A}\rho_3 \\ -i_{NBM}\left(\rho_1 + \rho_2\right) - \left(i_{NBM} + \frac{1}{Y_A}\right)\rho_3 + \rho_6 \\ -\rho_6 + \rho_8 \\ \left(i_{NBM} - fr_{XI}\, i_{NXI}\right)\left(\rho_4 + \rho_5\right) - \rho_8 \\ -\frac{1-Y_H}{Y_H}\rho_1 - \frac{4.57-Y_A}{Y_A}\rho_3 \end{pmatrix} \tag{2.5}$$

The separation of liquid and solid phases which takes place in the settler is assumed to be perfect: (i) the sum of the two outflows (outlet and recycled sludge flowrates) equals the settler influent flowrate and (ii) the effluent is exclusively constituted by soluble components, *i.e.*

- Effluent concentration:

$$S_i^{eff} = S_i^{br} \tag{2.6}$$
$$X_i^{eff} = 0 \tag{2.7}$$

- Recycled sludge concentration:

$$S_i^{rs} = S_i^{br} \tag{2.8}$$
$$X_i^{rs} = \frac{Q^{in} + Q^{rs}}{Q^{rs} + Q^{w}} X_i^{br} \tag{2.9}$$

By incorporating the rates in the mass balance equation for each component and assuming that the biological reactor is perfectly stirred, the model of the system can be stated as:

$$\dot{\boldsymbol{x}} = \boldsymbol{f}^{(1)}\left(\boldsymbol{x}\right) \qquad \text{(aeration periods)} \tag{2.10}$$
$$\dot{\boldsymbol{x}} = \boldsymbol{f}^{(2)}\left(\boldsymbol{x}\right) \qquad \text{(non-aeration periods)} \tag{2.11}$$

where $\boldsymbol{f}^{(1)}$ and $\boldsymbol{f}^{(2)}$ are right hand sides of the differential equations given by:

- For soluble components ($i = 1, 2, 7, 8, 9$):

$$f_i^{(1)}\left(\boldsymbol{x}\right) = f_i^{(2)}\left(\boldsymbol{x}\right) = \frac{Q^{in}}{V^{br}}\left(x_i^{in} - x_i^{br}\right) + r_i \tag{2.12}$$

- For particulate components ($i = 3, 4, 5, 6, 10$):

$$f_i^{(1)}\left(\boldsymbol{x}\right) = f_i^{(2)}\left(\boldsymbol{x}\right) = \frac{1}{V^{br}}\left[Q^{in}\left(x_i^{in} - x_i^{br}\right) + Q^{rs}\frac{Q^{in} - Q^{w}}{Q^{rs} + Q^{w}} x_i^{br}\right] + r_i \tag{2.13}$$

- For dissolved oxygen concentration ($i = 11$):

$$f_{11}^{(1)}\left(\boldsymbol{x}\right) = \frac{Q^{in}}{V^{br}}\left(x_{11}^{in} - x_{11}^{br}\right) + r_{11} + \mathcal{A}_{\mathcal{O}} \tag{2.14}$$
$$f_{11}^{(2)}\left(\boldsymbol{x}\right) = \frac{Q^{in}}{V^{br}}\left(x_{11}^{in} - x_{11}^{br}\right) + r_{11} \tag{2.15}$$

As we can see, the mass balance equation related to the concentration of dissolved oxygen may contain an additional term $\mathcal{A}_O$ which describes the oxygen transfer from the turbines:

$$\mathcal{A}_O = k_L a \left(S_O^{sat} - S_O \right) \tag{2.16}$$

where $k_L a$ is the oxygen transfer coefficient and S_O^{sat} is the dissolved oxygen saturation concentration ($S_O^{sat} = 10\,\mathrm{mg/L}$). Hence, the process is described by two models depending on whether the aeration system is on or off. Such systems are known as *hybrid discrete/continuous dynamic systems* (Barton and Pantelides, 1994).

2.3 Modified Model

The original model is described by two sets of differential equations (2.10), (2.11) that are switched depending on whether the aeration is on or off. Such a model is not directly suitable for a general purpose simulation or dynamic optimisation packages and it is necessary to implement a specialised code that is able to handle this situation.

In order to apply standard simulation or optimisation packages, it was necessary to modify the original model. Although that seems to be difficult, in fact, only a minor modifications had to be done. It can be observed, that the first ten differential equations are the same and only the equation (2.14) that describes the mass balance related to the concentration of dissolved oxygen is different. This suggest the introduction of the following modified equation

$$f_{11}\left(\boldsymbol{x}\right) = \frac{Q^{in}}{V^{br}} \left(x_{11}^{in} - x_{11}^{br} \right) + \mathcal{R}_{11} + \mathcal{A}_{\mathcal{O}} u_1 \tag{2.17}$$

where a new input variable u_1 has been introduced to the model. This variable is piece-wise constant and has values 0 and 1.

Therefore, the modified model can be written as

$$\dot{\boldsymbol{x}} = \boldsymbol{f}\left(\boldsymbol{x}, u_1\right) \tag{2.18}$$

Chapter 3

Control Problem Definition

Organic and nitrogen removal is achieved by switching the turbines on and off, resulting in alternated aerobic-anoxic conditions in the aeration basin. Hence, the process can be seen as a succession of cycles where each cycle consists of an aeration period followed by a non-aeration period, *i.e.* the period between two consecutive starts of the turbines. Each cycle is then completely determined by means of two parameters - time of aeration and time of non-aeration. Thus, for a specified number N_c of cycles, the total number of parameters involved in optimisation is $2 \times N_c$ and these are denoted by Δt_j. We will assume that at the beginning, the first operation will be aeration, therefore the odd times Δt_j will correspond to aeration and the even times to non-aeration.

3.1 Performance index

The objective of this study is to determine the aeration strategy that minimizes the operating costs of the process. The energy induced by the aeration process represents a significant part (60–80%) of the operating costs of the wastewater treatment plant. Therefore, the optimisation of the aeration process can result in significant costs savings. Other costs such as those associated with pumping are neglected.

The energy $\mathcal{E}$ which is dissipated by the aeration system is defined as:

$$\mathcal{E} = \int_{t_0}^{t_f} \mathcal{P}(t)\, \mathrm{d}t \tag{3.1}$$

where $\mathcal{P}$ denotes the power consumption. Here, oxygen is provided in the reactor by means of surface turbines with fixed rotating speed. For typical values of the sludge concentration in the reactor (*e.g.* between 1 g/L and 5 g/L), the flow is shown to be turbulent and the power consumption is therefore independent on the sludge concentration:

$$\mathcal{P} = N_p\, \rho \mathcal{N}^3\, d^5 \tag{3.2}$$

where N_p is the power number ($N_p = \mathrm{C}^{\underline{\mathrm{te}}}$), ρ the liquid density, $\mathcal{N}$ the aerator rotation speed and d the aerator diameter.

It is noteworthy that extra-power consumption induced by the starting process of the turbines is not accounted for in this work. Therefore, the energy consumption is directly proportional to the aeration time and for a given number N_c of cycles, one can express the energy consumption as:

$$\mathcal{E} = \mathcal{P} \sum_{j=1}^{N_c} \Delta t_{2j-1} \tag{3.3}$$

The performance index J_0 is here defined as the total aeration period divided by the time horizon:

$$J_0 = \frac{\sum_{j=1}^{N_c} \Delta t_{2j-1}}{\sum_{j=1}^{2N_c} \Delta t_j} \tag{3.4}$$

J_0 is a dimensionless variable and can be seen as the aeration rate over the optimisation horizon: the smaller the aeration rate, the larger the reduction of energy consumption.

3.2 Constraints

Two different sets of constraints are considered in order (*i*) to satisfy the requirements imposed on the residual concentrations in the effluent, and (*ii*) to ensure the feasibility of the computed aeration profiles.

3.2.1 Effluent requirements

Stricter regulations have been imposed during the last decade by the European Union on the effluent of wastewater treatment plants. Agglomeration effluents with a population-equivalent greater than $2,000$ should respect a minimum of secondary treatment (biological treatment with a secondary settler). The standards in terms of COD, BOD_5 and SS (suspended solids) are given by:

$$\mathrm{COD}_{\max} = 125\,\mathrm{mg/L} \tag{3.5}$$

$$\mathrm{BOD}_{5\max} = 25\,\mathrm{mg/L} \tag{3.6}$$

$$\mathrm{SS}_{\max} = 35\,\mathrm{mg/L} \tag{3.7}$$

The limit values for nitrogen are 15 mg/L (measured as N) for smaller agglomerations. However, it is worth noticing that an objective of 10 mg/L is often aimed at making the plant operation more flexible:

$$\mathrm{TN}_{\max} = 10\,\mathrm{mg/L} \tag{3.8}$$

3.2.2 Operating constraints

Operating constraints are imposed along with the aforementioned constraints on maximum residual concentrations. The objective of these additional constraints is (*i*) to prevent the turbines from damaging and (*ii*) to include aspects which are not described by the model.

Two operating constraints deal with minimum air-on ($t_{\min}^{\mathrm{on}}$) and air-off ($t_{\min}^{\mathrm{off}}$) cycle time. These lower limits are defined to avoid too frequent cycling of the turbines. Moreover, the minimum air-on time must be long enough to re-aerate most of the activated sludge after anoxic periods and to ensure that biomass and substrate are conveniently mixed in the aeration tank. In this work, both air-on and air-off times are set to 15 minutes.

A maximum air-on time ($t_{\max}^{\mathrm{on}}$) is also defined to prevent the propellers from early wear or from damaging. The upper limit is set to 120 minutes. At last, it is worth noticing that too long non-stirring and non-aeration periods can induce floc sedimentation in the aeration tank as well as anaerobic conditions, hence modifying the degradation performances. In addition, the CSTR assumption may be no longer valid for too long non-stirring periods. Therefore, a maximum air-off ($t_{\max}^{\mathrm{off}}$) time of 120 minutes is considered.

3.3 Optimisation Problem

Rigorously speaking, although the model has been modified for not to be hybrid discrete/continuous, the computation of aeration and non-aeration sequences which minimize the energy consumption is still stated as a *mixed-integer dynamic optimisation* (MIDO) problem since the optimal number of cycles N_c must be determined as well as the times Δt_j. However, (non-convex) MIDO problems are known to be difficult to solve (Allgor and Barton, 1997) and are currently under investigation. In this study, it has been chosen to keep the number of cycles constant in order to deal with a more regular optimisation problem. The computation of optimal aeration strategies for a varying number of cycles will be the topic for further research.

The dynamic optimisation problem on a given time horizon t_f can be stated as:

$$\begin{array}{rrcl}
\min_{\Delta t_1,\ldots,\Delta t_{2N_c}} & J_0 & = & \frac{\sum_{j=1}^{N_c} \Delta t_{2j-1}}{\sum_{j=1}^{2N_c} \Delta t_j} \\
\text{subject to :} & \dot{\boldsymbol{x}} & = & \boldsymbol{f}(\boldsymbol{x}, u_1) \\
 & 0 & \leq & \mathrm{TN}_{\max} - \left[S_{NO}^{br}(t) + S_{NH}^{br}(t) + S_{ND}^{br}(t)\right] \ , \ \forall t \\
 & 0 & \leq & \mathrm{COD}_{\max} - \left[S_{I}^{br}(t) + S_{S}^{br}(t)\right] \ , \ \forall t \\
 & 0 & \leq & \mathrm{BOD}_{5\max} - 0.25\left[S_{I}^{br}(t) + S_{S}^{br}(t)\right] \ , \ \forall t \\
 & t_{\max}^{\text{on}} & \geq & \Delta t_{2j-1} \ \geq \ t_{\min}^{\text{on}} \ , \ j = \overline{1, N_c} \\
 & t_{\max}^{\text{off}} & \geq & \Delta t_{2j} \ \geq \ t_{\min}^{\text{off}} \ , \ j = \overline{1, N_c} \\
 & t_f & = & \sum_{j=1}^{2N_c} \Delta t_j
\end{array} \tag{3.9}$$

Note that the limitation on the residual concentration of suspended solids is not included in the optimisation problem, since dynamic model (2.18) does not deal with particulate material in the effluent (the settler is assumed to work perfectly). The three remaining effluent requirements on COD, BOD$_5$ and TN are inequality path constraints that can be written in a general form as:

$$N(\boldsymbol{x}) - N_{\max} \ \leq \ 0 \tag{3.10}$$

To solve the dynamic optimisation problem, these constraints are converted into integral path constraints defined as :

$$J \ = \ \int_{t_0}^{t_f} F(\boldsymbol{x})\mathrm{d}t \ = \ \int_{t_0}^{t_f} \max\left[N(\boldsymbol{x}) - N_{\max}\, ;\, 0\right]^2 \mathrm{d}t \tag{3.11}$$

Chapter 4

Description of the Optimisation Method

4.1 System and Cost Description

Consider an ordinary differential system (ODE) system described by the following equations

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(t, \boldsymbol{x}, \boldsymbol{u}, \boldsymbol{p}) \tag{4.1}$$

where t denotes time, $\boldsymbol{x} \in \mathcal{R}_{n_x}$ is the vector of differential variables, $\boldsymbol{u} \in \mathcal{R}_{n_u}$ is the vector of controls, and $\boldsymbol{p} \in \mathcal{R}_{n_p}$ is the vector of parameters. The vector valued function $\boldsymbol{f} \in \mathcal{R}_{n_x}$ describes right hand sides of differential equations.

Consider now the criterion to be minimised and constraints of the form

$$J_0 = G_0(t_j, \boldsymbol{x}(t_f), \boldsymbol{p}) + \int_0^{t_f} F_0(t, \boldsymbol{x}, \boldsymbol{u}, \boldsymbol{p}) \mathrm{d}t \tag{4.2}$$

$$J_i = G_i(t_j, \boldsymbol{x}(t_f), \boldsymbol{p}) + \int_0^{t_f} F_i(t, \boldsymbol{x}, \boldsymbol{u}, \boldsymbol{p}) \mathrm{d}t \tag{4.3}$$

where the constraints are for $i = 1, \ldots, m$, (m is the number of constraints). The commutation times (times when the piece-wise control changes) are denoted by t_j, $j = 1, \ldots, P$

As it will be shown below, this formulation is general enough for the purposes needed.

4.2 Optimised Variables

The optimised variables $\boldsymbol{y}$ are parameters $\boldsymbol{p}$, piece-wise constant parametrisation of control

$$\boldsymbol{u}(t) = \boldsymbol{u}_j, \quad t_{j-1} \le t < t_j \tag{4.4}$$

and the time increments $\Delta t_j = t_j - t_{j-1}$.

Hence the vector $\boldsymbol{y} \in \mathcal{R}_q$ of optimised variables is given as

$$\boldsymbol{y}^T = (\Delta t_1, \ldots, \Delta t_P, \boldsymbol{u}_1^T, \ldots, \boldsymbol{u}_P^T, \boldsymbol{p}^T). \tag{4.5}$$

where P is the number of piece-wise constant control segments.

Although the piece-wise control approximation seems to be rather restrictive, actually any other control parametrisation can be described by its means - for example piece-wise linear control given as

$$u(t) = a_0 + a_1 t \tag{4.6}$$

can be seen as a piece-wise constant strategy with controls a_0, a_1.

4.3 Static Optimisation Problem

As the control trajectory is considered to be piece-wise constant, the original problem of dynamic optimisation has been converted into static optimisation – non-linear programming.

We then utilise static non-linear optimisation solver (NLP) of the form:

$$\begin{aligned} &\min_{\boldsymbol{y}} g_0(\boldsymbol{y}) \quad \text{subject to:} \\ &\qquad g_i(\boldsymbol{y}) = 0 \qquad i = 1 \dots m_e \\ &\qquad g_i(\boldsymbol{y}) \geq 0 \qquad i = m_e + 1 \dots m_i \end{aligned} \tag{4.7}$$

In addition to the cost function and constraints, their gradients with respect to optimised variables y must be given. Their calculation can be based on several methods. We have implemented two methods: adjoint approach based on optimality conditions, and finite differences.

4.4 Gradient Derivation

When the dynamic optimisation problem is to be solved, the nonlinear programming (NLP) solver needs to know gradients of the cost (and the constraints) with respect to the vector $\boldsymbol{y}$ of the optimised variables.

The equation (4.1) is a constraint to the cost function J_i and is adjoined to it by a vector of non-determined adjoint variables $\boldsymbol{\lambda}_i(t) \in \mathcal{R}_{n_x}$, thus

$$J_i = G_i + \int_0^{t_f} (F_i + \boldsymbol{\lambda}_i^T (\boldsymbol{f} - \dot{\boldsymbol{x}}))\mathrm{d}t \tag{4.8}$$

For any J_i we can form a Hamiltonian H_i defined as

$$H_i(t, \boldsymbol{x}, \boldsymbol{u}, \boldsymbol{p}, \boldsymbol{\lambda}) = F_i + \boldsymbol{\lambda}_i^T \boldsymbol{f} \tag{4.9}$$

Substituting for F_i in (4.2) yields for J_i

$$J_i \quad = \quad G_i + \int_0^{t_f} (H_i - \boldsymbol{\lambda}_i^T \dot{\boldsymbol{x}})\mathrm{d}t \tag{4.10}$$

In order to derive the necessary optimality conditions, variation of the cost is to be found. Taking a variation of the cost (see Bryson and Ho (1975)) and using integration by parts gives

$$\begin{aligned} \mathrm{d}J_i \quad = \quad & \frac{\partial G_i}{\partial \boldsymbol{x}^T(t_f)} \mathrm{d}\boldsymbol{x}(t_f) + \frac{\partial G_i}{\partial \boldsymbol{p}^T} \mathrm{d}\boldsymbol{p} + \sum_{j=1}^{P} \frac{\partial G_i}{\partial t_j} \mathrm{d}t_j \\ & + \sum_{j=1}^{P} \left[H_i - \boldsymbol{\lambda}_i^T \dot{\boldsymbol{x}}\right]_{t=t_j^-} \mathrm{d}t_j - \left[H_i - \boldsymbol{\lambda}_i^T \dot{\boldsymbol{x}}\right]_{t=t_{j-1}^+} \mathrm{d}t_{j-1} \\ & + \sum_{j=1}^{P} \left[-\boldsymbol{\lambda}^T \delta \boldsymbol{x}\right]_{t_{j-1}^+}^{t_j^-} \\ & + \sum_{j=1}^{P} \int_{t_{j-1}^+}^{t_j^-} \left[\left(\dot{\boldsymbol{\lambda}}_i^T + \frac{\partial H_i}{\partial \boldsymbol{x}^T}\right) \delta \boldsymbol{x} + \frac{\partial H_i}{\partial \boldsymbol{u}^T} \delta \boldsymbol{u}\right] \mathrm{d}t \\ & + \int_0^{t_f} \left[\frac{\partial H_i}{\partial \boldsymbol{p}^T} \delta \boldsymbol{p}\right] \mathrm{d}t \end{aligned} \tag{4.11}$$

where t_j^- signifies the time just before $t = t_j$ and t_j^+ is the time just after $t = t_j$.

Eliminating $\delta \boldsymbol{x}(t_j)$ using the relations

$$\mathrm{d}\boldsymbol{x}(t_j) = \delta \boldsymbol{x}(t_j^{\pm}) + \dot{\boldsymbol{x}}(t_j^{\pm})\mathrm{d}t_j \tag{4.12}$$

we get

$$\begin{aligned}
\mathrm{d}J_i &= \left[\frac{\partial G_i}{\partial \boldsymbol{x}^T(t_f)} - \boldsymbol{\lambda}^T(t_f)\right]\mathrm{d}\boldsymbol{x}(t_f) + \frac{\partial G_i}{\partial \boldsymbol{p}^T}\mathrm{d}\boldsymbol{p} \\
&+ \sum_{j=1}^{P}\left[\frac{\partial G_i}{\partial t_j} + H_j(t_j^-) - H_{j+1}(t_j^+)\right]\mathrm{d}t_j \\
&+ \sum_{j=1}^{P-1}\left[\boldsymbol{\lambda}^T(t_j^+) - \boldsymbol{\lambda}^T(t_j^-)\right]\mathrm{d}\boldsymbol{x}(t_j) \\
&+ \sum_{j=1}^{P}\int_{t_{j-1}^+}^{t_j^-}\left[\left(\dot{\boldsymbol{\lambda}}_i^T + \frac{\partial H_i}{\partial \boldsymbol{x}^T}\right)\delta\boldsymbol{x} + \frac{\partial H_i}{\partial \boldsymbol{u}^T}\delta\boldsymbol{u}\right]\mathrm{d}t \\
&+ \int_0^{t_f}\left[\frac{\partial H_i}{\partial \boldsymbol{p}^T}\delta\boldsymbol{p}\right]\mathrm{d}t
\end{aligned} \tag{4.13}$$

where is to be noted that $H_{P+1} = H(t_f^+) = 0$.

In order to simplify the expressions, we choose the vector $\boldsymbol{\lambda}(t)$ such that

$$\dot{\boldsymbol{\lambda}}_i^T = -\frac{\partial H_i}{\partial \boldsymbol{x}^T} \tag{4.14}$$

$$\boldsymbol{\lambda}^T(t_f) = \frac{\partial G_i}{\partial \boldsymbol{x}^T(t_f)} \tag{4.15}$$

$$\boldsymbol{\lambda}^T(t_j^+) = \boldsymbol{\lambda}^T(t_j^-), \quad j = 1, \ldots, P-1 \tag{4.16}$$

The variation of J_i can finally be expressed as

$$\begin{aligned}
\mathrm{d}J_i &= \sum_{j=1}^{P}\left[\frac{\partial G_i}{\partial t_j} + H_j(t_j^-) - H_{j+1}(t_j^+)\right]\mathrm{d}t_j \\
&+ \left[\frac{\partial G_i}{\partial \boldsymbol{p}^T} + \int_0^{t_f}\frac{\partial H_i}{\partial \boldsymbol{p}^T}\mathrm{d}t\right]\delta\boldsymbol{p} \\
&+ \sum_{j=1}^{P}\left[\int_{t_{j-1}^+}^{t_j^-}\frac{\partial H_i}{\partial \boldsymbol{u}^T}\mathrm{d}t\right]\delta\boldsymbol{u}
\end{aligned} \tag{4.17}$$

The conditions of optimality follow directly from the last equation. As it is required, that the variation of the cost J_i should be zero at the optimum, all terms in brackets have to be zero.

4.4.1 Procedure

Assume that functions G_i, F_i and their partial derivatives with respect to $t_j, \boldsymbol{x}, \boldsymbol{u}, \boldsymbol{p}$ are specified. Also needed is the function $\boldsymbol{f}$ and its derivatives with respect to $\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{p}$.

The actual algorithm can briefly be given as follows :

1. Integrate the system (4.1) and integral terms F_i together from $t = 0$ to $t = t_f$,
2. For $i = 0, \ldots, m$ repeat
   (a) Initialise adjoint variables $\boldsymbol{\lambda}_i(t_f)$ as

$$\boldsymbol{\lambda}(t_f) = \frac{\partial G_i}{\partial \boldsymbol{x}^T(t_f)} \tag{4.18}$$

   (b) Initialise the intermediate variables $\boldsymbol{J}_u$, $\boldsymbol{J}_p$ as zero

(c) Integrate backwards from $t = t_f$ to $t = 0$ the adjoint system and intermediate variables

$$\dot{\boldsymbol{\lambda}}_i^T = -\frac{\partial H_i}{\partial \boldsymbol{x}^T} \tag{4.19}$$

$$\dot{\boldsymbol{j}}_u^T = \frac{\partial H_i}{\partial \boldsymbol{u}^T} \tag{4.20}$$

$$\dot{\boldsymbol{j}}_p^T = \frac{\partial H_i}{\partial \boldsymbol{p}^T} \tag{4.21}$$

(d) Calculate the gradients of J_i with respect to times t_j, control $\boldsymbol{u}$ and parameters $\boldsymbol{p}$

$$\frac{\partial J_i}{\partial t_j} = \frac{\partial G_i}{\partial t_j} + H_j(t_j^-) - H_{j+1}(t_j^+) \tag{4.22}$$

$$\frac{\partial J_i}{\partial \boldsymbol{p}} = \frac{\partial G_i}{\partial \boldsymbol{p}^T} + \boldsymbol{J}_p(0) \tag{4.23}$$

$$\frac{\partial J_i}{\partial \boldsymbol{u}_j} = \boldsymbol{J}_u(t_{j-1}) - \boldsymbol{J}_u(t_j) \tag{4.24}$$

In this manner, the values of J_i are obtained in the step 1 and the values of gradients in the step 2d. This is all what is needed as input to non-linear programming routines – in our case NLPQL Schittkowski (1985).

Notes

Gradients with respect to times

The expressions (4.22) for the calculation of the gradient of the cost with respect to time did not take into account that the time increments rather than times are optimised. The relations between times and their increments are given as

$$\begin{aligned} t_1 &= \Delta t_1 \\ t_2 &= \Delta t_1 + \Delta t_2 \\ &\vdots \\ t_P &= \textstyle\sum_{j=1}^{P} \Delta t_j \end{aligned} \tag{4.25}$$

As the following holds for the derivatives

$$\frac{\partial J_i}{\partial \Delta t_j} = \sum_{k=1}^{P} \frac{\partial J_i}{\partial t_k} \frac{\partial t_k}{\partial \Delta t_j} \tag{4.26}$$

we finally get the desired expressions

$$\frac{\partial J_i}{\partial \Delta t_j} = \sum_{k=j}^{P} \frac{\partial J_i}{\partial t_k} \tag{4.27}$$

Integration of adjoint equations

When the adjoint equations are integrated backwards in time, the knowledge of states $\boldsymbol{x}(t)$ is needed. There are several ways to supply this information. For example, the state equations can be integrated together with adjoint equations backwards. Although this is certainly a correct approach, there may be numerical problems as the backward integration of states can be unstable. In Rosen and Luus (1991), the states are stored in equidistant intervals and integration of both states and adjoint equations is corrected at the begin of each interval. We have adopted another approach : stored the state vector in every integration step in forward pass and interpolated states in backward pass. The drawback of this approach is large memory requirement. It can be modified to store only certain number of integration steps. Several types of interpolation have been tested, the best results have been obtained with the approximations having continuous first derivatives. Although the time needed for calculation of such approximations is longer, the adjoint equations are easier to integrate and the overall time of gradient calculations has been greatly reduced.

It is always recommended to implement at least two methods of gradients calculation. In this manner, a user can cross-check if the gradients are correct. Also, if there is a problem in NLP algorithm, the gradient method can be changed.

Therefore, we have also implemented the method of finite differences: The system (4.1) is integrated q times and at each time one y_i is slightly perturbed. After the integrations, the gradients are given as

$$\nabla_{y_j} g_i = \frac{g_i(y_1, \ldots, \Delta y_j, \ldots, y_q) - g_i(\boldsymbol{y})}{\Delta y_j}, \quad i = 0, \ldots, m_e + m_i \tag{4.28}$$

4.5 Implementation

The above described algorithm of dynamic optimisation has been implemented in a rather general form in programming language FORTRAN. Its short user manual can be found in the Appendix A.

Chapter 5

Simulation Results

To find the optimal aeration profiles over a period of one day, the number of cycles per day N_c has to be specified. Although it may be treated as a variable of optimisation, the resulting problem is not to be solved easily, as it contains integer variables. Therefore, we have fixed N_c and tried to find in several optimisation runs, which value of N_c gives the smallest aeration rate. This optimum has been found at $N_c = 29$ (see also Chachuat et al. (2001)).

The solution statistics shown in the Table 5.1 reports IVP precision, which is the integration tolerance. NLP precision was used for the NLP solver. The number of iterations is a measure how difficult the problem is to solve, as about 90% of the computational time is spent for the integration. Several tolerance levels are given and the initial guess at the tighter tolerances was the final solution in the previous line. This is called the *sequenced initial guess method*. The rationale behind is that at the looser tolerances, the integration of the system and adjoint equations is much faster as at the tighter tolerances. The start of the optimisation is far away from the optimum and thus the precision of gradients is not so crucial. The required time for one iteration varies between 3 seconds with the loosest tolerances to several minutes with the tightest tolerances.

Iterations	IVP precision	NLP precision	Minimum
484	10^{-7}	10^{-7}	32.200436
323	10^{-8}	10^{-8}	31.922180
401	10^{-9}	10^{-9}	31.388008
1019	10^{-10}	10^{-10}	31.145038
402	10^{-11}	10^{-11}	31.137528

Table 5.1: Summary of the results for 29 cycles per day

We can see from Table 5.1 that an approximate minimum can be found very quickly with loose tolerances. However, to find the neighbourhood of the true optimum, several thousands of iterations have to be performed. The results at the tightest precisions are shown in Fig. 5.1 where the nitrogen concentration during the whole day is plotted as well as the corresponding state of the turbines. Although three effluent constraints have been specified in the problem formulation, only the nitrogen concentration is plotted, as the others were satisfactorily satisfied. The trajectory of aeration rates through the day is shown in Fig. 5.2.

These results give the minimum possible aeration rates for the given plant and provide several outcomes. The aeration rate can be compared with the actually used rates and helps to determine whether there is a room for an improvement over the existing aeration strategies and whether it is reasonable to make investments. Moreover, the aeration rates per day can serve as setpoint trajectories at the existing plant.

The drawback of the previous solution is the large amount of the computational time required to find the solution. The further aim of this study is to determine, whether it is possible to simplify the original problem and to obtain the comparable results with much less computations.

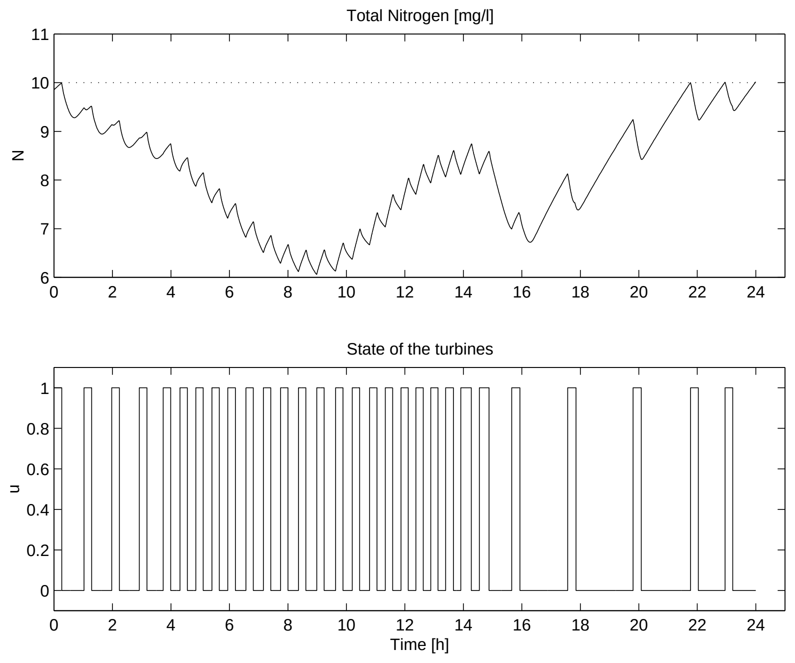

Figure 5.1: Concentration of the nitrogen during the day and the corresponding state of the turbines for 29 cycles per day

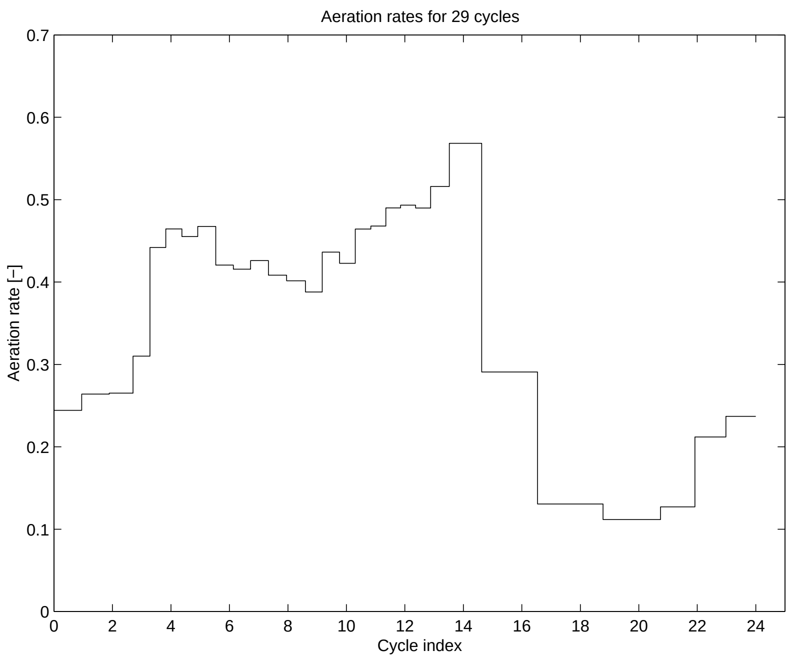

Figure 5.2: Optimal aeration rates for 29 cycles per day

Large number of iterations can by attributed to the number of variables that are to be optimised. When this number is reduced, the optimum can be found more rapidly.

Therefore, we will make the following modifications: the number of cycles per day will be reduced and it will be assumed that the cycles have the same time durations. The second assumption introduces the following equality constraints into the problem (3.9)

$$\Delta t_{2j-1} + \Delta t_{2j} = \frac{t_f}{N_c}, \quad j = 1, \dots, N_c \tag{5.1}$$

Two optimisation problems with 12 and 15 cycles per day have been treated. The results are shown in Figures 5.3, 5.4 for 12 cycles and in Figures 5.5, 5.6 for 15 cycles. Optimal aeration rate of 31.75 has been achieved after 209 iterations with 12 cycles and the optimum rate of 31.63 after 190 iterations with 15 cycles and the same initial conditions and precisions.

As we can see from the graphs, the optimum aeration rate for 12 cycles is not very similar to the optimum with 29 cycles. On the other hand, the 15 cycles per day strategy resembles better the 29 cycles strategy and can be thought as a good approximation of the optimum.

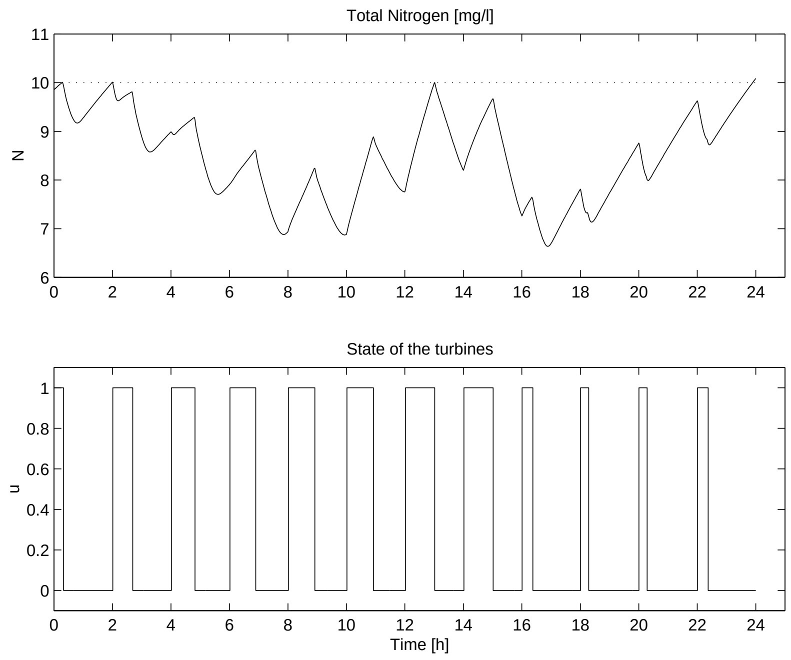

Figure 5.3: Concentration of the nitrogen during the day and the corresponding state of the turbines for 12 equal cycles per day

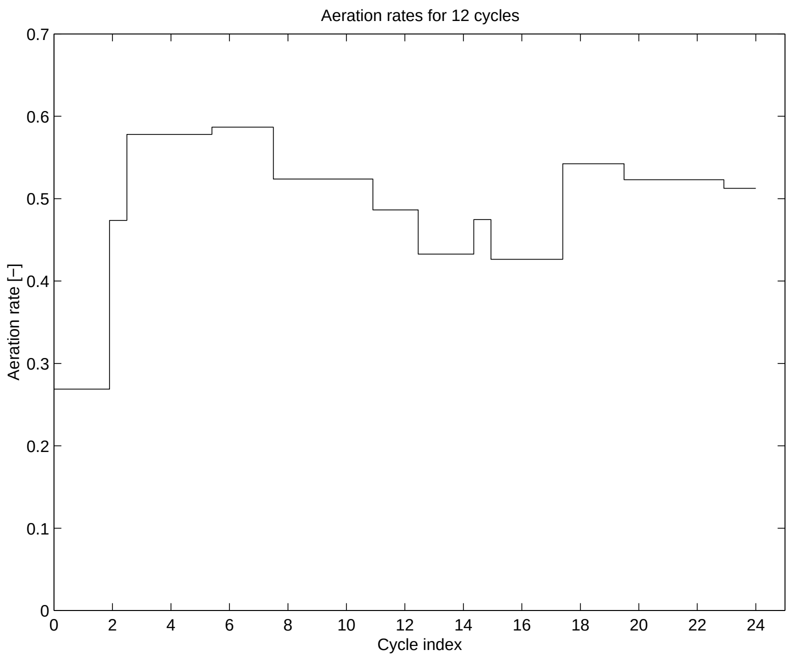

Figure 5.4: Optimal aeration rates for 12 equal cycles per day

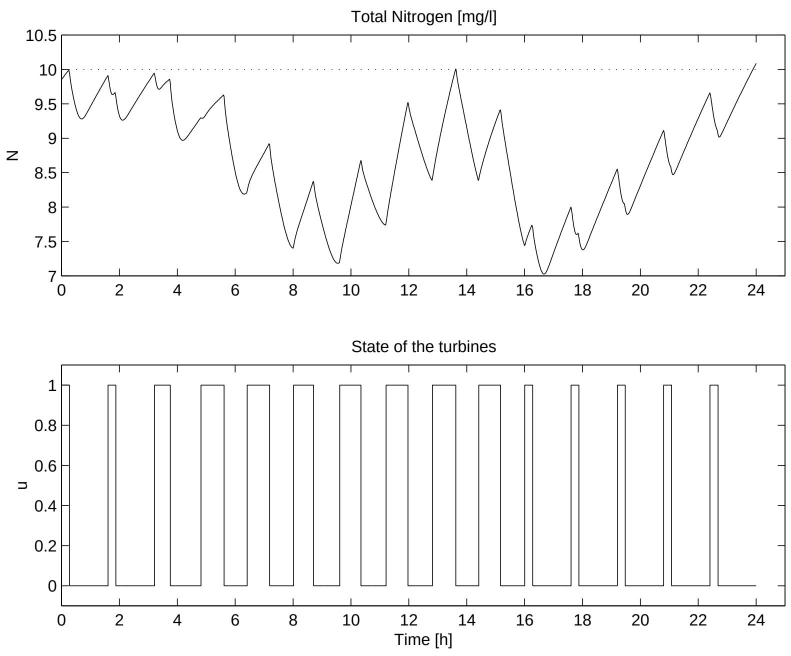

Figure 5.5: Concentration of the nitrogen during the day and the corresponding state of the turbines for 15 equal cycles per day

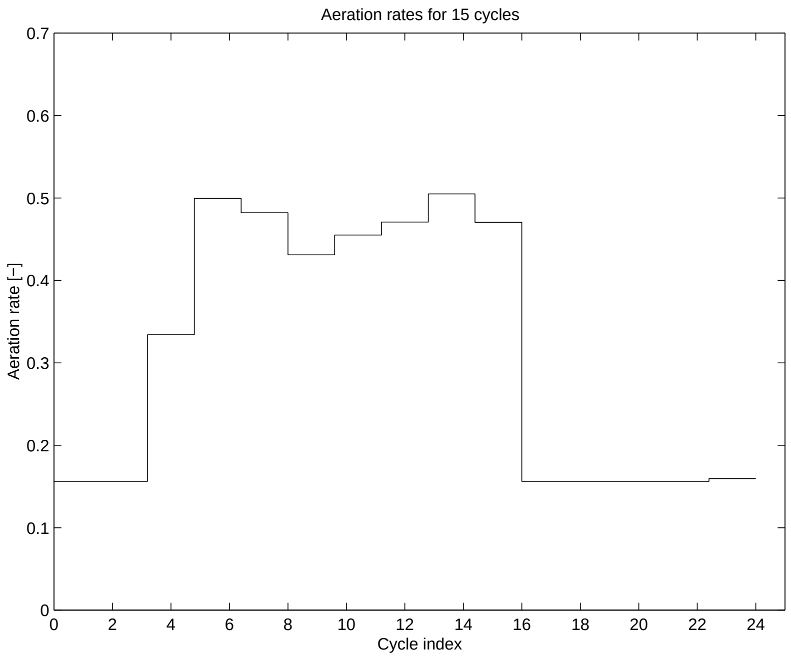

Figure 5.6: Optimal aeration rates for 15 equal cycles per day

Chapter 6

Conclusions

This report dealt with the determination of the optimal aeration strategies for small-size activated sludge plants and with the implementation of the dynamic optimisation.

In the first part, the importance of the problem was stated – as the new EU directives for the wastewater quality are specified, many existing wastewater plants have to be redesigned or their aeration strategies changed.

The approach to the solution that was undertaken here consisted in modelling of the plant and using the results from optimal control theory to give answers to some possible questions concerning reduction of the operational costs as well as specification of the optimal aeration profiles.

The implemented package `DYNO` enables to optimise a large class of processes described by ordinary differential equations. With the aid of this package, we are in position to solve the problems dealing with dynamic properties of the wastewater plants.

Acknowledgments

The author would like to thank the members of the Group TASC, especially M. A. Latifi and B. Chachuat for many inspiring discussions during this work.

Bibliography

J. Alex, J.-F. Beteau, J.B. Copp, C. Hellinga, U. Jeppsson, S. Marsili-Libelli, M.-N. Pons, H. Spanjers, and H. Vanhooren. Benchmark for Evaluating Control Strategies in Wastewater Treatment Plants. In *Proc. ECC'99*, Karlsruhe, Germany, 1999.

R.J. Allgor and P.I. Barton. Mixed-integer dynamic optimization. *Computers chem. Engng.*, 21 (Suppl.):S451–S456, 1997.

P.I. Barton and C.C. Pantelides. Modelling of Combined Discrete/Continuous Processes. *AIChE Journal*, 40(6):966–979, 1994.

A. E. Bryson, Jr. and Y. C. Ho. *Applied Optimal Control*. Hemisphere Publishing Corporation, 1975.

B. Chachuat, N. Roche, and M. A. Latifi. Dynamic optimisation of small size wastewater treatment plants including nitrification and denitrification processes. *Computers chem. Engng.*, 25(4–6): 585–593, 2001.

S. Crescitelli and S. Nicoletti. Near optimal control of batch reactors. *Chem. Eng. Sci.*, 28:463–471, 1973.

O. J. Hao and J. Huang. Alternating aerobic-anoxic process for nitrogen removal : Process evaluation. *Water Environ. Res.*, 68(1):83–93, 1996.

M. Henze, C. P. L. Grady, W. Gujer, G. v. R. Marais, and T. Matsuo. Activated Sludge Model No. 1. Technical Report 1, IAWQ, LONDON, 1987.

S. Julien. *Modélisation et estimation pour le contrôle d'un processus boues activées éliminant l'azote des eaux résiduaires urbaines*. PhD thesis, Institut National Polytechnique de Toulouse, 1997.

O. Rosen and R. Luus. Evaluation of gradients for piecewise constant optimal control. *Computers chem. Engng.*, 15(4):273–281, 1991.

K. Schittkowski. NLPQL : A FORTRAN subroutine solving constrained nonlinear programming problems. *Annals of Operations Research*, 5:485–500, 1985.

Appendix A

DYNO - A User Manual

This chapter describes features implemented in the dynamic optimisation package `DYNO`. In general, the package finds optimal control trajectory with variable time segments and with unknown parameters given the description of the process, the cost to be minimised, subject to equality and inequality constraints.

A.1 Required Files

The minimum set of files/subroutines required is follows:

`dyno.f` – dynamic optimisation,

`nlpql.f` – NLP solver,

`vodedo.f` – modified ODE solver `dvode.f`. Modifications include additional parameters passed to all relevant subroutines. The additional parameters are `rwmifi, iwmifi` and serve for communication between `dvode.f` and `dyno.f`,

`main.f` – main program,

`process.f` – definition of the optimised process,

`cost.f` – definition of the cost and constraints.

Probably the best way to explain the implementation is by the means of a simple example.

A.2 Batch Reactor Optimisation

Consider a simple batch reactor with reactions $A \rightarrow B \rightarrow C$ and problem of its dynamic optimisation as described in Crescitelli and Nicoletti (1973). The parameters of the reactor are $k_{10} = 0.535e11$, $k_{20} = 0.461e18$, $e_1 = 18000$, $e_2 = 30000$, $r = 2.0$, final time $t_f = 8.0$, $\beta_1 = 0.53$, $\beta_2 = 0.43$, $\alpha = e_2/e_1$, $c = k_{20}/k_{10}^{\alpha}$. For more detailed description of the parameters see Crescitelli and Nicoletti (1973).

The differential equations describing the process are as follows

$$\dot{x}_1 = f_1 = -ux_1 \tag{A.1}$$

$$\dot{x}_2 = f_2 = ux_1 - cu^{\alpha}x_2 \tag{A.2}$$

with the initial state

$$x_1(0) = \beta_1, \quad x_2(0) = \beta_2 \tag{A.3}$$

The control variable u is related to the reactor temperature T via the relation

$$T = -\frac{e_1}{r \log \frac{u}{k_{10}}} \tag{A.4}$$

The objective of the optimisation is to maximise the yield of product B at time t_f: $x_2(t_f)$ subject to piece-wise constant control. In the original article, 3 piece-wise constant control segments are considered, with segment lenghts being also optimised variables. We will assume P segments. The package `DYNO` assumes minimisation of some cost, therefore

$$J_0 = -x_2(t_f) \tag{A.5}$$

subject to the constraint imposed on the final time

$$-t_f + \sum_{j=1}^{P} \Delta t_j = 0 \tag{A.6}$$

The information needed by `DYNO` consists of all what has been described above as well as various partial derivatives:

Process $\boldsymbol{f}$

- partial derivative of the process $\boldsymbol{f}$ with respect to states $\boldsymbol{x}$:

$$\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}^T} = \begin{pmatrix} -u & 0 \\ u & -cu^{\alpha} \end{pmatrix} \tag{A.7}$$

- partial derivative of the process $\boldsymbol{f}$ with respect to control $\boldsymbol{u}$:

$$\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}^T} = \begin{pmatrix} -x_1 \\ x_1 - cx_2\alpha u^{\alpha-1} \end{pmatrix} \tag{A.8}$$

- partial derivative of the process $\boldsymbol{f}$ with respect to parameters $\boldsymbol{p}$:

$$\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{p}^T} = \boldsymbol{0} \tag{A.9}$$

Cost $J_0 = G_0 + \int F_0$

- partial derivative of the cost with respect to states $\boldsymbol{x}(t_f)$:

$$\frac{\partial G_0}{\partial \boldsymbol{x}^T} = \begin{pmatrix} 0 \\ -1 \end{pmatrix}, \quad \frac{\partial F_0}{\partial \boldsymbol{x}^T} = \boldsymbol{0} \tag{A.10}$$

- partial derivative of cost with respect to control $\boldsymbol{u}(t_f)$ (last segment of control):

$$\frac{\partial G_0}{\partial \boldsymbol{u}^T} = \boldsymbol{0}, \quad \frac{\partial F_0}{\partial \boldsymbol{u}^T} = \boldsymbol{0} \tag{A.11}$$

- partial derivative of the cost with respect to parameters $\boldsymbol{p}$:

$$\frac{\partial G_0}{\partial \boldsymbol{p}^T} = \boldsymbol{0}, \quad \frac{\partial F_0}{\partial \boldsymbol{p}^T} = \boldsymbol{0} \tag{A.12}$$

- partial derivative of the cost with respect to times Δt_j:

$$\frac{\partial G_0}{\partial \Delta t_j} = 0, \quad \frac{\partial F_0}{\partial \Delta t_j} = 0 \tag{A.13}$$

Constraint $J_1 = G_1 + \int F_1$: Only derivatives with respect to times are non-zero, thus

$$\frac{\partial G_1}{\partial \Delta t_j} = 1, \quad j = 1, \ldots, P \tag{A.14}$$

A.2.1 Subroutine `process`

The information about the optimised process is collected in the subroutine `process` with the following syntax:

```
      subroutine process(t, x, nsta, u, ncont, p, npar, sys, nsys, dsys,
     &      ndsys1, ndsys2, ipar, rpar, flag, iout)

      integer nsta, ncont, npar, nsys, ndsys1, ndsys2, ipar, flag, iout
      double precision t, x, u, p, sys, dsys, rpar
      dimension x(nsta), u(ncont), p(npar),
     &      sys(nsys), dsys(ndsys1, ndsys2), rpar(*), ipar(*)
```

The inputs to the routine are actual time `t`, actual state vector `x(nsta)`, actual control vector `u(ncont)`, vector of parameters `p(npar)`, user defined parameters `rpar(*)`, `ipar(*)`, output channel number `iout` as well as `flag` that decides what kind of information is to be returned from the subroutine either in the vector `sys(nsys)` or in the matrix `dsys(ndsys1,ndsys2)`.

`flag` can have the following values:

0 – return the vector $\boldsymbol{f}$ of differential equations in `sys(nsta)`,

1 – return the Jacobian matrix $\partial\boldsymbol{f}/\partial\boldsymbol{x}^T$ in `dsys(nsta,nsta)`,

2 – return the Jacobian matrix $\partial\boldsymbol{f}/\partial\boldsymbol{u}^T$ in `dsys(nsta,ncont)`,

3 – return the Jacobian matrix $\partial\boldsymbol{f}/\partial\boldsymbol{p}^T$ in `dsys(nsta,npar)`,

-1 – return the initial state and control at time $t = 0$ in `x(nsta)` and `u(ncont)`.

-3 – print information about the process using the values of input arguments

In our particular case the body of this subroutine is as follows:

```
      double precision k10, k20, e1, e2, r, tf, beta1, beta2, c, alpha
      k10 = 0.535d11
      k20 = 0.461d18
      e1 = 18000d0
      e2=30000e0
      r=2.0d0
      tf=8.0d0
      beta1 = 0.53d0
      beta2 = 0.43d0
      alpha = e2/e1
      c = k20/(k10**alpha)

      if (flag .eq. -1) then
         u(1) = 0.0d0
         x(1) = beta1
         x(2) = beta2
         return
      end if

      if (flag .eq. 0) then
         sys(1) = -u(1) * x(1)
         sys(2) = u(1) * x(1) - c * (u(1)**alpha) * x(2)
         return
      end if
      if (flag .eq. 1) then
         dsys(1,1) =-u(1)
         dsys(2,1) = u(1)
         dsys(2,2) = - c * (u(1)**alpha)
         return
      end if

      if (flag .eq. 2) then
         dsys(1,1) = -x(1)
         dsys(2,1) = x(1) - c * x(2) * alpha * (u(1)**(alpha-1))
         return
      end if

      if (flag .eq. 3) then
         return
      end if

      if (flag .eq. -3) then
         write(*,100) t/tf, -e1/(r*dlog(u(1)/k10)),u(1), x(1), x(2)
 100     format(f6.3,3X,2f16.6,2X,2f8.4)
         return
      end if
```

A.2.2 Subroutine `cost`

The information about the cost and constraints is collected in the subroutine `cost` with the following syntax:

```
      subroutine cost(t, x, nsta, u, ncont, p, npar, ti, ntime, sys,
     &      nsys, ipar, rpar, flag, xupt, Ncst, Ncste )

      integer nsta, ncont, npar, ntime, ipar, flag, Ncst, Ncste, xupt,
     &      nsys
      double precision t, x, u, p, ti, sys, rpar
      dimension x(nsta), u(ncont), p(npar), ti(ntime), sys(nsys),
     &      ipar(*), rpar(*)
```

The inputs to the routine are actual time `t`, actual state vector `x(nsta)`, actual control vector `u(ncont)`, vector of parameters `p(npar)`, vector of optimised times `ti(ntime)`, user defined parameters `rpar(*), ipar(*)`, number of constraints `Ncst`, number of equality constraints `Ncste` as well as `flag` and `xupt` that together decide what kind of information is to be returned from the subroutine in the vector `sys(nsys)`.

`flag` can have the following values:

1 – return all non-integral terms $\boldsymbol{G}$ of the cost and constraints together at $t = t_f$ in `sys(ncst+1)`,

2 – return all integral terms $\boldsymbol{F}$ of the cost and constraints together at any time in `sys(ncst+1)`,

-10-i – return the non-integral term $\partial G_i/\partial \boldsymbol{z}^T$ ($i = 0, \dots,$`ncst`) in `sys(*)`, where $\boldsymbol{z}$ can be $\boldsymbol{x}$ (xupt=1), $\boldsymbol{u}$ (xupt=2), $\boldsymbol{p}$ (xupt=3), or $\boldsymbol{\Delta t}$ (xupt=4),

10+i – return the integral term $\partial F_i/\partial \boldsymbol{z}^T$ ($i = 0, \dots,$`ncst`) in `sys(*)`, where $\boldsymbol{z}$ can be $\boldsymbol{x}$ (`xupt=1`), $\boldsymbol{u}$ (`xupt=2`), or $\boldsymbol{p}$ (`xupt=3`),

In our particular case the body of this subroutine is as follows:

```
      integer i
      if (flag .eq. 1) then
         sys(1) = -x(2)
         sys(2) = -8.0d0
         do i=1, ntime
            sys(2) = sys(2) + ti(i)
         end do
         return
      end if

      if (flag .eq. 2) then
         sys(1) = 0.0d0
         sys(2) = 0.0d0
         return
      end if

      if (flag .eq. -10) then
         if (xupt .eq. 1) then
            sys(2) = -1.0d0
         end if
         return
      end if

      if (flag .eq. 10) then
         return
      end if

      if (flag .eq. -11) then
         if (xupt .eq. 4) then
            do i=1, ntime
               sys(i) = 1.0d0
            end do
         end if
         return
      end if

      if (flag .eq. 11) then
         return
      end if
```

A.2.3 Subroutine `dyno`

After having set up the process and cost/constraints, it suffices to call the main routine `dyno`. The syntax of the call is as follows:

```
      call dyno(nsta, ncont, npar, ntime, ncst, ncste, ul, u, uu,
     &      pl, p, pu, tl, t, tu, ista, rwork, nrwork, iwork, niwork,
     &      lwork, nlwork, rpar, ipar, ifail)
```

where

`nsta, ncont, npar, ntime` – dimension of states, control, parameters, times,

`ncst, ncste` – total number of constraints and number of equality constraints, `ul, u, uu` – matrices (dimension `ncont, ntimes`): lower, initial, and upper bound on control trajectory,

`pl, p, pu` – vectors (dimension `npar`): lower, initial, and upper bound on parameters,

`tl, t, tu` – vectors (dimension `ntimes`): lower, initial, and upper bound on optimised time intervals,

`ista` – integer vector (dimension `ncst+1`) describing type of a constraint: 0 - does not contain states, 1 - contains states, but it is not a path constraint, 3 - path constraint,

`rwork(nrwork)` – double precision work array,

`iwork(niwork)` – integer work array,

`lwork(nlwork)` – logical work array,

`rpar, ipar` – user defined parameters,

`ifail` – status of the optimisation result.

In addition to this, the first positions of `rwork` and `iwork` should contain various switches, precisions, etc.

The length of the work arrays `nrwork`, `niwork`, and `nlwork` can be determined by specifying some small values and the program will return the correct values.

For our particular case, the main program with 6 time intervals of control is given as

```
      PROGRAM SIM
      implicit none
      integer nsta, ncont, npar, ntime, ncst
c     nsta - dimension of states
c     ncont - dimension of control
c     npar - dimension of parameters
c     ntime - number of time segments
c     ncst - number of constraints
      parameter (nsta = 2, ncont = 1, npar = 0, ntime = 6, ncst = 1)
      integer ncste, ista, nrwork, iwork, niwork, nlwork, ipar, ifail
      double precision ul, u, uu, pl, p, pu, tl, t, tu, rwork, rpar
      logical lwork

      dimension ul(ncont, ntime), u(ncont, ntime), uu(ncont, ntime)
      dimension pl(npar), p(npar), pu(npar)
      dimension tl(ntime), t(ntime), tu(ntime)
      dimension ista(ncst+1)

      parameter (niwork=400, nrwork=60000, nlwork = 50)
      dimension iwork(niwork), rwork(nrwork), lwork(nlwork)
      dimension ipar(10), rpar(50)

      integer i
      double precision tf
      tf=8.0d0

c     number of equality constraints
      ncste = ncst
c     constraints are: without/with states (0/1)
c     path constraints (0/2)
```

```
c     difficult constraints (0/4)
      ista(1) = 1
      ista(2) = 0
c     rtol for DVODE
      rwork(1) = 1d-11
c     atol for DVODE
      rwork(2) = 1d-15
c     acc for NLP
      rwork(3) = 1d-13
c     minimum rtol for DVODE
      rwork(4) = 1d-3
c     maxfun ! max number of function call evaluations in NLP/line search
      iwork(1) = 250
c     maxit  ! max number of iterations in NLP
      iwork(2) = 300
c     iprint ! level of information printed by the subroutine
      iwork(3) = 2
c     iout   ! number of output routine
      iwork(4) = 6
c     maximum number of time instants related to one
c     control/time segment when state is to be saved
      iwork(5) = 50
c     method of state interpolation
c     0 - none(left one), 1-linear, 2-poly 2rd order with
c     cont. derivative at the beginning, 3-poly 3rd order with cont.
c     derivative everywhere
      iwork(6) = 3
c     optimise what: 0/1-ti, 0/2-ui, 0/4-p
c     all 1+2+4 = 7
      iwork(7) = 3
c     gradients via  0 - adjoint equations,  1 - finite differences
      iwork(8) = 0
c     choice of the optimising strategy
c      0 - standard, 1 - mesh refining, 2 - multirate
      iwork(9) = 0
c     starting number of time intervals for iwork(9)=1,2
      iwork(10) = ntime
c     number of master NLP problems for iwork(9)=1,2
      iwork(11) = 1
c     periodicity
      iwork(12) = 1

c     initial values of the optimised parameters
c     upper, lower bounds
c     control and time
      do i=1,  ntime
         u(1,i) = 0.1707
         ul(1,i) = 0.0d0
         uu(1,i) = 1.00d0
         t(i) = tf/ntime
         tl(i) = 0.01d0
         tu(i) = tf
      end do
c     parameters p_i
```

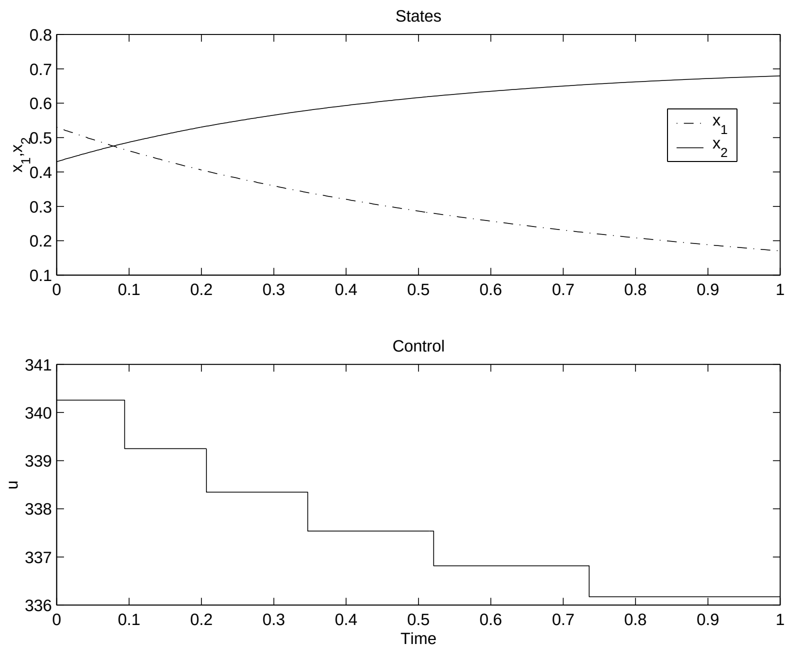

Figure A.1: Optimal trajectories of the batch reactor

```
c     IFAIL : 0 - normal execution, -1: initial trajectory simulation
c     -2: gradients check
      ifail =0
      call DYNO(nsta, ncont, npar, ntime, ncst, ncste, ul, u, uu,
     &      pl, p, pu, tl, t, tu, ista, rwork, nrwork, iwork, niwork,
     &      lwork, nlwork, rpar, ipar, ifail)
      END
```

A.2.4 Results

The program returns the following results

```
min: -0.67941937D+00
ti:        0.72811700D+00  0.90152518D+00  0.11137054D+01  0.13780938D+01
           0.17158681D+01  0.21626905D+01
ui:        0.17417489D+00  0.16100402D+00  0.15001159D+00  0.14076614D+00
           0.13295177D+00  0.12633420D+00
```

and the simulation with recalculated control variable (see eq. (A.4)) produces trajectories shown in Fig. A.1.