SLOVAK UNIVERSITY OF TECHNOLOGY IN BRATISLAVA
FACULTY OF CHEMICAL AND FOOD TECHNOLOGY
INSTITUTE OF INFORMATION ENGINEERING, AUTOMATION AND MATHEMATICS

Dynamic optimization of processes

Diploma thesis

FCHPT-5414-50937

Study Programme:	Automation and Informatization in Chemistry and Food Industry
Study Field:	5.2.14 Automation
Workplace:	Faculty of Chemical and Food Technology
Supervisor:	prof. Ing. Miroslav Fikar, DrSc.
Consultant:	Ing. Radoslav Paulen

Bratislava 2012 **Bc. Lívia Petáková**

Slovak University of Technology in Bratislava
Institute of Information Engineering, Automation and Mathematics

Faculty of Chemical and Food Technology
Academic year: 2011/2012
Reg. No.: FCHPT-5414-50937

S T U
F C H P T

DIPLOMA THESIS TOPIC

Student:	**Bc. Lívia Petáková**
Student's ID:	50937
Study programme:	Automation and Informatization in Chemistry and Food Industry
Study field:	5.2.14 Automation
Thesis supervisor:	prof. Ing. Miroslav Fikar, DrSc.
Consultant:	Ing. Radoslav Paulen
Workplace:	ÚIAM FCHPT STU v Bratislave

Topic: **Dynamic optimization of processes**

Specification of Assignment:

This work deals with dynamic systems optimal control design. In first stage various dynamic optimization methods are studied. Second stage is devoted to application of selected methods for computing several optimal control examples. One of the examples includes a study of copolymerization reaction of styrene and alphamethylstyrene. Problem of dynamic optimization is to compute dynamic temperature profile of emulsion polymerization reaction to ensure time-optimal production of required amount of polymer and desired quality of this product.

Length of thesis: 60

Selected bibliography:

1. FIKAR, M. *Dynamická optimalizácia procesov.* 2007. 161 s. ISBN 978-80-89316-08-3.
2. MIKLEŠ, J. – FIKAR, M. *Process Modelling, Identification, and Control.* Berlin Heidelberg: Springer Berlin Heidelberg New York, 2007. 480 s. ISBN 978-3-540-71969-4.

Assignment procedure from: 13. 02. 2012

Date of thesis submission: 19. 05. 2012

L.S.

Bc. Lívia Petáková
Student

prof. Ing. Miroslav Fikar, DrSc.
Head of office

prof. Ing. Miroslav Fikar, DrSc.
Study programme supervisor

First let me thank my supervising professor Miroslav Fikar, who inspired and helped me, not only during the diploma thesis elaboration, but also during my entire study and who gave me the opportunity to become an Erasmus student at a foreign university in France. My other thanks belongs to my consultant Radoslav Paulen, who helped me and supported me. His priceless advices significantly facilitated the development of this work. My last thanks is devoted to professor M. Abderazzak Latifi, who took care of me and supported me during my work in France.

Bratislava, 2012
Lívia Petáková

Abstrakt

Predkladaná práca sa zaoberá problematikou optimálneho riadenia procesov pri otvorenej slučke. Prvá časť práce sa zaoberá definíciou optimálneho riadenia procesov a definujú sa všeobecné prístupy k riešeniu problémov dynamickej optimalizácie. Druhá praktická časť práce je zameraná na riešenie konkrétnych úloh optimálneho riadenia. Pri riešení jednotlivých príkladov boli použité dve numerické metódy - ortogonálna kolokácia na konečných prvkoch a parametrizácia vektora riadenia. Optimalizačné úlohy boli vyriešené pomocou MATLABu a gPROMSu.

Kľúčové slová: Optimálne riadenie procesov pri otvorenej slučke, Dynamická optimalizácia procesov, Ortogonálna kolokácia na konečných prvkoch, Parametrizácia vektora riadenia

Abstract

The present work deals with optimal control of processes in an open-loop. The first part deals with optimal process control and several problem solving approaches. The second one, practical part, focuses on concrete problem solving, where two numerical methods were used - orthogonal collocation on finite elements and control vector parametrization. Optimization problems were solved using MATLAB and gPROMS.

Keywords: Optimal control of processes in open-loop, Dynamic optimization of processes, Orthogonal collocation on finite elements, Control vector parametrization

Contents

List of Figures

Nomenclature

a_s	surface area occupied by an emulsifier molecule [dm^2]
c_p	product concentration [$\mathrm{mol\,L}^{-1}$]
c_s	medium concentration [$\mathrm{mol\,L}^{-1}$]
F	volumetric flow [$\mathrm{L\,min}^{-1}$]
f	initiator efficiency
f_{MS}	α–methylstyrene molar fraction in the initial load
J	objective function
k_d	rate constant for initiator decomposition [s^{-1}]
K_i	inhibition coefficient [$\mathrm{mol\,L}^{-1}$]
K_m	Michaelis-Menten constant [$\mathrm{mol\,L}^{-1}$]
k_p	rate constant for propagation [$\mathrm{dm}^3\mathrm{mol}^{-1}\mathrm{s}^{-1}$]
k_{cm}	ate constant for initiator radical entry into micelles [$\mathrm{dm}^3\mathrm{micelle}^{-1}\mathrm{s}^{-1}$]
k_{cp}	rate constant for initiator radical entry into particles [$\mathrm{dm}^3\mathrm{part}^{-1}\mathrm{s}^{-1}$]
k_{trM}	rate constant for transfer to monomer [$\mathrm{dm}^3\mathrm{mol}^{-1}\mathrm{s}^{-1}$]
L	kinetic chain length [$\mathrm{g\,mol}^{-1}$]
$L_j^{(k)}$	transition condition of switching from mode S_k to S_j
M	global monomer concentration [$\mathrm{mol\,dm}^{-3}$]
m	number of micelles per unit volume [$\mathrm{micelle\,dm}^{-3}$]

M_{M} monomer molecular weight [$\mathrm{g\,mol^{-1}}$]

M_{p} monomer concentration in particles [$\mathrm{mol\,dm^{-3}}$]

$M_{\mathrm{p}c}$ critical monomer concentration in particles [$\mathrm{mol\,dm^{-3}}$]

$\bar{M}_n$ number–average molecular weight [$\mathrm{g\,mol^{-1}}$]

$\bar{n}$ average number of radicals per particle

$\boldsymbol{u}$ control variable

$\boldsymbol{x}$ state vector

N number of inactive particles per unit volume [$\mathrm{particle\,dm^{-3}}$]

$N^{\bullet}$ number of active particles per unit volume [$\mathrm{particle\,dm^{-3}}$]

N_{p} total number of particles per unit volume [$\mathrm{particle\,dm^{-3}}$]

N_A Avogadro's number [$\mathrm{mol^{-1}}$]

n_s aggregation number of micelles

NI number of intervals

P dead polymer concentration [$\mathrm{mol\,dm^{-3}}$]

$P^{(k)}$ set of all possible descendant modes of mode S_k

r reaction rate [$\mathrm{mol\,L^{-1}min^{-1}}$]

$R^{\bullet}$ initiator radical concentration [$\mathrm{mol\,dm^{-3}}$]

R_a initiator decomposition rate [$\mathrm{mol\,dm^{-3}s^{-1}}$]

R_i initiation rate [$\mathrm{mol\,dm^{-3}s^{-1}}$]

R_n particle formation rate [$\mathrm{mol\,dm^{-3}s^{-1}}$]

R_p polymerization rate [$\mathrm{mol\,dm^{-3}s^{-1}}$]

R_t termination rate [$\mathrm{mol\,dm^{-3}s^{-1}}$]

R_{trM} transfer to monomer rate [$\mathrm{mol\,dm^{-3}s^{-1}}$]

S emulsifier concentration [$\mathrm{mol\,dm^{-3}}$]

S_i ith mode of the process

T reactor temperature [K]

t time [s]

V volume [L]

v_m maximum reaction velocity [$\mathrm{mol\,L^{-1}min^{-1}}$]

$\dot{\boldsymbol{x}}$ vector of state derivatives

P_j number of polymers with chain length j

X monomer conversion

X_c critical monomer conversion

Greek Symbols

ε constant describing the efficiency of the particles relative to the micelles in collecting an initiator radical

ρ_{M} monomer density [$\mathrm{g\,dm^{-3}}$]

ρ_{p} polymer particle density [$\mathrm{g\,dm^{-3}}$]

ρ_{P} polymer density [$\mathrm{g\,dm^{-3}}$]

ω_{P} polymer weight fraction in the particles

Subscripts

0 initial

f final

Superscripts

L lower bound

U upper bound

Chapter 1

Introduction

Finding the optimal solution is something what we have to deal with in everyday life. Each day we are solving problems and trying to find the best solution to this problem. In general, we can say that optimal solution is the best solution to the given problem under the given conditions.

Finding the best solution stands for finding the mathematical solution of an optimization problem. In chemical industry we can often encounter with optimization problems, where our goal is not only simply the production of the required product. Our aim is to produce the maximum amount of this required product, but also to minimize the total costs, to minimize the consumption of raw materials, consumption of energy and other factors. These problems are solved by dynamic optimization. More concretely, dynamic optimization stands for finding such an optimal control profile, which will minimize or maximize the value of the given objective function without violating the constraints. Dynamic optimization is used for finding the solution of optimal process control problems which are described by a set of ordinary differential equations (ODE) or by differential algebraic equations (DAE).

Solving dynamic optimization problems consists of several steps. The first step is the problem definition. In other words, we have to specify what our aim is, what we are going to minimize or maximize, under what conditions and limits. When the problem is well defined and described, the next step is to describe the process as good as possible. Thus, the second step is the process model formation. In general rules, the better is the process description, the better solution we are able to find. The last step consists of using the optimization algorithm for solving the optimization problem.

Recently, there are several approaches for solving dynamic optimization problems. In general they can be divided into analytical and numerical methods. Dynamic programming, Pontryagin's minimum principle and variational calculus belong to analytical methods. Analytical approaches are mainly used for simple problems and often ensure important properties such as global optimality. Numerical methods can be subdivided into direct and indirect

ones. The group of direct numerical methods includes several manners such as sequential or simultaneous ones. The group of indirect numerical methods includes approaches such as boundary condition iteration or control vector iteration. The main difference between direct and indirect ones is that the direct numerical methods consider the discretization of dynamic variables which are directly involved in the problem, while the indirect ones are discretizing the optimality conditions.

The presented work is divided into two parts - theoretical and application one. In theoretical part, we define the dynamic optimization problem in general and we explain dynamic programming, Pontryagin's minimum principle and variational calculus from the group of analytical methods and orthogonal collocation, together with control vector parametrization from the numerical ones. The second part is dedicated to particular examples which are solved by numerical methods.

Part I

Theoretical Part

Chapter 2

Dynamic Optimization

This chapter deals with the general formulation of dynamic optimization problem, also called open-loop optimal control problem, with constraints on state and control variables. Our aim is to define the time-varying forms of systems (Fikar 2007).

First, it is useful to explain the difference between closed-loop and open-loop optimal control (Fig.2.1).

Closed-Loop Optimal Control Function ω is called closed-loop optimal control, if for

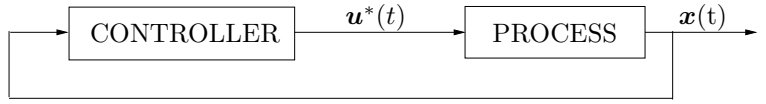
$$\boldsymbol{u}^*(t) = \omega\left(t, \boldsymbol{x}(t)\right) \tag{2.1}$$

we are able to find the optimal control value at time t (Chachuat et al. 2006).

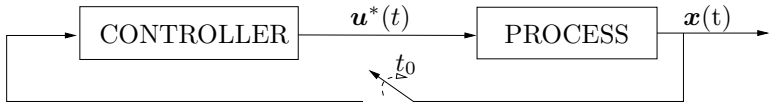
Open-Loop Optimal Control If the optimal control law is characterized as a function of time for a specified initial value as follows

$$\boldsymbol{u}^*(t) = \omega\left(t, x(t_0)\right) \tag{2.2}$$

then it is called open-loop control law.



(a) Closed-Loop Optimal Control



(b) Open-Loop Optimal Control

Figure 2.1: Optimal Control Loops

Notice that an open-loop optimal control is optimal only for one particular initial state, but if we know the optimal control law, we are able to formulate the optimal control history for any initial value (Chachuat et al. 2006).

Solving any problem requires a sufficient problem description which consists of

1. Mathematical description or model of the controlled process
2. Constraints definition
3. Performance criterion.

2.1 Mathematical Description

The definition of process model is the non-trivial part of solving any optimal control problem. Our main objective during modelling a physical system is to derive the simplest mathematical description that adequately predicts the response of this system at all expected inputs (Kirk 1970). Assume the process state variables (or simply states) $x_i,\ (i = 1, \ldots, n)$ and the controlled inputs $u_j,\ (j = 1, \ldots, m)$ at the time t

$$x_1(t), x_2(t), \quad \ldots \quad , x_n(t) \tag{2.3}$$

$$u_1(t), u_2(t), \quad \ldots \quad , u_m(t) \tag{2.4}$$

Afterwards, the system can be described by n first order equations

$$\begin{aligned}
\dot{x}_1(t) &= f_1\left(x_1(t), x_2(t), \ldots, x_n(t), u_1(t), \ldots, u_m(t), t\right) \\
\dot{x}_2(t) &= f_2\left(x_1(t), x_2(t), \ldots, x_n(t), u_1(t), \ldots, u_m(t), t\right) \\
&\vdots \\
\dot{x}_n(t) &= f_n\left(x_1(t), x_2(t), \ldots, x_n(t), u_1(t), \ldots, u_m(t), t\right)
\end{aligned} \tag{2.5}$$

Next, define the vector of state variables and the vector of controlled variables

$$\boldsymbol{x}(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \\ \vdots \\ x_n(t) \end{bmatrix} \tag{2.6}$$

$$\boldsymbol{u}(t) = \begin{bmatrix} u_1(t) \\ u_2(t) \\ \vdots \\ u_m(t) \end{bmatrix} \tag{2.7}$$

Thus, the non-linear and time varying system can be expressed as follows (Kirk 1970)

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t), t) \tag{2.8}$$

here the definition of $\boldsymbol{f}$ results from Eq. 2.5.

2.2 Physical Constraints

While solving an optimization problem, we have to think about some physical constraints which keep the state and control variables within boundaries. For example, if we are deriving a model of a heat exchanger, we must bear in mind that the temperature inside the exchanger can not exceed the metallurgical temperature as the material of the heat exchanger is made of. There are three basic groups of constraints - point, path and isoperimetric constraints. All of them can be expressed in a form of equality or inequality.

1. *Point Constraints* are usually used in optimal control problems as terminal constraints, end-point constraints (point constraints expressed in the final time)

$$l\left(t_f, x\left(t_f\right)\right) \leq 0 \tag{2.9a}$$

$$\acute{l}\left(t_f, x\left(t_f\right)\right) = 0 \tag{2.9b}$$

2. *Isoperimetric Constraints* are constraints containing the integral of a given objective function over the entire time period $[t_0, t_f]$

$$\int_{t_o}^{t_f} h\left(t, \boldsymbol{x}(t), \boldsymbol{u}(t)\right) \mathrm{d}t \leq C \tag{2.10}$$

   here C stands for real number (Chachuat et al. 2006).

3. *Path Constraints* are the most commonly used in optimal control. Path constraints can be characterized by ambivalent functions of the control and state variables and over the time $[t_0, t_f]$ too (Chachuat et al. 2006).

$$\boldsymbol{g}\left(t, \boldsymbol{x}(t), \boldsymbol{u}(t)\right) \leq 0, \qquad \forall t \in [t_0, t_f] \tag{2.11}$$

Next, it will be useful to define the term of feasible control: an admissible control $\bar{\boldsymbol{u}}(\cdot) \in U[t_0, t_f]$ is called feasible, if

(i) the response $\bar{\boldsymbol{x}}(\cdot, x_0, \boldsymbol{u}(\cdot))$ is characterized on the entire time $t_0 \leq t \leq t_f$ and if

(ii) $\bar{\boldsymbol{u}}$ and $\bar{\boldsymbol{x}}(\cdot, x_0, \boldsymbol{u}(\cdot))$ are satisfying all physical constraints.

Then the pair $(\bar{\boldsymbol{u}}(\cdot), \bar{\boldsymbol{x}}(\cdot))$ are said to be feasible pair and the set of feasible controls $\Omega[t_0, t_f]$ is characterized as follows (Chachuat et al. 2006)

$$\Omega\left[t_0, t_f\right] = \{\boldsymbol{u}(\cdot) \in U[t_0, t_f] : \boldsymbol{u}(\cdot) \text{ is feasible}\} \tag{2.12}$$

2.3 Performance Criterion

Performance criterion, or also called objective function, is a predictive function, which assigns a real number to the function. Generally, it can be written in three basic forms (Čižniar 2005)

- Bolza form

$$J = G(\boldsymbol{x}(t_f), t_f) + \int_{t_0}^{t_f} F(\boldsymbol{x}(t), \boldsymbol{u}(t), t)\mathrm{d}t \tag{2.13}$$

- Lagrange form

$$J = \int_{t_0}^{t_f} F(\boldsymbol{x}(t), \boldsymbol{u}(t), t)\mathrm{d}t \tag{2.14}$$

- Mayer form

$$J = G(\boldsymbol{x}(t_f), t_f) \tag{2.15}$$

where J denotes optimization criterion, $G(\cdot)$ stands for a part of objective function evaluated at final conditions, $F(\cdot)$ stands for a part of the objective function evaluated over the entire time period, $\boldsymbol{x}(t)$ denotes state variable profile vector, and $\boldsymbol{u}(t)$ means control profile vector.

2.4 Necessary Conditions for the Extreme

The aim of dynamic optimization is to find an optimal control $\boldsymbol{u}(t), t \in [t_0, t_f]$ which minimizes the objective function Eq. 2.13. In practice, we encounter problems on which different requirements and constraints are set. Some of these cases are shown on Fig. 2.2. Fig. 2.2(a) demonstrates the case of fixed final time t_f and free final state $x(t_f)$. Case of fixed final time and state is shown on Fig. 2.2(b) while Fig. 2.2(c) demonstrates the case of free final time and state. The last Fig. 2.2(d) presents the case of free final time t_f and fixed final state $x(t_f)$.

We assume that $\boldsymbol{u}^\star(t)$ is the optimal control that exists. For each control $\boldsymbol{u}(t)$ we can say that

$$J[\boldsymbol{u}(t)] \geq J[\boldsymbol{u}^\star(t)] \tag{2.16}$$

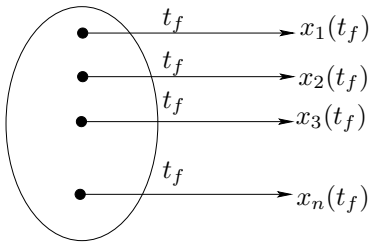
Eq. 2.16 rules in general. First, we derive the necessary optimality condition so that the control variable is changed according to optimal control $\boldsymbol{u}^\star$. If $\boldsymbol{u}^\star$ is optimal control then $\boldsymbol{x}^\star$ is the response to this system, thus we can write

$$\boldsymbol{x}(t) = \boldsymbol{x}^\star(t) + \delta\boldsymbol{x}(t) \tag{2.17}$$

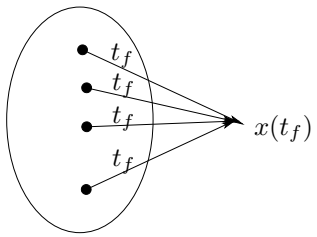
$$\boldsymbol{u}(t) = \boldsymbol{u}^\star(t) + \delta\boldsymbol{u}(t) \tag{2.18}$$

where $\delta\boldsymbol{x}(t)$ is the response to the control variation $\delta\boldsymbol{u}(t)$ (Fikar 2007). After approximation using the Taylor's series in the optimum vicinity we can say that

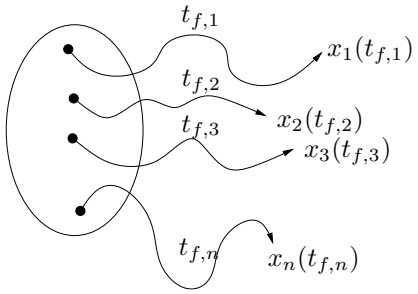
$$\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) = \boldsymbol{f}(\boldsymbol{x}^\star, \boldsymbol{u}^\star) + \left(\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}\right)^\star \delta\boldsymbol{x} + \left(\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}}\right)^\star \delta\boldsymbol{u} \tag{2.19}$$



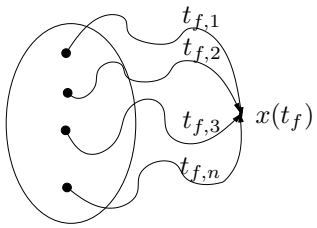
(a) Fixed t_f, free $x(t_f)$.



(b) Fixed t_f, fixed $x(t_f)$.



(c) Free t_f, free $x(t_f)$.



(d) Free t_f, fixed $x(t_f)$.

Figure 2.2: Different possible cases of final time and state.

Matrices $(\cdot)^\star$ are Jacobi matrices for the optimal trajectories of $\boldsymbol{x}^\star$ and $\boldsymbol{u}^\star$ (Fikar 2007). If we consider only the linear elements and if Eq. 2.20 is valid

$$\begin{aligned}\delta\left(\frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}t}\right) =& \frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}t} - \frac{\mathrm{d}\boldsymbol{x}^\star}{\mathrm{d}t} = \\ =& \frac{\mathrm{d}\boldsymbol{x}^\star}{\mathrm{d}t} + \frac{\mathrm{d}(\delta\boldsymbol{x})}{\mathrm{d}t} - \frac{\mathrm{d}\boldsymbol{x}^\star}{\mathrm{d}t} = \\ =& \frac{\mathrm{d}(\delta\boldsymbol{x})}{\mathrm{d}t}\end{aligned} \tag{2.20}$$

then the Eq. 2.19 can be rearranged in the following way

$$\frac{\mathrm{d}(\delta\boldsymbol{x})}{\mathrm{d}t} = \left(\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}\right)\delta\boldsymbol{x} + \left(\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}}\right)\delta\boldsymbol{u} \tag{2.21}$$

Functional $J(\boldsymbol{u})$ reaches the absolute minimum for the function $\boldsymbol{u}^\star = \boldsymbol{u}^\star(t)$ from the class of permissible functions, if for any feasible function $\boldsymbol{u}(t)$ the inequality Eq. 2.16 is valid. The necessary condition for extreme will be (Fikar 2007)

$$\delta J = 0 \tag{2.22}$$

We can express the variation of the objective function Eq. 2.13 as follows

$$\delta J = \left(\frac{\partial G}{\partial x(t_f)}\right)^T \delta\boldsymbol{x}(t_f) + \int_{t_0}^{t_f}\left[\left(\frac{\partial F}{\partial \boldsymbol{x}}\right)^T \delta\boldsymbol{x} + \left(\frac{\partial F}{\partial \boldsymbol{u}}\right)^T \delta\boldsymbol{u}\right]\mathrm{d}t \tag{2.23}$$

Eq. 2.23 is the case of fixed final time t_f and free final state $x(t_f)$ (Fig. 2.2(a)). If we define the vector of adjoint variables as $\boldsymbol{\lambda}(t)$ then we can rearrange the Eq. 2.21 in the following way

$$\boldsymbol{\lambda}^T \frac{\mathrm{d}(\delta \boldsymbol{x})}{\mathrm{d}t} = \boldsymbol{\lambda}^T \left(\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}} \right) \delta \boldsymbol{x} + \boldsymbol{\lambda}^T \left(\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}} \right) \delta \boldsymbol{u} \tag{2.24}$$

Next, integrate the Eq. 2.24 on the time interval from $t = t_0$ to $t = t_f$ in the following way

$$\int_{t_0}^{t_f} \left[\boldsymbol{\lambda}^T \frac{\mathrm{d}(\lambda \boldsymbol{x})}{\mathrm{d}t} - \boldsymbol{\lambda}^T \left(\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}} \right) \delta \boldsymbol{x} - \boldsymbol{\lambda}^T \left(\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}} \right) \delta \boldsymbol{u} \right] \mathrm{d}t = 0 \tag{2.25}$$

The summation of the Eqs. 2.23 and 2.25 is expressed as in Fikar (2007)

$$\begin{aligned} \delta J = & \left(\frac{\partial G}{\partial x(t_f)} \right)^T \delta x(t_f) - \int_{t_0}^{t_f} \boldsymbol{\lambda}^T \frac{\mathrm{d}(\delta \boldsymbol{x})}{\mathrm{d}t} \mathrm{d}t \\ & + \int_{t_0}^{t_f} \left(\left[\left(\frac{\partial F}{\partial \boldsymbol{x}} \right)^T + \boldsymbol{\lambda}^T \left(\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}} \right) \right] \boldsymbol{\lambda} \boldsymbol{x} + \left[\left(\frac{\partial F}{\partial \boldsymbol{u}} \right)^T + \boldsymbol{\lambda}^T \left(\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}} \right) \right] \boldsymbol{\lambda} \boldsymbol{u} \right) \mathrm{d}t \end{aligned} \tag{2.26}$$

If the Hamiltonian function is defined as

$$H = F + \boldsymbol{\lambda}^T \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) \tag{2.27}$$

and if the vector of adjoint variables $\boldsymbol{\lambda}(t)$ satisfies the differential equation

$$\frac{\mathrm{d}\boldsymbol{\lambda}}{\mathrm{d}t} = -\frac{\partial H}{\partial \boldsymbol{x}} \tag{2.28}$$

Then the necessary extremum condition will be fulfilled if the following equation holds for an arbitrary variation $\delta \boldsymbol{u}(t)$

$$\frac{\partial H}{\partial \boldsymbol{u}} = 0 \tag{2.29}$$

From Eqs.2.27 and 2.28 accrue the following findings (Fikar 2007)

$$\frac{\partial H}{\partial \boldsymbol{\lambda}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) \tag{2.30}$$

$$\frac{\partial H}{\partial \boldsymbol{\lambda}} = \frac{\mathrm{d}x}{\mathrm{d}t} \tag{2.31}$$

$$\frac{\partial H}{\partial \boldsymbol{x}} = \frac{\partial F}{\partial \boldsymbol{x}} + \left(\boldsymbol{\lambda}^T \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}} \right)^T \tag{2.32}$$

$$\frac{\mathrm{d}\boldsymbol{\lambda}}{\mathrm{d}t} = -\frac{\partial F}{\partial \boldsymbol{x}} - \left(\boldsymbol{\lambda}^T \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}} \right)^T \tag{2.33}$$

$$\frac{\mathrm{d}\boldsymbol{\lambda}}{\mathrm{d}t} = -\frac{\partial H}{\partial \boldsymbol{x}} \tag{2.34}$$

Next, derive the Hamiltonian function as follows

$$\frac{\mathrm{d}H}{\mathrm{d}t} = \left(\frac{\partial H}{\partial \boldsymbol{x}} \right)^T \frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}t} + \left(\frac{\partial H}{\partial \boldsymbol{u}} \right)^T \frac{\mathrm{d}\boldsymbol{u}}{\mathrm{d}t} + \left(\frac{\partial H}{\partial \boldsymbol{\lambda}} \right)^T \frac{\mathrm{d}\boldsymbol{\lambda}}{\mathrm{d}t} \tag{2.35}$$

The following equation results from Eqs. 2.28 and 2.31

$$\left(\frac{\partial H}{\partial \boldsymbol{x}}\right)^T \frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}t} + \left(\frac{\partial H}{\partial \boldsymbol{\lambda}}\right)^T \frac{\mathrm{d}\boldsymbol{\lambda}}{\mathrm{d}t} = 0 \tag{2.36}$$

Because the Eq. 2.29 have the value of zero, the righthand side of Eq. 2.36 will be also equal to zero

$$\frac{\mathrm{d}H}{\mathrm{d}t} = 0 \tag{2.37}$$

is valid for an unbounded control or for a control which never reaches the boundaries. From Eq. 2.37 results that the function H remains constant during the optimal response.

Chapter 3

Methods of Dynamic Optimization

This chapter deals with several approaches used for solving dynamic optimization problems. These approaches can be separated into two basic groups - analytical and numerical methods.

3.1 Analytical Methods

There are several analytical methods but the most known and used ones are (Hirmajer 2007):

- Dynamic programming
- Pontryagin's principle of minimum
- Variational calculus

3.1.1 Dynamic Programming

Dynamic programming is based on the Bellman's principle of optimality, which says (Fikar 2007): the optimal path depends on the initial conditions and of the goal and not on the path the goal was achieved with. Assume an optimal trajectory and a point $\boldsymbol{x}(t^\circ)$ which subdivides this trajectory into two parts as shown in Fig. 3.1. The optimality principle says, that the trajectory between $\boldsymbol{x}(t^\circ)$ and $\boldsymbol{x}(t_f)$ is an optimal trajectory too. It means that the trajectory from initial state $\boldsymbol{x}(t^\circ)$ to final state $\boldsymbol{x}(t_f)$ is optimal, regardless of how it reached this initial state.

Assume the following optimal control problem expressed in Bolza form (Čižniar 2005)

$$J = G(\boldsymbol{x}(t_f), t_f) + \int_{t_0}^{t_f} F(\boldsymbol{x}(t), \boldsymbol{u}(t), t)\mathrm{d}t \tag{3.1a}$$

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t), t) \qquad \boldsymbol{x}(t_0) = \boldsymbol{x}_0 \tag{3.1b}$$

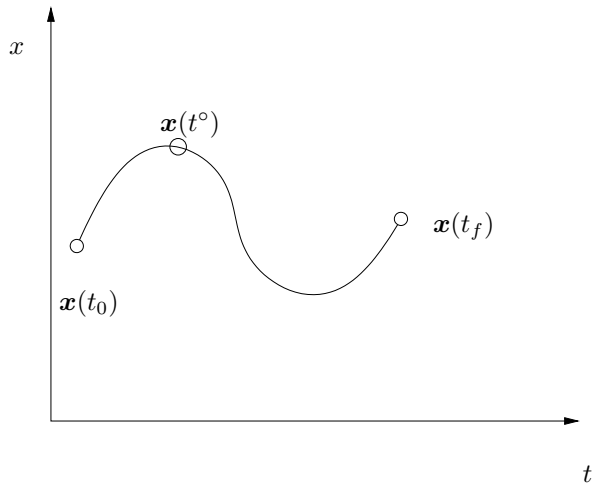


Figure 3.1: Bellman's principle of optimality

Further we suppose, that this problem has its solution. Next, define a function called Bellman's function

$$\nu(\boldsymbol{x}(t),t) = \min_{\boldsymbol{u}(t)} \left[G(\boldsymbol{x}(t_f),t_f) + \int_{t_0}^{t_f} F(\boldsymbol{x}(t),\boldsymbol{u}(t),\tau)d\tau \right] \tag{3.2}$$

After differentiating Eq. 3.2 we obtain Bellman's partial differential equation in the following form

$$-\frac{\partial \nu}{\partial t} = \min_{\boldsymbol{u}(t)} \left[F(\boldsymbol{x},\boldsymbol{u},t) + \left(\frac{\partial \nu}{\partial \boldsymbol{x}}\right)^T \boldsymbol{f}(\boldsymbol{x},\boldsymbol{u},t) \right] \tag{3.3}$$

Which has to fulfill the following boundary condition

$$\nu(\boldsymbol{x}_f,t_f) = G(\boldsymbol{x}_f,t_f) \tag{3.4}$$

Eqs. 3.3 and 3.4 define the necessary conditions for minimum of the optimization problem (Čižniar 2005). After replacing $\boldsymbol{u}(t)$ with $\boldsymbol{u}^*(t)$ in Bellman's partial differential equation, we will obtain the following partial differential equation, also known as Hamilton-Jacobi-Bellman's equation

$$-\frac{\partial \nu}{\partial t} = F(\boldsymbol{x},\boldsymbol{u}^*,t) + \left(\frac{\partial \nu}{\partial \boldsymbol{x}}\right)^T \boldsymbol{f}(\boldsymbol{x},\boldsymbol{u},t) \tag{3.5}$$

Next, define Hamiltonian function in the following form

$$H\left(\boldsymbol{x},\boldsymbol{u},\frac{\partial \nu}{\partial \boldsymbol{x}},t\right) = F(\boldsymbol{x},\boldsymbol{u},t) + \left(\frac{\partial \nu}{\partial \boldsymbol{x}}\right)^T \boldsymbol{f}(\boldsymbol{x},\boldsymbol{u},t) \tag{3.6}$$

By replacing the Eq. 3.6 into Eq. 3.3 we obtain the modified form of the Bellman's partial differential equation

$$-\frac{\partial \nu}{\partial t} = \min_{\boldsymbol{u}(t)} H\left(\boldsymbol{x},\boldsymbol{u},\frac{\partial \nu}{\partial \boldsymbol{x}},t\right) \tag{3.7}$$

3.1.2 Potryagin's Principle of Minimum

Pontryagin's principle of minimum is similar to dynamic programming. This analytical method can be used for solving difficult problems with constraints to the state and control variables. Assume the same optimization problem as mentioned above (Eq. 3.1) and replace $\frac{\partial \nu}{\partial \boldsymbol{x}}$ by $\boldsymbol{\lambda}(t)$ adjoint variable. Then the Hamiltonian function takes the form as follows

$$H(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{p}, t) = F(\boldsymbol{x}, \boldsymbol{u}, t) + \boldsymbol{\lambda}^T \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, t) \tag{3.8}$$

After a substitution in Bellman's partial differential equation (Eq. 3.7) it will take the following form (Čižniar 2005)

$$-\frac{\partial \nu}{\partial t} = \min_{\boldsymbol{u}(t)} H(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\lambda}, t) \tag{3.9}$$

If we differentiate the left- and the right-hand side of $\boldsymbol{\lambda}(t) = \frac{\partial \nu}{\partial \boldsymbol{x}}$ we will obtain the following equations

$$-\frac{\partial^2 \nu}{\partial \boldsymbol{x} \partial t} = \frac{\partial H}{\partial \boldsymbol{x}} + \frac{\partial^2 \nu}{\partial \boldsymbol{x}^2} \frac{\partial H}{\partial \boldsymbol{\lambda}} \tag{3.10a}$$

$$\dot{\boldsymbol{\lambda}} = \frac{\partial^2 \nu}{\partial \boldsymbol{x}^2} \dot{\boldsymbol{x}} + \frac{\partial^2 \nu}{\partial t \partial \boldsymbol{x}} \tag{3.10b}$$

Assume the following canonical differential equations (Čižniar 2005)

$$\dot{\boldsymbol{x}} = \frac{\partial H}{\partial \boldsymbol{\lambda}} \tag{3.11}$$

$$\dot{\boldsymbol{\lambda}} = -\frac{\partial H}{\partial \boldsymbol{x}} \tag{3.12}$$

Thus, the necessary conditions according to Potryagin's principle of minimum are (Čižniar 2005)

- Control variable optimality conditions

$$0 = \frac{\partial H}{\partial \boldsymbol{u}} \qquad \forall t \in [t_0, t_f] \tag{3.13}$$

- Adjoint variables

$$\dot{\boldsymbol{\lambda}} = -\frac{\partial H}{\partial \boldsymbol{x}} \qquad \forall t \in [t_0, t_f] \tag{3.14}$$

- Adjoint variables final conditions

$$\boldsymbol{\lambda}(t_f) = \left.\frac{\partial G}{\partial \boldsymbol{x}}\right|_{t_f} \tag{3.15}$$

3.1.3 Variational Calculus

Variational calculus is based on the Bellman's partial differential equation (Eq. 3.3). First, we define variational calculus problem as follows

$$\frac{\partial \boldsymbol{\Gamma}}{\partial \boldsymbol{x}} - \frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial \boldsymbol{\Gamma}}{\partial \dot{\boldsymbol{x}}}\right) = 0 \tag{3.16}$$

where the Lagrange function $\boldsymbol{\Gamma}$ is defined as follows

$$\boldsymbol{\Gamma}(\boldsymbol{x}, \dot{\boldsymbol{x}}, \boldsymbol{u}, \boldsymbol{\lambda}, t) = F(\boldsymbol{x}, \boldsymbol{u}, t) + \boldsymbol{\lambda}^T \left[\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, t) - \dot{\boldsymbol{x}}\right] \tag{3.17}$$

Assume the optimal control problem already mentioned above Eq. 3.1. For this problem we are going to express the necessary conditions (Hirmajer 2007)

- Control variable optimality condition

$$\frac{\boldsymbol{\Gamma}}{\boldsymbol{u}} = 0 \qquad \forall t \in [t_0, t_f] \tag{3.18}$$

- Adjoint variables

$$-\frac{\partial \boldsymbol{\Gamma}}{\partial \boldsymbol{x}} = \dot{\boldsymbol{\lambda}} \qquad \forall t \in [t_0, t_f] \tag{3.19}$$

- Adjoint variable final conditions

$$\boldsymbol{\lambda}(t_f) = \left.\frac{\partial G}{\partial \boldsymbol{x}}\right|_{t_f} \tag{3.20}$$

3.2 Numerical Methods

Numerical methods are used in cases, when it is too complicated to find the optimal solution by the analytical ones. Numerical methods can be divided into two main groups, namely direct and indirect methods.

3.2.1 Direct Numerical Methods

Direct numerical methods are based on transformation of the formal infinite dimensional problem into finite dimensional problem of nonlinear programming (NLP). Direct numerical methods can be subdivided into sequential and simultaneous methods (Paulen et al. 2010).

Sequential method, also known as control vector parametrization (CVP), is based on the control trajectory approximation by a function of only few parameters (Čižniar 2005).The former continuous control trajectory is divided into few piece-wise polynomial (usually constant) parts, thus the former infinite dimensional problem is transformed into finite dimensional NLP problem, which can be easily solved by any gradient method (Hirmajer 2007).

Simultaneous method, known as complete parametrization, means total state and control variables discretization using polynomials. Order of these polynomials decide about cardinality of the NLP problem (Čižniar 2005).

3.2.2 Indirect Numerical Methods

The main aim of these methods is to solve the two point boundary value problem (TPBVP) and thus indirectly solve the problem of dynamic optimization.

Boundary condition iteration is an indirect numerical method based on finding the missing boundary conditions $\boldsymbol{\lambda}(t_0)$ by minimizing the errors between the boundary conditions (Ševčík 2005).

Control vector iteration is method based on finding the control trajectory by fulfilling the optimality conditions.

Chapter 4

Orthogonal Collocation

This chapter is dedicated to a numerical method called orthogonal collocation on finite elements. This method is based on complete parametrization of both control and state profiles (Čižniar 2005). Thus, the former control and state trajectories are approximated by linear combination of some basis functions. Our aim is to find optimal control by optimizing the coefficients of these functions (Ševčík 2005).

4.1 Problem Definition

The objective function describing the optimization problem is expressed in Bolza form as follows

$$J\left(\boldsymbol{u}(t), \boldsymbol{p}\right) = G\left(\boldsymbol{x}(t_f), \boldsymbol{p}, t_f\right) + \int_{t_0}^{t_f} F\left(\boldsymbol{x}(t), \boldsymbol{u}(t), t\right) \mathrm{d}t \tag{4.1}$$

Our goal is to minimize the objective function J according to the control vector $\boldsymbol{u}(t)$ and to the vector of parameters $\boldsymbol{p}$.

We assume that the dynamic model is expressed by a set of ordinary differential equations (ODE)

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}\left(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{p}, t\right) \qquad \boldsymbol{x}(t_0) = \boldsymbol{x}_0(\boldsymbol{p}) \tag{4.2}$$

Path constraints, which are limiting this system can be expressed in both equality and inequality form

$$\acute{g}\left(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{p}, t\right) = 0 \tag{4.3a}$$

$$g\left(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{p}, t\right) \leq 0 \tag{4.3b}$$

And the upper ^{U} and the lower ^{L} bounds can be defined as follows (Fikar 2007)

$$\boldsymbol{x}(t)^L \leq \boldsymbol{x}(t) \leq \boldsymbol{x}(t)^U \tag{4.4a}$$

$$\boldsymbol{u}(t)^L \leq \boldsymbol{u}(t) \leq \boldsymbol{u}(t)^U \tag{4.4b}$$

$$\boldsymbol{p}^L \leq \boldsymbol{p} \leq \boldsymbol{p}^U \tag{4.4c}$$

4.2 NLP Formulation

We consider the element $i, (i = 1, \ldots, N_I)$, where N_I stands for number of time elements, with times $t \in [\xi_i, \xi_{i+1}]$ as it is shown on Fig. 4.1. The solution is approximated by Lagrange polynomials ϕ_j and θ_j over element i (Fikar 2007)

$$\boldsymbol{x}_{K+1}(t) = \sum_{j=0}^{K} \boldsymbol{x}_{ij}\phi_j(t), \qquad \phi_j(t) = \prod_{k=0,k\neq j}^{K} \frac{t - t_{ik}}{t_{ij} - t_{ik}} \tag{4.5}$$

$$\boldsymbol{u}_K(t) = \sum_{j=1}^{K} \boldsymbol{u}_{ij}\theta_j(t), \qquad \theta_j(t) = \prod_{k=1,k\neq j}^{K} \frac{t - t_{ik}}{t_{ij} - t_{ik}} \tag{4.6}$$

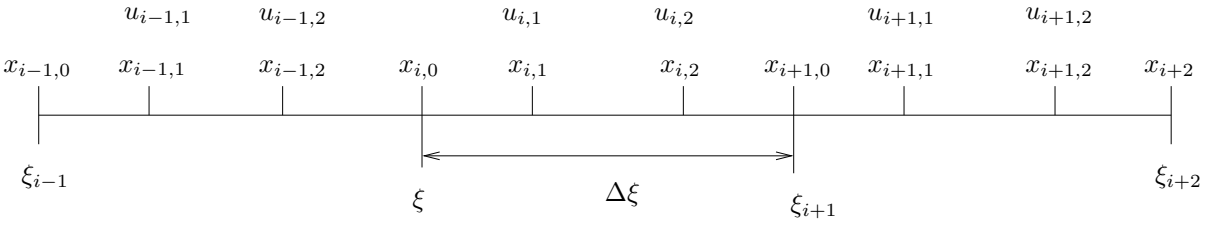


Figure 4.1: Finite element collocation for state and control variables

where K is the number of collocation points, j denotes the order of the collocation point at the given time t_k $(k = 0, \ldots, j)$. Polynomial $\boldsymbol{x}_{K+1}(t)$ is of $(K+1)$th order, while polynomial $\boldsymbol{u}_K(t)$ is of Kth degree. This degree difference results from the fact, that the initial condition of the control variable is dismissed. Points t_{ik} are calculated as Legendre's polynomials roots. A considerable property of Lagrange polynomials is

$$\boldsymbol{x}_{K+1}(t_{ij}) = \boldsymbol{x}_{ij} \tag{4.7}$$

And due to $\phi_k(t_j) = \delta_{kj}$ we are able to directly define the state and control boundaries. Here δ_{kj} is the Kronecker delta, which is in case of $k = j$ equal to one, otherwise it is equal to zero. If we use K collocation points on finite elements (Fig. 4.1) , we can

define basic functions normalized over each element

$$\Delta\xi_i r(t_{ik}) = \sum_{j=0}^{K} \boldsymbol{x}_{i,j}\dot{\phi}(\tau_k) - \Delta\xi_i \boldsymbol{F}(t_{ik}, \boldsymbol{x}_{ik}, \boldsymbol{u}_{ik}) \tag{4.8}$$
$$i = 1, \ldots, N_I \quad j = 0, \ldots, K \quad k = 1, \ldots, K$$

where function $\dot{\phi}_j(\tau_k) = \mathrm{d}\phi_j/\mathrm{d}\tau$ can be calculated off-line and for $t_{ik} = \xi_i + \Delta\xi_i\tau_k$ rules $\tau \in [0, 1]$. The continuity condition will be fulfilled by the following equation

$$\boldsymbol{x}^i_{K+1}(\xi_i) = \boldsymbol{x}^{i-1}_{K+1}(\xi_i), \qquad i = 2, \ldots, N_I \tag{4.9}$$

Or in the form of

$$\boldsymbol{x}_{i0} = \sum_{j=0}^{K} \boldsymbol{x}_{i-1,j}\phi_j(\tau = 1) \tag{4.10}$$
$$i = 2, \ldots, N_I \quad j = 0, \ldots, K$$

Points defined in Eq. 4.10 determine the initial conditions for the next time element (Ševčík 2005). Control profiles are bounded only at collocation points. These control boundaries can be defined in the following way

$$\boldsymbol{u}^L_i \leq \boldsymbol{u}^i_K(\xi_i) \leq \boldsymbol{u}^U_i \qquad i = 1, \ldots, N_I \tag{4.11a}$$
$$\boldsymbol{u}^L_i \leq \boldsymbol{u}^i_K(\xi_{i+1}) \leq \boldsymbol{u}^U_i \qquad i = 1, \ldots, N_I \tag{4.11b}$$

These ones are enforced by polynomial extrapolation of the endpoints of each element as follows

$$\boldsymbol{u}^i_K(\xi_i) = \sum_{j=1}^{K} \boldsymbol{u}_{ij}\theta_j(\tau = 0) \quad i = 1, \ldots, N_I \tag{4.12a}$$

$$\boldsymbol{u}^i_K(\xi_{i+1}) = \sum_{j=1}^{K} \boldsymbol{u}_{ij}\theta_j(\tau = 1) \quad i = 1, \ldots, N_I \tag{4.12b}$$

Complementing these conditions will ensure that the control final values will be in the range of $[\boldsymbol{u}^L_i, \boldsymbol{u}^U_i]$. Then the NLP formulation of Eq. 4.1 has the following form

$$\min_{\boldsymbol{x}_{ij}, \boldsymbol{u}_{ij}, \Delta\xi_i} \left[G\left(\boldsymbol{x}_f, t_f\right) + \sum_{i=1}^{N_I} \sum_{j=1}^{K} \boldsymbol{\omega} F\left(\boldsymbol{x}_{ij}, \boldsymbol{u}_{ij}, \Delta\xi_i\right) \right] \tag{4.13}$$

With the following presumptions

$$\boldsymbol{x}_{10} - \boldsymbol{x}_0 = 0 \tag{4.14}$$

$$\boldsymbol{x}_{i0} - \boldsymbol{x}_{K+1}^{i-1}(\xi_i) = 0 \qquad i = 2, \ldots, N_I \tag{4.15}$$

$$\boldsymbol{x}_f - \boldsymbol{x}_{K+1}^{N_I}(\xi_{N_I+1}) = 0 \tag{4.16}$$

$$\boldsymbol{x}_{ij}^L \leq \boldsymbol{x}_{K+1}(\tau_j) \leq \boldsymbol{x}_{ij}^U \qquad i = 1, \ldots, N_I \quad j = 0, \ldots, K \tag{4.17}$$

$$\boldsymbol{u}_i^L \leq \boldsymbol{u}_K^i(\xi_i) \leq \boldsymbol{u}_i^U \qquad i = 1, \ldots, N_I \tag{4.18}$$

$$\boldsymbol{u}_i^L \leq \boldsymbol{u}_K^i(\xi_{i+1}) \leq \boldsymbol{u}_i^U \qquad i = 1, \ldots, N_I \tag{4.19}$$

$$\boldsymbol{u}_{ij}^L \leq \boldsymbol{u}_K(\tau_j) \leq \boldsymbol{u}_{ij}^U \qquad i = 1, \ldots, N_I \quad j = 1, \ldots, K \tag{4.20}$$

$$\boldsymbol{p}^L \leq \boldsymbol{p} \leq \boldsymbol{p}^U \tag{4.21}$$

$$\Delta\xi_i{}^L \leq \Delta\xi_i \leq \Delta\xi_i^U \qquad i = 1, \ldots, N_I \tag{4.22}$$

$$\Delta\xi_i \boldsymbol{r} = \dot{\boldsymbol{x}}_{k+1}(\tau_j) - \Delta\xi_i \boldsymbol{F}(t_{ik}, \boldsymbol{x}_{ik}, \boldsymbol{u}_{ik}) \qquad i = 1, \ldots, N_I \quad j = 1, \ldots, K \tag{4.23}$$

$$\sum_{i=1}^{N_I} \Delta\xi_i = \xi_{TOTAL} \tag{4.24}$$

$$\boldsymbol{h}(\boldsymbol{x}_{ij}, \boldsymbol{u}_{ij}, t_{ij}, p) = 0 \tag{4.25}$$

$$\boldsymbol{g}(\boldsymbol{x}_{ij}, \boldsymbol{u}_{ij}, t_{ij}, p) \leq 0 \tag{4.26}$$

$$\acute{g}_f(\boldsymbol{x}_f) = 0 \tag{4.27}$$

$$g_f(\boldsymbol{x}_f) \leq 0 \tag{4.28}$$

Here i denotes the time interval, j stands for the collocation point, $\Delta\xi_i$ defines the finite element lengths ($i = 1, \ldots, N_I$), $\boldsymbol{x}_f$ denotes the state variable at the final time t_f, $\boldsymbol{h}$ and $\boldsymbol{g}$ express the equality and inequality constraints and finally $\boldsymbol{x}_{ij}$ and $\boldsymbol{u}_{ij}$ denotes the state and control profiles collocations coefficients (Ševčík 2005).

Chapter 5

Control Vector Parametrization

Control vector parametrization (CVP) is a numerical method based on the replacing the former continuous control trajectory by another one, which can be described by a finite set of parameters (Fikar 2007). In other words, the primary continuous dynamic optimization problem is replaced by a static problem of nonlinear programming. As shown in Fig. 5.1, the former continuous trajectory (Fig. 5.1(a)) is approximated by piece-wise linear control trajectory (Fig. 5.1(b)).

5.1 Problem Formulation

Assume the following set of ordinary differential equations (ODE)

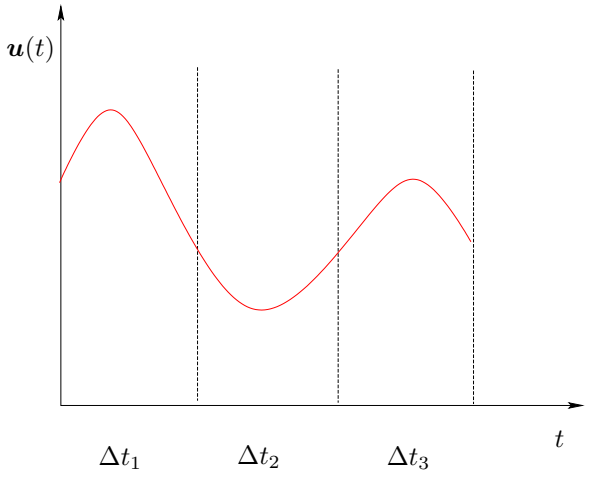
$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{p}, t) \qquad \boldsymbol{x}(t_0) = \boldsymbol{x}_0(\boldsymbol{p}) \tag{5.1}$$

here t stands for the time $t \in [t_0, t_f]$. Further, we assume that initial conditions can be functions of the parameters.We also presume, that the former continuous control trajectory is approximated by piece-wise constant trajectory (Fikar 2007) as it is shown in Fig. 5.1

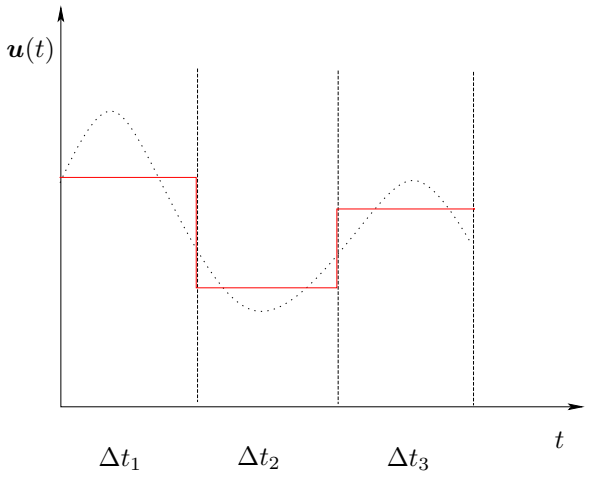
$$\boldsymbol{u}(t) = \boldsymbol{u}_j \qquad t_{j-1} \leq t < t_j \tag{5.2}$$

Define Δt_j as a length of a time interval $\Delta t_j = t_j - t_{j-1}$. The objective function, which has to be minimized, is expressed in the Bolza form (Eq. 2.13) (Fikar 2007). Next, define the upper and lower boundaries

$$\begin{aligned} \boldsymbol{u}_j &\in [\boldsymbol{u}_j^L, \boldsymbol{u}_j^U] \\ \boldsymbol{p} &\in [\boldsymbol{p}^L, \boldsymbol{p}^U] \\ \Delta t_j &\in [\Delta t_j^L, \Delta t_j^U] \end{aligned} \tag{5.3}$$



(a) Continuous control trajectory.



(b) Discretized control trajectory.

Figure 5.1: Illustration of control parameterization (discretization) into piece-wise constant polynomial.

5.2 Methods for Computing Gradients

Gradients used for solving dynamic optimization problems can be calculated by three different deterministic approaches (Hirmajer 2007)

- *Finite differences* is a method based on recurrent system integration with a small change of value of one of the optimized parameters (Hirmajer 2007). Gradients to the objective function are then computed as follows

$$\nabla_{p_i} J = \frac{J(p_1, \ldots, p_i + \Delta p_i, \ldots, p_{n_p}) - J(\boldsymbol{p})}{\Delta p_i} \tag{5.4}$$

  here n_p stands for the number of optimized parameters. Gradients of constraints can be emanated in the same way.The main advantage of this finite differences method is, that it does not require addition of any other differential equations. Beyond that, the entire system has to be integrated n_p–times for every small change of each parameter. This method is often used with another gradient method for gradient accuracy comparison.

- *Adjoint variables* method is based on backward system integration, thus is useful for systems with small number of differential equations and constraints (Petzold et al. 2006).The biggest asset of this method is that the adjoint system has to be integrated only once irrespective of the number of optimized parameters. The size of the adjoint system is equal to $(n_x \times n_c)$, where n_x stands for the number of state variables and n_c denotes the number of constraints. Equations describing the system are integrated separately from the adjoint variables because the system's equations are integrated forward, while the adjoint system is integrated backward in time.

- *Sensitivity equations* method is useful for systems, which consist of a large number of differential equations with small number of optimized parameters, because the sensitivity equations system consists of $(n_x \times n_p)$ equations. The biggest benefit of this method lies in the fact, that the system of sensitivity equations is linear, and the Jacobian matrices are the same for the sensitivity and the original system, so the entire system can be integrated together.

5.2.1 Method of Adjoint Variables

According to Potryagin's minimum principle we are able to define the adjoint variables $\boldsymbol{\lambda}$ as

$$\dot{\boldsymbol{\lambda}} = -\frac{\partial H}{\partial \boldsymbol{x}} \tag{5.5}$$

$$\boldsymbol{\lambda}(t_f) = \left.\frac{\partial G}{\partial \boldsymbol{x}}\right|_{t=t_f} \tag{5.6}$$

here Hamiltonian function H is defined in Eq. 3.8. If we take into account the rest of the optimality conditions (Hull 2003), then the gradients to the objective function can be expressed as

$$\frac{\partial J}{\partial t_f} = H(t_f) + \frac{\partial G}{\partial t_f} \tag{5.7}$$

$$\frac{\partial J}{\partial t_j} = H(t_j^-) - H(t_j^+) \qquad j = 1, \ldots, N_I - 1 \tag{5.8}$$

$$\frac{\partial J}{\partial \boldsymbol{u}_j} = J_{\boldsymbol{u}}(t_{j-1}) - J_{\boldsymbol{u}}(t_j) \qquad j = 1, \ldots, N_I - 1 \tag{5.9}$$

$$\frac{\partial J}{\partial \boldsymbol{p}^T} = \frac{\partial G}{\partial \boldsymbol{p}^T} + J_{\boldsymbol{p}}(t_0) + \boldsymbol{\lambda}_{t_0}^T \frac{\partial \boldsymbol{x}_0}{\partial \boldsymbol{p}^T} \tag{5.10}$$

and also

$$\dot{J}_{\boldsymbol{u}} = \frac{\partial H}{\partial \boldsymbol{u}^T} \qquad J_{\boldsymbol{u}}(t_0) = 0 \tag{5.11}$$

$$\dot{J}_{\boldsymbol{p}} = \frac{\partial H}{\partial \boldsymbol{p}^T} \qquad J_{\boldsymbol{p}}(t_0) \tag{5.12}$$

5.2.2 Method of Sensitivity Equations

Sensitivity functions are defined as partial derivatives of the variable system with regard to the parameters (Feehery 1998). Thus the sensitivity coefficients can be expressed as

$$\boldsymbol{s}_i = \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{p}} \qquad \boldsymbol{s}_i = 0 \tag{5.13}$$

where $i = 1, \ldots, n_p$. Or after differentiation

$$\dot{\boldsymbol{s}}_i = \frac{\partial \boldsymbol{s}_i}{\partial t} = \frac{\partial}{\partial t}\left(\frac{\partial \boldsymbol{x}}{\partial \boldsymbol{p}}\right) = \frac{\partial}{\partial \boldsymbol{p}}\left(\frac{\partial \boldsymbol{x}}{\partial t}\right) = \frac{\partial \dot{\boldsymbol{x}}}{\partial \boldsymbol{p}} \tag{5.14}$$

Then the gradients to the objective function can be characterized as the partial derivatives of the objective function, with respect to the optimized parameters

$$\nabla_{\boldsymbol{p}} J = \left.\frac{\partial G}{\partial \boldsymbol{x}}\right|_{t_f} \boldsymbol{s}_x(t_f) + \frac{\partial G}{\partial t_f}\frac{\partial t_f}{\partial \boldsymbol{p}} + \int_{t_0}^{t_f} \left(\frac{\partial F}{\partial \boldsymbol{x}}\boldsymbol{s}_x + \frac{\partial F}{\partial \boldsymbol{u}}\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{p}}\right) \mathrm{d}t \tag{5.15}$$

Gradients of constraints can be derived in the same way.

Chapter 6

Parametric Sensitivities for Hybrid Continuous Systems

This chapter copes with sensitivity equations for hybrid systems. The main difference between simple systems as mentioned in previous chapters and hybrid systems is that the dynamic properties of a simple system are not changing, while properties of an hybrid system are changing according to the phase in which they are situated. Parametric sensitivities are used to observe the influence of parameters variance of the model on its solution (Feehery 1998).

6.1 Mathematical Model of Hybrid Systems

Assume a process characterized by the following set of differential equations

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{p}, t) \qquad \forall t \in [t_0, t_f] \tag{6.1}$$

where the function $\boldsymbol{f}$ belongs to $\mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_p} \times [t_0, t_f] \to \mathbb{R}^{n_x}$ and initial conditions $\boldsymbol{x}(t_0, \boldsymbol{p}) = x_0(\boldsymbol{p})$ are such that $x_0 : \mathbb{R}^{n_p} \to \mathbb{R}^{n_x}$. We can rewrite Eq. 6.1 into following autonomous form

$$\dot{\boldsymbol{x}} = f\left(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{p}\right) \qquad \forall t \in [t_0, t_f] \tag{6.2}$$

Next, study a system defined by a phase-space $S = \bigcup_{k=1}^{n_k} S_k$ where every single mode S_k is identified by

1. List of variables: $\{\dot{\boldsymbol{x}}^{(k)}, \boldsymbol{x}^{(k)}, \boldsymbol{u}^{(k)}, \boldsymbol{p}, t\}$, where $\boldsymbol{x}^{(k)} \in \mathbb{R}^{n_x^{(k)}}$ represents differential state-variables, $\boldsymbol{u}^{(k)} \in \mathbb{R}^{n_u^{(k)}}$ represents the controls, $\boldsymbol{p} \in \mathbb{R}^{n_p}$ stands for the time-invariant parameter and t denotes the time.

2. List of equations: $\boldsymbol{f}^{(k)}(\dot{\boldsymbol{x}}^{(k)}, \boldsymbol{x}^{(k)}, \boldsymbol{u}^{(k)}, \boldsymbol{p}, t) = 0$, for $\boldsymbol{f}^{(k)} : \mathbb{R}^{n_x{}^{(k)}} \times \mathbb{R}^{n_x{}^{(k)}} \times \mathbb{R}^{n_u{}^{(k)}} \times \mathbb{R}^{n_p} \times \mathbb{R} \to \mathbb{R}^{n_x{}^{(k)}}$. Parameters $\boldsymbol{p}$ determined in mode S_k must be joined with the following initial conditions determining the system change during $[t_0{}^{(k)}, t_f{}^{(k)}]$: $T_k(\dot{\boldsymbol{x}}^{(k)}, \boldsymbol{x}^{(k)}, \boldsymbol{u}^{(k)}, \boldsymbol{p}, t) = 0$ at $t = t_0{}^{(k)}$.

3. List of transitions $J^{(k)}$ from mode S_k to mode S_j. These obtainable transitions are characterized by

   (a) Transition conditions: $L_j^{(k)}(\dot{\boldsymbol{x}}^{(k)}, \boldsymbol{x}^{(k)}, \boldsymbol{u}^{(k)}, \dot{\boldsymbol{x}}^{(j)}, \boldsymbol{x}^{(j)}, \boldsymbol{u}^{(j)}, \boldsymbol{p}, t), j \in J^{(k)}$ ruling the switching time from mode k to mode j.

   (b) Transition functions: $T_j^{(k)}(\dot{\boldsymbol{x}}^{(k)}, \boldsymbol{x}^{(k)}, \boldsymbol{u}^{(k)}, \dot{\boldsymbol{x}}^{(j)}, \boldsymbol{x}^{(j)}, \boldsymbol{u}^{(j)}, \boldsymbol{p}, t)$ are pointing out the possible discontinuity in model. Different case represent the initial conditions $T_0^{(1)}$ for the 1^{th} mode:

$$T_1^{(0)}(\dot{\boldsymbol{x}}^{(1)}, \boldsymbol{x}^{(1)}, \boldsymbol{u}^{(1)}, \boldsymbol{p}, t_0) = 0 \Rightarrow x^{(1)}(t_0^{(1)}) = x_0^{(1)} \tag{6.3}$$

6.2 Sensitivities of Hybrid Systems

Sensitivity functions are defined as partial derivatives of the variable system with regard to the parameters (Feehery 1998). Assume the following system with independent variables $\boldsymbol{p}$ and t

$$\begin{bmatrix} \dfrac{\partial \boldsymbol{f}^{(k)}}{\partial \dot{\boldsymbol{x}}^{(k)}} & \dfrac{\partial \boldsymbol{f}^{(k)}}{\partial \boldsymbol{x}^{(k)}} \end{bmatrix} \begin{bmatrix} \frac{\partial \dot{\boldsymbol{x}}^{(k)}}{\partial \boldsymbol{p}} & \frac{\partial \dot{\boldsymbol{x}}^{(k)}}{\partial t} \\ \frac{\partial \boldsymbol{x}^{(k)}}{\partial \boldsymbol{p}} & \frac{\partial \boldsymbol{x}^{(k)}}{\partial t} \end{bmatrix} = - \begin{bmatrix} \dfrac{\partial \boldsymbol{f}^{(k)}}{\partial \boldsymbol{u}^{(k)}} \dfrac{\partial \boldsymbol{u}^{(k)}}{\partial \boldsymbol{p}} + \dfrac{\partial \boldsymbol{f}^{(k)}}{\partial \boldsymbol{p}} & \dfrac{\partial \boldsymbol{f}^{(k)}}{\partial \boldsymbol{u}^{(k)}} \dfrac{\partial \boldsymbol{u}^{(k)}}{\partial t} + \dfrac{\partial \boldsymbol{f}^{(k)}}{\partial t} \end{bmatrix} \tag{6.4}$$

Now, we are able to define sensitivities as follows

$$\boldsymbol{s}_x^{(k)} = \frac{\partial \boldsymbol{x}^{(k)}}{\partial \boldsymbol{p}} \tag{6.5}$$

$$\dot{\boldsymbol{s}}_x^{(k)} = \frac{\partial \boldsymbol{s}_x^{(k)}}{\partial t} = \frac{\partial}{\partial t}\left(\frac{\partial \boldsymbol{x}^{(k)}}{\partial \boldsymbol{p}}\right) = \frac{\partial}{\partial \boldsymbol{p}}\left(\frac{\partial \boldsymbol{x}^{(k)}}{\partial t}\right) = \frac{\partial \dot{\boldsymbol{x}}^{(k)}}{\partial \boldsymbol{p}} \tag{6.6}$$

From which results the formulation of sensitivity trajectories (Feehery 1998)

$$\begin{bmatrix} \dfrac{\partial \boldsymbol{f}^{(k)}}{\partial \dot{\boldsymbol{x}}^{(k)}} & \dfrac{\partial \boldsymbol{f}^{(k)}}{\partial \boldsymbol{x}^{(k)}} \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{s}}_x^{(k)} \\ \boldsymbol{s}_x^{(k)} \end{bmatrix} = - \begin{bmatrix} \dfrac{\partial \boldsymbol{f}^{(k)}}{\partial \boldsymbol{u}^{(k)}} \dfrac{\partial \boldsymbol{u}^{(k)}}{\partial \boldsymbol{p}} + \dfrac{\partial \boldsymbol{f}^{(k)}}{\partial \boldsymbol{p}} \end{bmatrix} \tag{6.7}$$

6.3 Sensitivity Transfer at the Time of Transition

To evaluate sensitivities the discontinuity system is differentiated with the only degrees of freedom, time-invariant parameters. The discontinuity function is expressed as $g_{ij}^{(k)}\left(\dot{\boldsymbol{x}}^{(k)}, \boldsymbol{x}^{(k)}, \boldsymbol{u}^{(k)}, \boldsymbol{p}, t\right)$ for $i = 1, ..., n_j^{(k)}$. If the following system exists later in the new mode $k+1$ (Feehery 1998)

$$\dot{\boldsymbol{x}}^{(k)} = \dot{\boldsymbol{x}}^{(k)}(\boldsymbol{p}, t) \qquad \boldsymbol{x}^{(k)} = \boldsymbol{x}^{(k)}(\boldsymbol{p}, t) \tag{6.8}$$

$$g_{k+1}^{(k)}(\dot{\boldsymbol{x}}^{(k)}, \boldsymbol{x}^{(k)}, \boldsymbol{u}^{(k)}, \boldsymbol{p}, t) = 0 \tag{6.9}$$

$$T_k(\dot{\boldsymbol{x}}^{(k)}, \boldsymbol{x}^{(k)}, \boldsymbol{u}^{(k)}, \boldsymbol{p}, t) = 0 \tag{6.10}$$

$$\boldsymbol{f}^{(k+1)}(\dot{\boldsymbol{x}}^{(k+1)}, \boldsymbol{x}^{(k+1)}, \boldsymbol{u}^{(k+1)}, \boldsymbol{p}, t) = 0 \tag{6.11}$$

we can easily use the chain rule

$$\begin{bmatrix} \frac{\partial T_{k+1}^{(k)}}{\partial \dot{\boldsymbol{x}}^{(k)}} & 0 \\ \frac{\partial T_{k+1}^{(k)}}{\partial \boldsymbol{x}^{(k)}} & 0 \\ \frac{\partial T_{k+1}^{(k)}}{\partial \boldsymbol{u}^{(k)}} & 0 \\ \frac{\partial T_{k+1}^{(k)}}{\partial \dot{\boldsymbol{x}}^{(k+1)}} & \frac{\partial \boldsymbol{f}^{(k+1)}}{\partial \dot{\boldsymbol{x}}^{(k+1)}} \\ \frac{\partial T_{k+1}^{(k)}}{\partial \boldsymbol{x}^{(k+1)}} & \frac{\partial \boldsymbol{f}^{(k+1)}}{\partial \boldsymbol{x}^{(k+1)}} \\ \frac{\partial T_{k+1}^{(k)}}{\partial \boldsymbol{u}^{(k+1)}} & \frac{\partial \boldsymbol{f}^{(k+1)}}{\partial \boldsymbol{u}^{(k+1)}} \\ \frac{\partial T_{k+1}^{(k)}}{\partial \boldsymbol{p}} & \frac{\partial \boldsymbol{f}^{(k+1)}}{\partial \boldsymbol{p}} \\ \frac{\partial T_{k+1}^{(k)}}{\partial t} & \frac{\partial \boldsymbol{f}^{(k+1)}}{\partial t} \end{bmatrix}^T \begin{bmatrix} \frac{\partial \dot{\boldsymbol{x}}^{(k)}}{\partial \boldsymbol{p}} & \frac{\partial \dot{\boldsymbol{x}}^{(k)}}{\partial t} \\ \frac{\partial \boldsymbol{x}^{(k)}}{\partial \boldsymbol{p}} & \frac{\partial \boldsymbol{x}^{(k)}}{\partial t} \\ \frac{\partial \boldsymbol{u}^{(k)}}{\partial \boldsymbol{p}} & \frac{\partial \boldsymbol{u}^{(k)}}{\partial t} \\ \frac{\partial \dot{\boldsymbol{x}}^{(k+1)}}{\partial \boldsymbol{p}} & \frac{\partial \dot{\boldsymbol{x}}^{(k+1)}}{\partial t} \\ \frac{\partial \boldsymbol{x}^{(k+1)}}{\partial \boldsymbol{p}} & \frac{\partial \boldsymbol{x}^{(k+1)}}{\partial t} \\ \frac{\partial \boldsymbol{u}^{(k+1)}}{\partial \boldsymbol{p}} & \frac{\partial \boldsymbol{u}^{(k+1)}}{\partial t} \\ I & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} I \\ \frac{\partial t}{\partial \boldsymbol{p}} \end{bmatrix} = 0 \tag{6.12}$$

Now, by rearranging according to the known and unknown variables, we obtain

$$\begin{aligned} \begin{bmatrix} \frac{\partial T_{k+1}^{(k)}}{\partial \dot{\boldsymbol{x}}^{(k+1)}} & \frac{\partial \boldsymbol{f}^{(k+1)}}{\partial \dot{\boldsymbol{x}}^{(k+1)}} \\ \frac{\partial T_{k+1}^{(k)}}{\partial \boldsymbol{x}^{(k+1)}} & -\frac{\partial \boldsymbol{f}^{(k+1)}}{\partial \boldsymbol{x}^{(k+1)}} \end{bmatrix}^T \begin{bmatrix} \frac{\partial \dot{\boldsymbol{x}}^{(k+1)}}{\partial \boldsymbol{p}} \\ \frac{\partial \boldsymbol{x}^{k+1}}{\partial \boldsymbol{p}} \end{bmatrix} &= - \begin{bmatrix} \frac{\partial T_{k+1}^{(k)}}{\partial \dot{\boldsymbol{x}}^{(k+1)}} & \frac{\partial \boldsymbol{f}^{(k+1)}}{\partial \dot{\boldsymbol{x}}^{(k+1)}} \\ \frac{\partial T_{k+1}^{(k)}}{\partial \boldsymbol{x}^{(k+1)}} & \frac{\partial \boldsymbol{f}^{(k+1)}}{\partial \boldsymbol{x}^{(k+1)}} \end{bmatrix}^T \begin{bmatrix} \frac{\partial \dot{\boldsymbol{x}}^{(k+1)}}{\partial t} \\ \frac{\partial \boldsymbol{x}^{k+1}}{\partial t} \end{bmatrix} \frac{\partial t}{\partial \boldsymbol{p}} \\ &\quad - \begin{bmatrix} \frac{\partial T_{k+1}^{(k)}}{\partial \dot{\boldsymbol{x}}^{(k)}} & 0 \\ \frac{\partial T_{k+1}^{(k)}}{\partial \boldsymbol{x}^{(k)}} & 0 \\ \frac{\partial T_{k+1}^{(k)}}{\partial \boldsymbol{u}^{(k)}} & 0 \\ \frac{\partial T_{k+1}^{(k)}}{\partial \boldsymbol{u}^{(k+1)}} & \frac{\partial \boldsymbol{f}^{(k+1)}}{\partial \boldsymbol{u}^{(k+1)}} \\ \frac{\partial T_{k+1}^{(k)}}{\partial \boldsymbol{p}} & \frac{\partial \boldsymbol{f}^{(k+1)}}{\partial \boldsymbol{p}} \\ \frac{\partial T_{k+1}^{(k)}}{\partial t} & \frac{\partial \boldsymbol{f}^{(k+1)}}{\partial t} \end{bmatrix}^T \begin{bmatrix} \frac{\partial \dot{\boldsymbol{x}}^{(k)}}{\partial \boldsymbol{p}} + \frac{\partial \dot{\boldsymbol{x}}^{(k)}}{\partial t}\frac{\partial t}{\partial \boldsymbol{p}} \\ \frac{\partial \boldsymbol{x}^{(k)}}{\partial \boldsymbol{p}} + \frac{\partial \boldsymbol{x}^{(k)}}{\partial t}\frac{\partial t}{\partial \boldsymbol{p}} \\ \frac{\partial \boldsymbol{u}^{(k)}}{\partial \boldsymbol{p}} + \frac{\partial \boldsymbol{u}^{(k)}}{\partial t}\frac{\partial t}{\partial \boldsymbol{p}} \\ \frac{\partial \boldsymbol{u}^{(k+1)}}{\partial \boldsymbol{p}} + \frac{\partial \boldsymbol{u}^{(k+1)}}{\partial t}\frac{\partial t}{\partial \boldsymbol{p}} \\ I \\ \frac{\partial t}{\partial \boldsymbol{p}} \end{bmatrix} \end{aligned} \tag{6.13}$$

If we assume a transition function in the form of $T_{k+1}^{(k)} = \left[x^{(k+1)} - x^{(k)}\right]_{k+1}^{(k)}$ then the

Eq. 6.13 will take the following form

$$\begin{bmatrix} \frac{\partial T_{k+1}^{(k)}}{\partial \dot{\boldsymbol{x}}^{(k+1)}} & \frac{\partial \boldsymbol{f}^{(k+1)}}{\partial \dot{\boldsymbol{x}}^{(k+1)}} \\ \frac{\partial T_{k+1}^{(k)}}{\partial \boldsymbol{x}^{(k+1)}} & -\frac{\partial \boldsymbol{f}^{(k+1)}}{\partial \boldsymbol{x}^{(k+1)}} \end{bmatrix}^T \begin{bmatrix} \frac{\partial \dot{\boldsymbol{x}}^{(k+1)}}{\partial \boldsymbol{p}} \\ \frac{\partial \boldsymbol{x}^{k+1}}{\partial \boldsymbol{p}} \end{bmatrix} = - \begin{bmatrix} \frac{\partial T_{k+1}^{(k)}}{\partial \dot{\boldsymbol{x}}^{(k)}} & 0 \\ \frac{\partial T_{k+1}^{(k)}}{\partial \boldsymbol{x}^{(k)}} & 0 \\ \frac{\partial T_{k+1}^{(k)}}{\partial \boldsymbol{u}^{(k)}} & 0 \\ \frac{\partial T_{k+1}^{(k)}}{\partial \boldsymbol{u}^{(k+1)}} & \frac{\partial \boldsymbol{f}^{(k+1)}}{\partial \boldsymbol{u}^{(k+1)}} \\ \frac{\partial T_{k+1}^{(k)}}{\partial \boldsymbol{p}} & \frac{\partial \boldsymbol{f}^{(k+1)}}{\partial \boldsymbol{p}} \\ \frac{\partial T_{k+1}^{(k)}}{\partial t} & \frac{\partial \boldsymbol{f}^{(k+1)}}{\partial t} \end{bmatrix}^T \begin{bmatrix} \frac{\partial \dot{\boldsymbol{x}}^{(k)}}{\partial \boldsymbol{p}} \\ \frac{\partial \boldsymbol{x}^{(k)}}{\partial \boldsymbol{p}} \\ \frac{\partial \boldsymbol{u}^{(k)}}{\partial \boldsymbol{p}} \\ \frac{\partial \boldsymbol{u}^{(k+1)}}{\partial \boldsymbol{p}} \\ I \\ 0 \end{bmatrix} \tag{6.14}$$

Eq. 6.14 can be easily simplified as follows

$$\begin{bmatrix} 0 & 1 \\ 1 & -\frac{\partial \boldsymbol{f}^{(k+1)}}{\partial \boldsymbol{x}^{(k+1)}} \end{bmatrix}^T \begin{bmatrix} \dot{\boldsymbol{s}}_x^{(k+1)} \\ \boldsymbol{s}_x^{(k+1)} \end{bmatrix} = - \begin{bmatrix} 0 & 0 \\ -1 & 0 \\ 0 & 0 \\ 0 & -\frac{\partial \boldsymbol{f}^{(k+1)}}{\partial \boldsymbol{u}^{(k+1)}} \\ 0 & -\frac{\partial \boldsymbol{f}^{(k+1)}}{\partial \boldsymbol{p}} \\ 0 & 0 \end{bmatrix}^T \begin{bmatrix} \dot{\boldsymbol{s}}_x^{(k)} \\ \boldsymbol{s}_x^{(k)} \\ \frac{\partial \boldsymbol{u}^{(k)}}{\partial \boldsymbol{p}} \\ \frac{\partial \boldsymbol{u}^{(k+1)}}{\partial \boldsymbol{p}} \\ I \\ 0 \end{bmatrix} \tag{6.15}$$

After some smaller rearrangements we can rewrite the Eq. 6.15 in the form of

$$\begin{bmatrix} \boldsymbol{s}_x^{(k+1)} \\ \dot{\boldsymbol{s}}_x^{(k+1)} - \frac{\partial \boldsymbol{f}^{(k+1)}}{\partial \boldsymbol{x}^{(k+1)}} \boldsymbol{s}_x^{(k+1)} \end{bmatrix} = \begin{bmatrix} \boldsymbol{s}_x^{(k)} \\ \frac{\partial \boldsymbol{f}^{(k+1)}}{\partial \boldsymbol{u}^{(k+1)}} \frac{\partial \boldsymbol{u}^{(k+1)}}{\partial \boldsymbol{p}} + \frac{\partial \boldsymbol{f}^{(k+1)}}{\partial \boldsymbol{p}} \end{bmatrix} \tag{6.16}$$

The first row of Eq. 6.16 proves the continuity of sensitivity equations over each stage. The second row defines the sensitivities over the stage $(k+1)$.

Part II

Application Part

Chapter 7

Hydrolysis of Sucrose by Invertase

The sucrose hydrolysis is an enzymatically catalyzed reaction during which the sucrose is hydrolyzed into glucose and fructose by invertase obtained from Saccharomyces cerevisiae. This irreversible reaction can be expressed as

$$sucrose + H_2O \xrightarrow{r} \mathrm{D} - glucose + \mathrm{D} - fructose \tag{7.1}$$

here r stands for the reaction rate.

7.1 Problem Formulation

Assume the above mentioned chemical reaction 7.1. The velocity of this chemical reaction r is a function of the sucrose concentration c_s

$$r = \frac{v_m c_s}{K_m + c_s \left(1 + \frac{c_s}{K_i}\right)} \tag{7.2}$$

here v_m denotes the maximal reaction velocity, K_m stands for Michaelis-Menten constant and K_i defines the inhibition coefficient. We distinguish the batch and the continuous reactor. In both cases we assume a constant reactor volume $V = 0.9[L]$.Volumetric feed-flow is equal $F = 0$ for batch process and $F = 1.02 \times 10^{-3}[L.min^{-1}]$ for the continuous process. Our task is to find the optimal kinetic parameters. First of all, we have to generate pseudo-experimental data for c_s with random error. To generate the experimental data we used the following values of kinetic parameters

$$v_m = 0.0026[molL^{-1}min^{-1}] \qquad K_m = 0.0417[molL^{-1}] \qquad K_i = 0.3916[molL^{-1}] \tag{7.3}$$

7.2 Hydrolysis of Sucrose in a Batch Reactor

We assume the following reaction taking place in a batch reactor

$$S \rightarrow P \tag{7.4}$$

For simplicity, we also assume that an equimolar amount of glucose and fructose is produced, which is denoted as product P. Further, we assume that the mixture is thoroughly mixed during the reaction and there is no enzyme deactivation. Then the initial value problem can be expressed as follows

- *Medium*

$$\frac{\mathrm{d}c_s}{\mathrm{d}t} = \frac{F}{V}(c_{s,f} - c_s) - \frac{v_m c_s}{K_m + c_s\left(1 + \frac{c_s}{K_i}\right)} \qquad c_s(0) = c_{s0} \tag{7.5}$$

- *Product*

$$\frac{\mathrm{d}c_p}{\mathrm{d}t} = -\frac{F}{V}c_p + \frac{F}{V}(c_{s,f} - c_s) - \frac{v_m c_s}{K_m + c_s\left(1 + \frac{c_s}{K_i}\right)} \qquad c_p(0) = c_{p0} \tag{7.6}$$

As we assume that the chemical reactor is a batch reactor, thus the volumetric flow F will be equal to zero. Then the Eqs. 7.5 and 7.6 can be altered in the following way

- *Medium*

$$\frac{\mathrm{d}c_s}{\mathrm{d}t} = -\frac{v_m c_s}{K_m + c_s\left(1 + \frac{c_s}{K_i}\right)} \qquad c_s(0) = c_{s0} \tag{7.7}$$

- *Product*

$$\frac{\mathrm{d}c_p}{\mathrm{d}t} = -\frac{v_m c_s}{K_m + c_s\left(1 + \frac{c_s}{K_i}\right)} \qquad c_p(0) = c_{p0} \tag{7.8}$$

From Eqs. 7.7 and 7.8 is clear that the medium and the product concentration has the same value with a different sign.

7.2.1 Solving the parameter estimation problem in a batch reactor

Our aim is to find the optimal values of the kinetic parameters v_m, K_m and K_i and to minimize the value of the following objective function

$$J = \min_{v_m, K_m, K_i} \sum_{j=1}^{20} \left(c_s\left(t_j\right) - c_s^{exp}\left(t_j\right)\right)^2 \tag{7.9}$$

For the initial medium concentration of $c_{s0} = 1.5[molL^{-1}]$. The dynamic optimization problem was solved by using the method of control vector parametrization and the gradients were computed by finite differences and by sensitivity equations.

To solve this optimization problem we used the method of control vector parametrization with two different approaches of gradient computing. Obtained results are shown and compared in Tab. 7.1. Our aim was to minimize the value of the objective function J. As we can see the value of the objective was slightly smaller finite differences, than for sensitivity equations. The yielded values of the first kinetic parameter v_m were the same. More obvious is the difference between the optimal values of the second kinetic parameter K_m. The parameter value obtained by using the first method was much closer to the given value. And finally, the obtained value of the third parameter K_i was closer to the given value by using sensitivity equations. Optimal values obtained by using the finite differences approach showed to be more appropriate in this case because optimal values were reached after 12 iterations and were more accurate. Sensitivity equations approach achieved the optimal values after 6 iterations but were less accurate. This results from the fact, that this problem is strongly nonlinear, thus it has several local extremes.

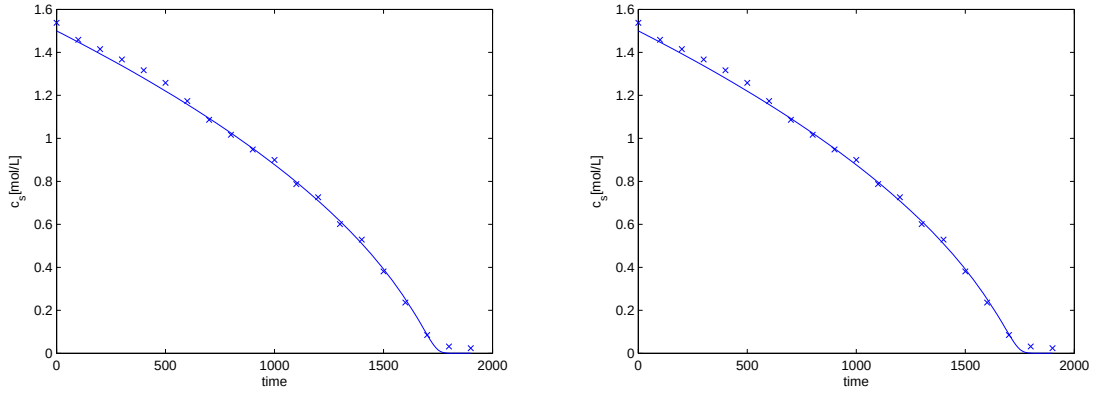
Table 7.1: Comparison of computational aspects for different methods for computing gradients.

	Finite differences	Sensitivity Equations
J	0.0094	0.0096
v_m [$\mathrm{mol\,L^{-1}\,min^{-1}}$]	0.0300	0.0030
K_m [$\mathrm{mol\,L^{-1}}$]	0.0405	0.0428
K_i [$\mathrm{mol\,L^{-1}}$]	0.3145	0.3194

Graphical comparison of optimal state trajectories, obtained by using different methods of computing gradients, is shown in Fig. 7.1. As we can see there are only minor differences between the optimal state trajectory obtained by method of finite differences Fig. 7.1(a) and state trajectory obtained by using method of sensitivity equations Fig. 7.1(a). The final concentration of the medium was equal to zero in both cases, that means that all medium was converted into product during the reaction.

7.3 Hydrolysis of Sucrose in a Continuous Reactor

In the next case we assume a continuous reactor as shown on Fig. 7.2. This continuous reactor includes an outer membrane which is fully permeable to the medium but im-



(a) Finite differences method

(b) Sensitivity equations method

Figure 7.1: Experimental points and optimal state variables trajectories for different methods of computing gradients.

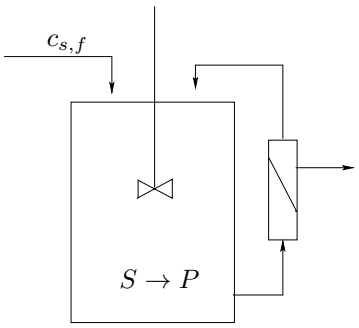


Figure 7.2: Continuous reactor

permeable to the enzyme. We presume an equal input and output flow and a constant volume V. We also assume that the mixture is thoroughly mixed during the reaction and there is no enzyme deactivation.

7.3.1 Solving the parameter estimation problem in a continuous reactor

We consider the same initial value problem as in the previous case described by Eqs. (7.5) and (7.6). We also assume that the concentration of the input stream will be constant with the following initial estimates $c_{s,0} = 3[molL^{-1}]$ for the medium and $c_{p,0} = 0.2645[molL^{-1}]$ for the product. Our aim is to find optimal values of kinetic parameter v_m, K_m and K_i and fulfil the end-point constraint $c_{s,f} = 1.5[molL^{-1}]$. The objective function is expressed as

$$J = \min_{v_m, K_m, K_i} \sum_{j=1}^{20} \sum_{i=1}^{2} \left(x_i(t = t_j, v_m, K_m, K_i) - x_i^{exp}(t_j) \right)^2 \tag{7.10}$$

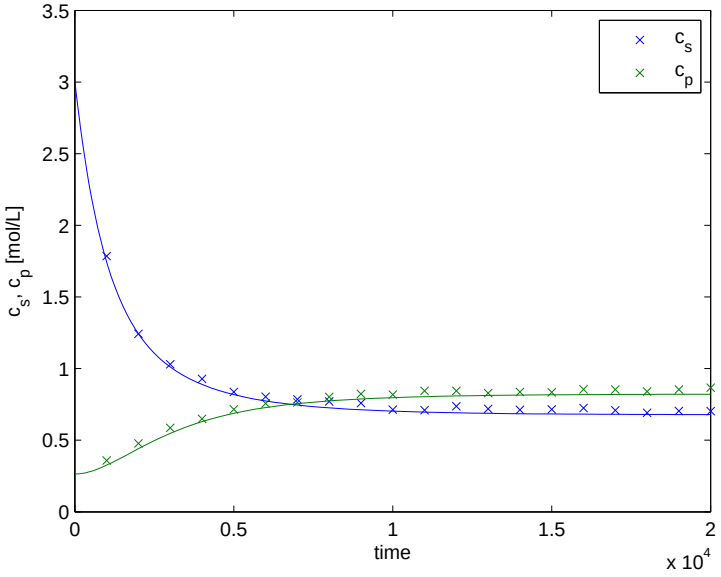
Gained results are gathered in Tab. 7.2. Needed gradients were computed by finite

differences and by sensitivity equations. Obtained values of the objective function J and the first kinetic parameter v_m had the same values. Conspicuous differences are in values of the last two kinetic parameters. For both of these parameters, the method of finite differences shown to be more accurate, because finite differences approach reached the optimal value of the objective function within 24 iterations, while the sensitivity equations approach needed 33 iterations. And also optimal values of kinetic parameters were closer to the given ones.

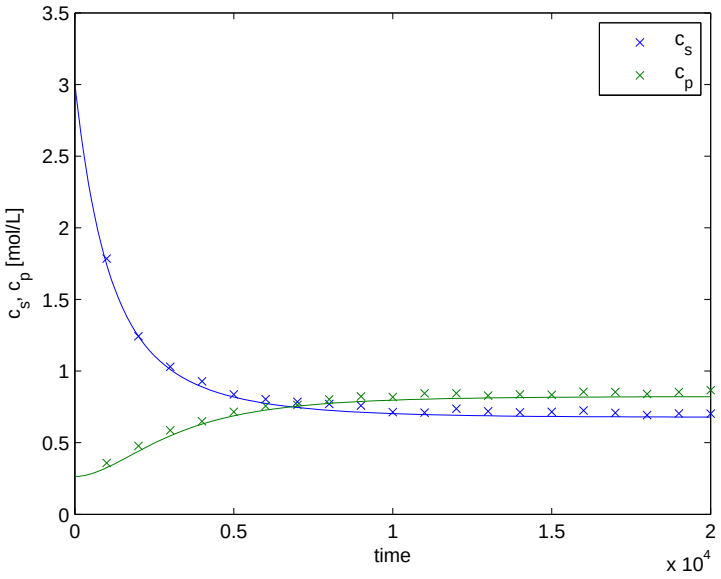
Table 7.2: Comparison of computational aspects for different methods for computing gradients.

	Finite differences	Sensitivity Equations
J	0.0405	0.0405
v_m $[\mathrm{mol\,L^{-1}\,min^{-1}}]$	0.0026	0.0026
K_m $[\mathrm{mol\,L^{-1}}]$	0.0411	0.0404
K_i $[\mathrm{mol\,L^{-1}}]$	0.3812	0.3809

Graphical comparison is shown on Fig. 7.3. As we can see there are only minor differences between the optimal state trajectories obtained by using finite differences Fig. 7.3(a) and sensitivity equations Fig. 7.3(b). The final concentrations of the medium and product were $c_s = 0.6792[molL^{-1}]$, $c_p = 0.8206[molL^{-1}]$ for the case of finite differences and $c_s = 0.6793[molL^{-1}]$, $c_p = 0.8209[molL^{-1}]$.



(a) Finite differences method



(b) Sensitivity equations method

Figure 7.3: Experimental points and optimal state variables trajectories for different methods of computing gradients.

Chapter 8

Catalytic Cracking of Gas Oil

Crude oil is a brown liquid which consists of a mixture of hydrocarbons. Catalytic cracking is the most common way of crude oil processing, during which long hydrocarbon chains are broken down into shorter ones.

8.1 Problem Formulation

Catalytic cracking is an example of parameter estimation where three parameters are going to be optimized. This process is described by two differential equations with constraints. We assume simplified reaction of gas oil (A) to gasoline (Q) and other products (S) (Paulen 2008)

$$A \xrightarrow{k_1} Q$$
$$Q \xrightarrow{k_2} S$$
$$A \xrightarrow{k_3} S$$

Differential equations which are describing the process are defined as follows

$$\dot{x}_1 = -(k_1 + k_3)x_1^2 \qquad \forall t \in [0,1] \tag{8.1}$$
$$\dot{x}_2 = k_1 x_1^2 - k_2 x_2 \qquad \forall t \in [0,1] \tag{8.2}$$

with the following initial conditions a constraints

$$\begin{aligned} x_1(t=0, k_1, k_2, k_3) &= 1 \\ x_2(t=0, k_1, k_2, k_3) &= 0 \\ 0 \leq k_1 &\leq 20 \\ 0 \leq k_2 &\leq 20 \\ 0 \leq k_3 &\leq 20 \end{aligned} \tag{8.3}$$

here x_1 and x_2 are the molar fractions of constituents A and Q (Paulen 2008) and constants k_1, k_2 and k_3 are the rate constants of single reaction. The objective of dynamic optimization problem can be then expressed as

$$\min_{k_1,k_2,k_3} \sum_{j=1}^{20} \sum_{i=1}^{2} \left(x_i \left(t = t_j, k_1, k_2, k_3\right) - x_i^{exp}\left(t_j\right)\right)^2 \tag{8.4}$$

here $x_i(t_j)$ is an experimental point for the variable i at time t_j (Paulen 2008).

8.2 Solving the parameter estimation problem

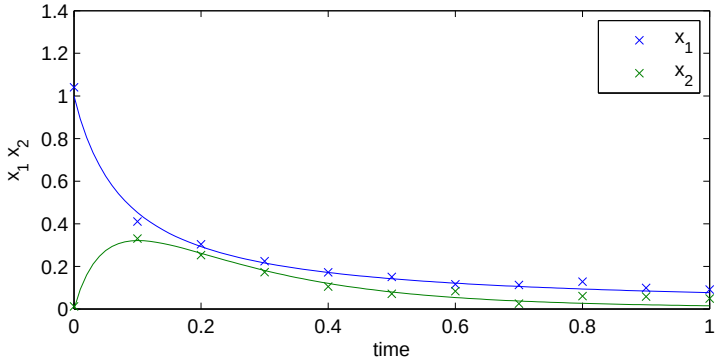
Problem of dynamic optimization was solved by using the method of control vector parametrization. Our aim is to find the optimal values of parameters k_1, k_2 and k_3 and to minimize the objective function 8.4. We generated the pseudo-experimental data with a 5% error.

Obtained results are summarized in Tab. 8.1. The first method used was the method of finite differences. The optimal value of the objective function J was reached after 21 iterations. The second one were the sensitivity equations. We achieved the same value of the objective function within the same number of iterations. If we compare the values of each parameter, we can see that there are only small differences in their values.

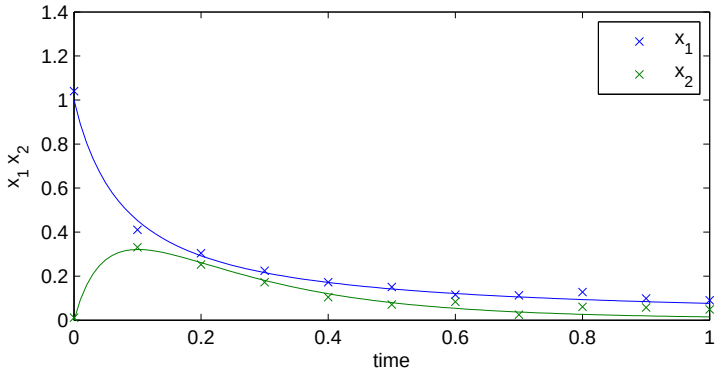
Table 8.1: Comparison of computational aspects for different methods for computing gradients.

	Finite differences	Sensitivity Equations
J	0.0108	0.0108
k_1	10.6472	10.6488
k_2	6.7322	6.7331
k_3	1.4469	1.4458

The experimental points and the optimal state variables trajectories for different gradient computing approaches are shown on Fig. 8.1. Fig. 8.1(a) depicts the optimal state trajectory obtained by finite differences while Fig. 8.1(b) shows the optimal state trajectory obtained by using sensitivity equations. As it can be seen, there are only minor differences between these trajectories, what is expected, because the obtained results are almost the same.



(a) Finite differences method



(b) Sensitivity equations method

Figure 8.1: Experimental points and optimal state variables trajectories for the process of catalytic cracking.

Chapter 9

Batch Reactor

9.1 Problem Formulation

We consider a batch reactor with the following consecutive reactions

$$A \xrightarrow{k_1} B \xrightarrow{k_2} C$$

Our aim is to find such an optimal temperature trajectory inside the reactor that maximizes the concentration of the intermediate product B at the end of the entire process. Hence the optimization problem can be defined as follows

$$\max_{T(t)} J = x_2(t_f) \tag{9.1}$$

The system is described by the following ODE system

$$\dot{x}_1 = -k_1 x_1^2 \qquad x_1(0) = 1 \tag{9.2}$$

$$\dot{x}_2 = k_1 x_1^2 - k_2 x_2 \qquad x_2(0) = 0 \tag{9.3}$$

$$0 = k_1 - 4000 e^{(-\frac{2500}{T})} \tag{9.4}$$

$$0 = k_2 - 620000 e^{(-\frac{5000}{T})} \tag{9.5}$$

here the state variables x_1 stands for the concentration of reactant A and x_2 denotes the concentration of product B. The control variable is the temperature inside the reactor T. We also consider the following initial conditions and constraints

$$\begin{aligned} T &\in [298, 398] \\ x_1(t_0) &= 1 \\ x_2(t_0) &= 0 \\ t_f &= 1 \end{aligned}$$

where t_f stands for the final time. We are going to solve this optimization problem using two different numerical methods - orthogonal collocation and sensitivity equations. In both cases we consider a time period $t \in [t_0, t_f]$ subdivided into four different time intervals which lengths Δt_i are optimized too.

9.2 Finding Optimal Control Using Orthogonal Collocation

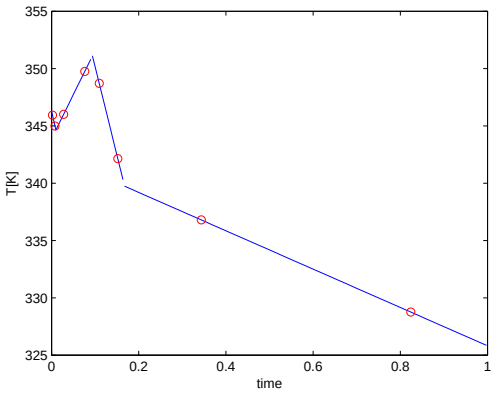
Orthogonal collocation was the first method used. We used five collocation points for state variables and two collocation points for control variable. The optimization problem was solved by NLP solver fmincon using three different algorithms. Optimization toolbox fmincon is based on finding constrained minimum of a scalar function. The first used was the algorithm of trust-region-reflective. This algorithm is based on the approximation of the former function with a simpler function, which accurately reflects the behavior of the former function. The second tested algorithm was the sequential quadratic programming or SQP, based on Newton's method, which is based on finding the places, where the gradient is equal to zero. And the last used algorithm was the algorithm of interior point which belongs to barrier methods.

Table 9.1: Orthogonal collocation: Comparison of resulting objective function values for different algorithms of piece-wise linear control.

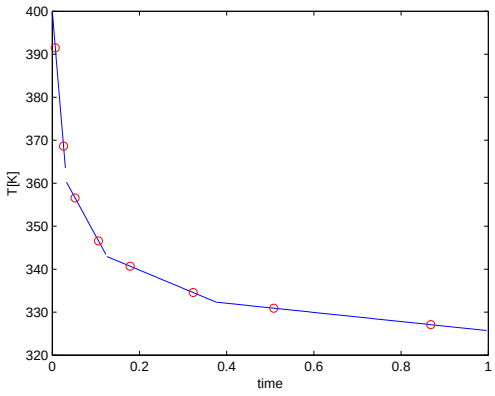
	Trust-region-reflective	SQP	Interior point
Optimal value	0.6091	0.6108	0.6108
Number of iterations	771	397	8890
CPU-time [s]	51.5169	19.8568	334.4820

Obtained results are shown and compared in Tab. 9.1. We used the same initial guess for each algorithm and the same end-point constraint $t_f = 1$. Our aim was to find such an optimal temperature trajectory, which maximizes the value of the objective function in the final time. This optimization problem was solved by (Logsdon and Biegler 1989) and (Rajesh et al. 2001). (Logsdon and Biegler 1989) reached the optimal value 0.6108 while (Rajesh et al. 2001) obtained the optimum value 0.6105. We reached the same optimal values by SQP and by interior point. With respect to this criterion the worst was the algorithm of trust-region-reflective. If we compare the number of iterations and the computing time, SQP algorithm proved to be the fastest. If we compare the iteration velocity [iteration/CPUtime], the fastest is the trust-region-reflective algorithm with the value of 14.5 [iteration/s]. The second is the

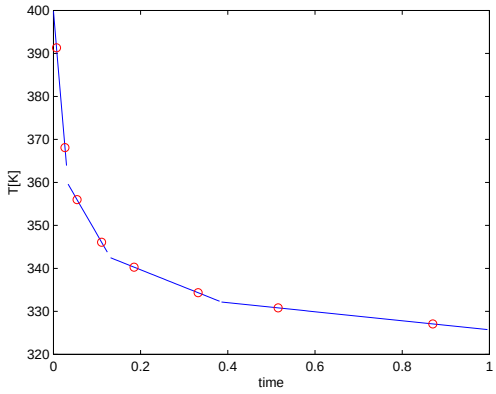
SQP algorithm with 20 [iteration/s] and the slowest is the interior point algorithm with 26.6 [iteration/s]. After the overall comparison we can say, that the most suitable is the SQP algorithm because the same optimal value was reached in shorter time.



(a) Trust-region-reflective



(b) Sequential quadratic programming



(c) Interior point

Figure 9.1: Orthogonal collocation: Comparison of the optimal control trajectories for the case of 4 time intervals with optimized lengths.

Graphical comparison of optimal temperature trajectories obtained using different algorithms are shown on Fig. 9.1. As we can see there is only minor difference in temperature trajectories in Figs. 9.1(b) and 9.1(c). Temperature trajectory obtained using trust-region-reflective shown on Fig. 9.1(a) is apparently different, thus this method proved to be inappropriate for this kind of optimization problem.

9.3 Finding Optimal Control Using Sensitivity Equations

9.3.1 Solving the optimization problem using fmincon

We assumed the same chemical reactor described by Eq. 9.5. First, we defined the sensitivity equations as partial derivatives of state variables according to the parameters. We obtained 6 ($n_x \times n_p$) sensitivity equations for each time period. Optimization problem was solved by the same NLP solver fmincon using three algorithms - trust-region-reflective, SQP and interior point. We used the same initial estimates and same end-point constraint.

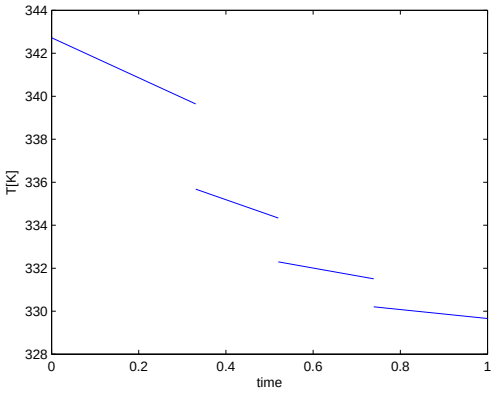
Table 9.2: Sensitivity equations: Comparison of resulting objective function values for different algorithms of piece-wise linear control.

	Trust-region-reflective	SQP	Interior point
Optimal value	0.6084	0.6108	0.6108
Number of iterations	9	115	135
CPU-time [s]	3.3481	29.0953	17.5581

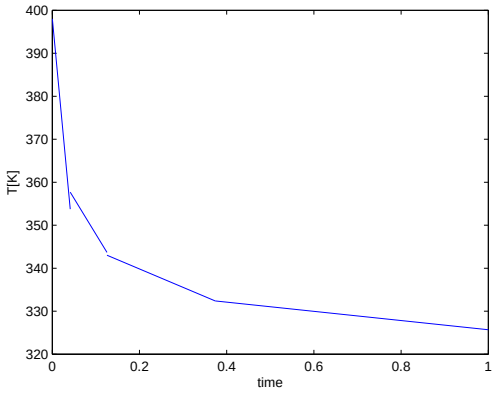
Tab. 9.2 shows the results obtained using sensitivity equations. If we compare the values of objective functions, we can see that the worst result was obtained by using trust-region-reflective algorithm despite the fact, that this algorithm needed only 9 iterations to get the optimal value. Comparing the iteration velocity [iteration/CPUtime] it can be seen, that the fastest is the trust-region-reflective with the value of 2.7 [iteration/s]. Next is the SQP algorithm with 3.9 [iteration/s] and the slowest is the algorithm of interior point with 7.7 [iteration/s]. After comparing the same criteria as in the previous case, we can say that the most appropriate was the solution obtained by using SQP algorithm.

Graphical comparison of optimal control trajectories obtained by using three different algorithms are shown in Fig. 9.2. As we can see, optimal control trajectories obtained by SQP Fig. 9.2(b) and by interior point Fig. 9.2(c) have almost the same profile. Optimal control profile shown in Fig. 9.2(a), obtained by using the trust region reflective algorithm, shown to be inaccurate.

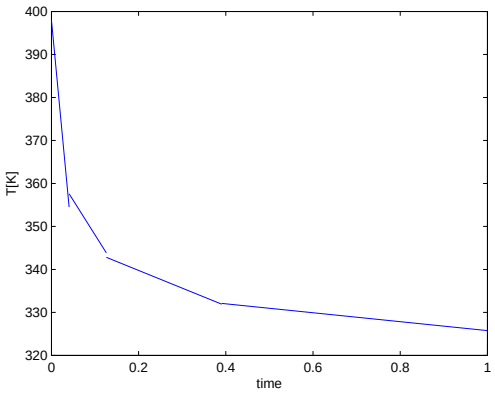
The results clearly show, that the SQP algorithm is the most appropriate for both methods. The comparison of these numerical methods is shown in Tab. 9.3. As we can see, both of them reached the same value of the objective function, but with a different number of iterations and different CPU-time. The optimization problem solved by the method of orthogonal collocation reached the optimum value in 397 iterations within



(a) Trust-region-reflective



(b) Sequential quadratic programming



(c) Interior point

Figure 9.2: Sensitivity equations: Comparison of the optimal control trajectories for the case of 4 time intervals with optimized lengths.

20[s]. In contrast, the optimum value for the method of sensitivity equations was reached after 115 iterations within 30[s].In this case, the method of sensitivity equations shown to be more suitable, because it reached the same value of the objective function with less iterations and within shorter time.

Table 9.3: Comparison of resulting chosen criteria for two different numerical methods.

	Orthogonal collocation	CVP(Sensitivity equations)
Optimal value	0.6108	0.6108
Number of iterations	397	115
CPU-time [s]	19.8568	29.0953
[iteration/s]	20	4

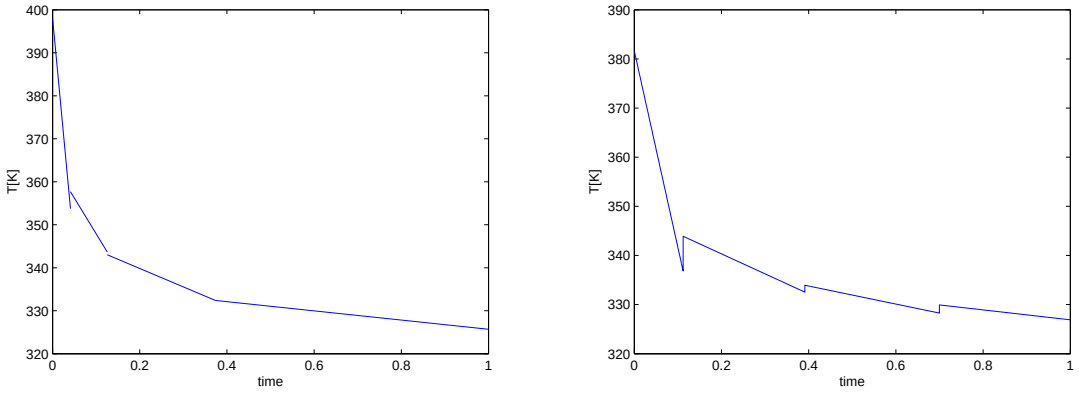
9.3.2 Solving the optimization problem using CVP_SS

gPROMS is a platform used for process modeling. Solving dynamic optimization problems is based on the sensitivity equations method. We have used the CVP_SS solver, which implements the algorithm of control vector parametrization via single shooting. Comparison of results obtained by two different solvers is shown in Tab. 9.4. In both cases the method of sensitivity equations was used. We used the same initial estimates and the same end-point constraint $t_f = 1$ as previously. The value of the objective function obtained using fmincon shown to be a little better than the value obtained by CVP_SS. On the other side CVP_SS needed only 10 iterations to reach the optimum while fmincon needed 115. If we compare the iteration velocity, fmincon is iterating faster then CVP_SS. After comparing all criteria we can say, that CVP_SS solver is more suitable, even it had little bit worse objective value, because the difference between reached optimal values is small, but it was achieved in a shorter time within less iterations.

Table 9.4: Results comparison of different NLP solvers.

	fmincon	CVP_SS
Optimal value	0.6108	0.6106
Number of iterations	115	10
CPU-time [s]	29.0953	10.265
[iteration/s]	4	0.7

Graphical solutions are shown on Fig. 9.3. The entire time interval was subdivided into 4 time elements, which lengths were optimized. We can observe significant differences in these two trajectories. The first 2 time intervals counted by fmincon are small (10^{-2}order), while the rest 2 intervals are 10^{-1}order. Each time interval obtained by CVP_SS has the same rank(10^{-1}).



(a) fmincon (b) PVC_SS

Figure 9.3: Sensitivity equations: Comparison of the optimal control trajectories obtained using different NLP solvers.

Chapter 10

Polymerization Process

The emulsion polymerization is a procedure in which, an aqueous dispersion of monomer (or a mixture of monomer) is transformed by radical polymerization in a stable dispersion of polymer particles. The reaction medium is mainly composed of (Fournier 1998)

- Dispersing medium (water in general).
- Monomer (which must be insoluble in the dispersing medium).
- Initiator (which is soluble in the dispersing medium, and insoluble in the monomer). Choice of the initiator depends on the temperature field. For moderate to high temperature initiators such as potassium persulfate and sodium are commonly used. For polymerizations conducted at lower temperatures redox initiators are usually used.
- Emulsifier. The emulsifier molecules have a hydrophilic end and a hydrophobic hydrocarbon skeleton. Due to the forces between the hydrophobic ends, the molecules of emusifiant will form aggregates, called micelles, starting from a critical concentration, also called the critical micelle concentration. A micelle is a set of 50 to 100 molecules of emulsifier having their hydrophilic end oriented towards the aqueous phase.

Sometimes it may include transfer agents and other additives too.

At the beginning of the polymerization, the emulsifier can be found in three forms - dissolved in the dispersing medium, micelles form and adsorbed on the surface of monomer droplets.

10.1 Polymerization Mechanism

The mechanisms of emulsion polymerization reactions are described in three stages

1. Step 1: The polymer particles are nucleated. The overall rate of polymerization increases with time as and as the number of particles increases.

2. Step 2: This step is the stage of particles growth. Their amount is now constant until the end of the polymerization. Because of the rapid diffusion of the monomer droplets to particles, they are saturated as long as monomer droplets exist, and consequently the concentration of particles in monomer remains constant. At the end of Step 2, the monomer droplets have disappeared.

3. Step 3: The particles are no longer supplied with monomer, concentration of monomer decreases regularly.

In general, radical polymerization can be divided into four stages

Initiator decomposition:	$\mathrm{A} \longrightarrow 2\mathrm{R}^\bullet$	$R_a = 2fk_dA$
Particle formation:	$\mathrm{R}^\bullet + \mathrm{m} \longrightarrow \mathrm{N}^\bullet$	$R_n = k_{cm}mR^\bullet$
Initiation:	$\mathrm{N} + \mathrm{R}^\bullet \longrightarrow \mathrm{N}^\bullet$	$R_i = k_{cp}NR^\bullet$
Termination:	$\mathrm{N}^\bullet + \mathrm{R}^\bullet \longrightarrow \mathrm{N}$	$R_t = k_{cp}N^\bullet R^\bullet$
Propagation:	$\mathrm{P}_j^\bullet + \mathrm{M} \longrightarrow \mathrm{P}_{j+1}^\bullet$	$R_p = k_p M_\mathrm{p} N^\bullet$
Transfer to monomer:	$\mathrm{P}_j^\bullet + \mathrm{M} \longrightarrow \mathrm{M}^\bullet + \mathrm{P}_j^\bullet$	$R_{trM} = k_{trM} M_\mathrm{p} N^\bullet$

Where A denotes the concentration of initiator, $R^\bullet$ denotes the initiator radical concentration, f represents initiator efficiency, M is monomer concentration and m denotes the number of micelles per unit volume.

10.2 Kinetic Model

Material balance is composed of monomer and particles balance.

The assumption of quasi-stationary state applied to radicals decomposed from the initiator in the aqueous phase allows to write

$$\dot{N}_\mathrm{p} = k_{cm}mR^\bullet N_A = k_{cm}m\frac{R_aN_A}{1+\dfrac{\varepsilon N_\mathrm{p}}{SN_A}} \tag{10.1}$$

Where N_A denotes Avogadro's number and ε is a factor linked to the efficiency of particles with respect to micelles capturing radicals from the decomposition of the initiator.

$$\varepsilon = \frac{k_{cp}n_s}{k_{cm}} \tag{10.2}$$

n_s is the number of aggregation of micelles characterized as

$$n_s = \frac{SN_A}{m} \tag{10.3}$$

Now, the concentration of the emulsifier can be expressed as

$$S = S_o - k_v(XM_o)^{2/3}N_{\mathrm{p}}^{1/3} \tag{10.4}$$

where S_o announces initial emulsifier concentration, M_o announces initial monomer concentration and X refers to monomer conversion. Along with

$$k_v = \left[\frac{36\pi M_{\mathrm{M}}^2}{\omega_{\mathrm{P}}^2(a_sN_A)^3\rho_{\mathrm{P}}^2}\right]^{1/3} \tag{10.5}$$

where M_{M} is referring to monomer molecular weight, ω_{P} to polymer weight fraction in particles, variable a_s to surface area occupied by an emulsifier molecule and ρ_{P} is referring to polymer particle density. The monomer disappears during the propagation within the particles and the transfer rate to monomer is negligible at the speed of propagation. This relation is expressed as

$$\dot{M} = -R_p = -k_pM_{\mathrm{p}}\frac{N_{\mathrm{p}}}{N_A}\bar{n} \tag{10.6}$$

Where $\bar{n}$ is the average number of radicals per particle ($\bar{n} = 0.5$) and the constant of overall propagation speed is formulates as

$$k_p = k_p'\exp(-a.f_{\mathrm{MS}}) \tag{10.7}$$

where f_{MS} denotes molar fraction of α–methylstyrene at the beginning of a reaction, k_p' denotes the propagation rate constant for styrene homopolymerization and a is a constant (Fournier 1998). To complete the model it is necessary to express the monomer concentration in particles M_{p}

$$M_{\mathrm{p}} = M_{\mathrm{pc}} = \frac{(1-X_c)\rho_{\mathrm{M}}}{[(1-X_c)+X_c\rho_{\mathrm{M}}/\rho_{\mathrm{P}}]\,M_{\mathrm{M}}} \qquad X \le X_c \tag{10.8a}$$

$$M_{\mathrm{p}} = \frac{(1-X)\rho_{\mathrm{M}}}{[(1-X)+X\rho_{\mathrm{M}}/\rho_{\mathrm{P}}]\,M_{\mathrm{M}}} \qquad X > X_c \tag{10.8b}$$

where M_{pc} stands for critical monomer concentration in particles, X_c represents critical monomer conversion and finally ρ_{M} denotes monomer density. Macromolecules formulation occurs in two processes: during the initiator decomposition and during the radical attempting R_t and transferring to the monomer R_{trM}. Velocities of these processes are defined as

$$R_t = \frac{R_a\bar{n}N_p}{N_p + \frac{S}{\epsilon}} \tag{10.9}$$

$$R_{trM} = k_{trM}M_p\frac{N_p}{N_A}\bar{n} \tag{10.10}$$

Hence we are able to define the total polymer formation velocity as follows

$$\dot{P} = R_t + R_{trM} = \frac{R_a \bar{n} N_p}{N_p + \frac{S}{\epsilon}} + k_{trM} M_p \frac{N_p}{N_A} \bar{n} \tag{10.11}$$

here the monomer transfer velocity constant is defined as

$$k_{trM} = k_{trM_0} \exp\left(-\frac{E_{trM}}{RT}\right) \exp(b f_{MS}) \tag{10.12}$$

10.3 Problem Formulation

At this stage, a model of the copolymerization reaction of emulsion styrene and α–methylstyrene in batch reactor has been developed. This is a simple model, which will be easily used in the steps of optimization and control of the reactor, but retains good predictive ability of the most important variables.

The emulsion polymerization reaction is divided into three stages (Fournier 1998)

- Nucleation:

$$\dot{x}_1 = -R_{\mathrm{p}}(T, x_2) \tag{10.13}$$

$$\dot{x}_2 = \frac{R_a(T) N_A}{1 + \frac{\epsilon x_2}{S(x_1, x_2)}} \tag{10.14}$$

$$\dot{x}_3 = R_{\mathrm{t}}(T, x_1, x_2) + R_{\mathrm{trM}}(T, x_2) \tag{10.15}$$

  Where the propagation velocity stands for $R_p(T, x_2) = k_p(T) M_{pc} \frac{x_2}{N_A} \bar{n}$ and monomer transfer velocity denotes $R_{trM} = k_{trM}(T) M_{pc} \frac{x_2}{N_A} \bar{n}$.

- Particles growth is same as first step except:

$$\dot{x}_2 = 0 \tag{10.16}$$

- The concentration decrease

$$\dot{x}_1 = -R_{\mathrm{p}}(T, x_1, x_2) \tag{10.17}$$

$$\dot{x}_2 = 0 \tag{10.18}$$

$$\dot{x}_3 = R_{\mathrm{t}}(T, x_1, x_2) + R_{\mathrm{trM}}(T, x_1, x_2) \tag{10.19}$$

  Propagation velocity stands for $R_p(T, x_1, x_2) = k_p(T) M_p(x_1) \frac{x_2}{N_A} \bar{n}$, monomer concentration is expressed as $M_p(x_1) = \frac{(1 - X)\rho_M}{[(1 - X) + X\rho_M/\rho_P] M_M}$ and monomer transfer velocity is characterized as $R_{trM} = k_{trM}(T) M_p(x) 1) \frac{x_2}{N_A} \bar{n}$.

The initial conditions are given by the composition of the reactor at t_0

$$x_1(t_0) = M_0$$

$$x_2(t_0) = 0$$

$$x_3(t_0) = 0$$

Our aim is to reach the desired final state that is characterized by conversion X_f (which characterizes the amount of formed polymer) and by final number average molecular weight $M_n(tf)$ which are defined as

$$X(t_f) = 1 - \frac{x_1(t_f)}{M_0} \tag{10.20}$$

$$M_n(t_f) = M_M \frac{M_0 - M(t_f)}{P(t_f)} \tag{10.21}$$

Assume a model described by the Eqs. 10.1 and 10.6. Our aim is to find such an optimal control trajectory which minimizes the final time of the reaction thus the optimization problem can be expressed as

$$\min_{u(t)} \; J = t_f \tag{10.22}$$

With respect to the following limit

$$0 \leq t_f \leq 10000$$

$$313.15\,\mathrm{K} \leq T \leq 343.15\,\mathrm{K}$$

10.4 Solving the Optimization Problem Using SNOPT and CVP_SS

The optimization problem was solved by using SNOPT and CVP_SS. SNOPT(Sparse Nonlinear Optimizer) is a NLP solver based on sparse sequential quadratic programming (SQP)(see K.Holmstrom (2008)) used in MATLAB. Integration was provided by *ode45* integrator based on 4th order Runge-Kutta method. The second used solver was CVP_SS which stands for control vector parametrization - single shooting algorithm used in gPROMS. To solve this problem we used the method of control vector parametrization, where the needed gradients where computed by sensitivity equations with same initial estimates and same end-point constraints $X_f = 0.6$ and $M_{nf} = 3 \times 10^{-6}[\mathrm{g.mol}^{-1}]$ for both cases. Obtained results are summarized and compared in Tab. 10.1. Here N_I stands for the number of control intervals. Our aim was to minimize the value of the objective function, in which case it meant to minimize the

final time. As we can see an increasing number of time intervals caused the decrease of the objective function. This is natural, because the increase of optimized time intervals caused an increase in number of degrees of freedom, and thus enabled a better approximation. We can also see that the best value was reached by using the CVP_SS solver. Further increase in number of intervals caused only minor differences in the value of the objective function.

Table 10.1: Comparison of resulting objective function for different numbers of intervals of piece-wise constant control and different NLP solvers.

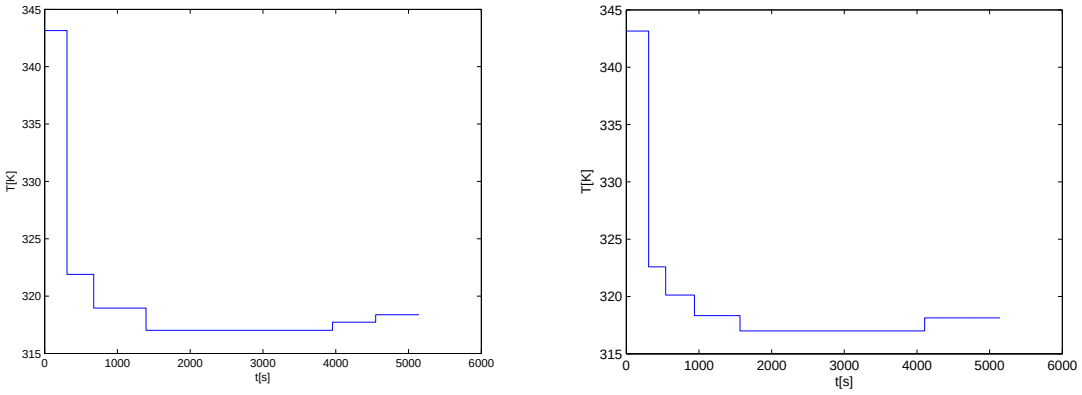
N_I	X_f	$M_{nf} \times 10^{-6}$	SNOPT: t_f[s]	CVP_SS: t_f[s]
1	0.6	3.0	7924.10	7924.09
2	0.6	3.0	5158.10	5158.10
3	0.6	3.0	5156.90	5158.06
4	0.6	3.0	5147.83	5146.70
5	0.6	3.0	5146.10	5146.56
6	0.6	3.0	5144.30	5143.84
7	0.6	3.0	5144.11	5143.27

In Tab. 10.2 we compare other proprieties such as computing time *CPU* and number of NLP iterations *#it*. If we compare the computing time we can see that CVP_SS solver reached the optimal values much faster than SNOPT. Further, if we compare number of iterations it can be seen that for the higher number of iterations SNOPT less iterations that CVP_SS. In conclusion we can say that for this case CVP_SS solver seemed to be more appropriate because it reached smaller value of the objective function within shorter time.

Graphical comparison of optimal control trajectories, obtained by using different NLP solvers for the case of $N_I = 6$ is shown in Fig. 10.1. As we can see there are only minor differences between optimal control trajectory obtained by using SNOPT (Fig. 10.1(a)) and trajectory obtained by CVP_SS (Fig. 10.1(b)).

Table 10.2: Comparison of computational aspects for different numbers of intervals of piecewise constant control and different methods for computing gradients.

	SNOPT		CVP_SS	
N_I	CPU [s]	#it	CPU [s]	#it
1	42.3963	22	3.6660	78
2	596.0426	16	5.5540	13
3	1552.2000	45	20.9510	44
4	1013.7000	168	37.9710	79
5	1825.1000	46	40.0210	80
6	1014.5000	48	36.0670	73
7	823.8831	40	51.8700	95



(a) SNOPT (b) CVP_SS

Figure 10.1: Optimal control trajectories for $N_I = 6$ using different NLP solvers.

Chapter 11

Conclusion

The aim of this diploma thesis was to study dynamic optimization of processes. Several analytical and numerical methods were presented. We explained the base of Pontryagin's minimum principle, dynamic programming and variational calculus. Further, we explained two numerical methods - orthogonal collocation and control vector parametrization. Within control vector parametrization we defined three gradient computing approaches. The last section of the theoretical part was dedicated to hybrid processes.

In the second part we discussed concrete problems. The first discussed problem was the sucrose hydrolysis, where the dynamic optimization problem was solved separately for batch and for continuous system. Control vector parametrization was the used as problem-solving method. Needed gradients were computed by finite differences and also by sensitivity equations. The second mentioned process was the catalytic cracking of gas oil. The optimization problem was solved by control vector parametrization, where the gradients were computed by finite differences approach and also by sensitivity equations. The method of orthogonal collocation was explained for the case of simple batch reactor. This optimization problem was also solved by control vector parametrization. Here the method of sensitivity equations shown to be more accurate.

The last considered problem was an emulsion polymerization reaction. In this case we had to keep in mind that this process is a hybrid process, thus there may occur discontinuities. The optimization problem was solved by using the method of control vector parametrization and the needed gradients were computed by sensitivity equations. The problem was implemented in MATLAB and gPROMS too. All results are summarized and discussed at the end of each chapter.

Chapter 12

Resumé

Každý deň riešime problémy, na ktoré sa snažíme nájst' optimálne riešenie. Vo všeobecnosti môžeme povedat', že optimálne riešenie je najlepšie riešenie daného problému za daných podmienok.

Nájst' najlepšie riešenie znamená nájst' matematické riešenie optimalizačného problému. V chemickom priemysle sa bežne stretávame s optimalizačnými úlohami, kde naším cieľom nie je iba vyrobit' maximálne množstvo požadovaného produktu, ale aj minimalizovat' celkové náklady, minimalizovat' spotrebu surového materiálu a energií a rôzne iné faktory. Takéto problémy rieši dynamická optimalizácia. Presnejšie, dynamická optimalizácia sa snaží nájst' taký optimálny priebeh riadenia, ktorý by minimalizoval alebo maximalizoval hodnotu danej účelovej funkcie pri dodržaní obmedzení.

Riešenie problémov dynamickej optimalizácie pozostáva z niekoľkých krokov. Prvým krokom je definícia problému. K tomu, aby sme mohli hocijaký problém vyriešit', musíme presne vymedzit' náš cieľ, teda čo budeme minimalizovat' resp. maximalizovat' a za akých podmienok a obmedzení. Ďalším krokom je opis procesu. Vo všeobecnosti platí, že čím je model procesu lepší, tým lepšie riešenie vieme nájst'. Posledným krokom je použitie optimalizačného algoritmu na daný optimalizačný problém.

V súčasnosti existuje niekoľko metód používaných pri riešení problémov dynamickej optimalizácie. Vo všeobecnosti ich môžeme rozdelit' do dvoch skupín - do skupiny analytických a do skupiny numerických metód. Medzi analytické metódy radíme Pontryaginov princíp minima, dynamické programovanie a variačný počet. Skupinu numerických metód d'alej rozdeľujeme na priame a nepriame numerické metódy. Do skupiny priamych numerických metód radíme sekvenčnú a simultánnu metódy, zatiaľ čo do skupiny tých nepriamych patrí iterácia hraničnej podmienky a parametrizácia vektora riadenia.

Táto diplomová práca je rozdelená do dvoch častí. Prvá časť je venovaná teoretickým poznatkom, kým v druhej, aplikačnej časti sa venujeme riešeniu konkrétnych optimalizačných problémov.

Prvá kapitola teoretickej časti pojednáva o dynamickej optimalizácii. K tomu, aby sme mohli vyriešiť problém dynamickej optimalizácie, alebo aj problém optimálneho riadenia pri otvorenej slučke, musíme najprv vytvoriť matematický model riadeného procesu, definova v t obmedzenia a účelovú funkciu. Pri vytváraní modelu procesu, by sme mali snažiť o vytvorenie čo najjednoduchšieho matematického opisu, ktorý by adekvátne predpovedal odozvu systému na všetky predpokladané vstupy. Pri hľadaní optimálneho riešenia musíme mať na mysli aj obmedzenia, ktoré dané systém nejakým spôsobom limitujú. Účelová funkcia je prediktíva, na základe ktorej sa inej funkcii priradí reálne číslo. Môže byť vyjadrená v troch základných tvaroch - Bolzov, Mayerov a Langrangeov tvar.

Druhá kapitola je venovaná analytickým a numerickým metódam používaných pri riešení optimalizačných problémov. Prvou analytickou metódou, ktorej sa venujeme, je dynamické programovanie. Táto metóda je založená na Bellmanovom princípe optimality, ktorý hovorí,že optimálna trajektória závisí len od počiatočných podmienok a od cieľa a nie od cesty, ktorou sme tento cieľ dosiahli. Veľmi podobnou metódou je Pontryaginov princíp minima, ktorá sa používa na riešenie komplexnejších problémov s obmedzeniami na stavové a riadiace veličiny. Poslednou študovanou analytickou metódou je variač ný počet, ktorý je v tejto práci odvodený z Bellmanovho princípu optimality. Medzi priame numerické metódy radíme sekvenčnú metódu, známu aj pod názvom parametrizácia vektora riadenia a simultánnu metódu, známu aj ako kompletná parametrizácia. Metódy ako iterácia hraničnej podmienky a iterácia vektora riadenia radíme do skupiny nepriamych numerických metód.

V tretej teoretickej kapitole sa zaoberáme metódou ortogonálnej kolokácie na konečných prvkoch. Táto metóda je založená na kompletnej parametrizácii stavových aj riadiacich profilov. To znamená, že pôvodná stavová a riadiaca trajektória je aproximovaná lineárnou kombináciou bázických funkcií. Naším cieľom je potom nájsť optimálne riadenie optimalizáciou koeficinetov týchto funkcií.

Druhou skúmanou numerickou metódou je parametrizćia vektora riadenia, ktorá je založená na nahradení pôvodnej spojitej trajektórie riadenia inou trajektóriou, ktorá sa dá opísať konečným počtom parametrov. To znamená, že pôvodne spojitý prob-

lém dynamickej optimalizácie bude nahradený statickým problémom nelineárneho programovania. Gradienty potrebné pre riešenie problémov dynamickej optimalizácie môžeme vypočítať niekoľkými spôsobmi. Prvou metódou je metóda konečných diferencií. Podtstatou tejto metódy je rekurentná integrácia systému so zmenou hodnoty niektorého z optimalizovaných parametrov. Najväčšou výhodou tejto metódy je, že nevyžaduje pridanie ďalšej diferenciálnej rovnice, na druhú stranu, celý systém musí byť pri každej zmene každého parametra integrovaný n_p–krát, kde n_p označuje počet parametrov. Druhou metódou vhodnou na výpočet gradientov, je metóda adjungovaných premenných. Podstata tejto metódy spočíva vo vytvorení adjungovaného systému, ktorý sa potom integruje spätne, zatiaľ čo pôvodný systém sa integruje dopredne. Poslednou analyzovanou gradientovou metódou je metóda citlivostných rovníc. Tá to metóda sa používa najmä pre systémy, ktoré sú opísané veľkým počtom diferenciálnych rovníc, avšak s malým počtom opimalizovaných premenných, pretože počet citlivostných rovnív závisí nielen od počtu stavových premenných, ale aj od počtu optimalizovaných parametrov.

Poslednou kapitolou teoretickej časti sú parametrické citlivosti hybridných systémov. Dôležitou charakteristikou hybridných systémov je, že ich dynamické vlastnosti sa menia v závislosti od fázy, v ktorej sa práve nachádzajú. Citlivostné funkcie sú definované ako parciálne derivácie premenných systému s ohľadom na optimalizované parametre. Parametrické citlivosti sa používajú na sledovanie vplyvu zmeny parametrov na riešenie.

V druhej časti diplomovej práce sa venujeme riešeniu konkrétnych optimalizačných problémov. Prvým príkladom odhadu parametrov je hydrolýza sacharócy pomocou invertázy. Budeme rozlišovať vsádzkový a prietokový režim. Našou úlohou bude v oboch prípadoch hľadať optimálne hodnoty kinetických parametrov a minimalizovať hodnotu účelovej funkcie, teda minimalizovať sumu štvorcov rozdielov nameraných koncentrácií a koncentrácií predikovaných modelom. Tento optimalizačný problém sme riešili pomocou parametrizácie vektora riadenia. Potrebné gradienty sme vypočítali pomocou metódy konečných diferencií a aj pomocou citlivostných rovníc. Výsledky získané pomocou konečných diferencií boli o niečo lepšie ako tie, ktoré sme vypočítali pomocou citlivostných rovníc. Toto mohlo byť spôsobené tým, že daný systém je nelineárny, a teda vykazuje niekoľko lokálnych extrémov.

Druhým riešeným príkladom bolo katalytické krakovanie. V tomto prípade je systém opísaný dvoma diferenciálnymi rovnicami. Našou úlohou bolo nájsť také optimálne

hodnoty troch kinetických parametrov, ktoré by minimalizovali sumu štvorcov rozdielov nameraných a modelom predikovaných molárnych frakcií reaktanta a medziproduktu. Na vyriešenie problému odhadu parametrov sme použili parametrizáciu vektora riadenia a potrebné parametre sme vypočítali najprv metódou konečných diferencií a potom pomocou citlivostnćych rovníc. Hodnota účelovej funkcie bola rovnaká v oboch prípadoch a aj medzi optimálnymi hodnotami kinetických parametrov boli len malé rozdiely. Tieto malé rozdiely optimálnych hodnôt boli vďaka tomu, že v tomto prípade sa jednalo o relatívne jednoduchý príklad, a teda aj metóda konečných diferencií bola postačujúca.

Tretí príklad riešil problém optimálneho riadenia v sádzkovom reaktore s následnou reakciou. Našou úlohou je nájsť taký optimálny teplotný profil vo vnútry reaktora, ktorý by maximalizoval koncentráciu medziproduktu na konci celého procesu. Optimalizačný problém sme najprv riešili pomocou ortogonálnej kolokácie. V tomto prípade sme hľadali hodnoty piatich kolokačných bodov pre riadiacu veličinu a dvoch kolokačných bodov pre stavové veličiny. Druhou použitou metódou bola parametrizácia vektora riadenia, kde sme potrebné gradienty vypočítali pomocou citlivostných rovníc. Metóda citlivostných rovníc sa v tomto prípade ukázala byť vhodnejšou, pretože optimálne hodnoty dosiahla s menším počtom iterácií.

Posledným študovaým procesom bola emulzná polymerizácia styrénu a α–metylstyrénu. V tomto prípade sa jedná o hybridný systém, pretože celá reakcia prebieha v troch etapách - nukleácia, rast častíc a pokles koncentrácie. Každá jedna fáza je opísaná troma diferenciálnymi rovnicami, avšak pravé strany týchto rovníc sa menia podľa fázy, v ktorej sa práve nachádzajú. Našou úlohou je nájsť takú optimálnu trajektóriu riadenia, ktorá by minimalizovala hodnotu koncového času. Problém bol riešený pomocou citlivostných rovníc pomocou dvoch odlišných NLP solverov - SNOPT a CVP_SS. CVP_SS solver sa v tomto prípade ukázal byť vhodnejším, pretože dosiahol lepšie hodnoty účelovej funkcie, a to v kratšom čase.

Cieľom tejto diplomovej práce bolo oboznámiť sa s problematikou dynamickej optimalizácie. V teoretickej rovine sme vysvetlili podstatu dynamického programovania, Pontryaginovho princípu minima a variačného počtu. Čo sa týka numerických metód, bližšie sme si vysvetlili ortogonálnu kolokáciu na konečných prvkoch a parametrizáciu vektora riadenia. V druhej časti práce sme si ukázali aplikáciu týchto dvoch numerických metód na niekoľkých príkladoch.

Bibliography

B. Chachuat, A. B. Singer, and P. I. Barton. Global methods for dynamic optimization and mixed-integer dynamic optimization. *Ind. Eng. Chem. Res.*, 45(25):8373–8392, 2006.

W. F. Feehery. *Dynamic Optimization with Path Constraints*. PhD thesis, Massachusetts Institute of Technology, June 1998.

M. Fikar. *Dynamická optimalizácia procesov*. STU in Bratislava and Slovenská akadémia, n.o., Bratislava, 2007.

F. Fournier. *Méthodologie d'optimisation dynamique et de commande optimale des réacteurs électrochimiques discontinus*. PhD thesis, ENSIC-INPL Nancy, 1998.

T. Hirmajer. *Dynamická optimalizácia procesov s hybridnou dynamikou*. PhD thesis, Oddelenie informatizácie a riadenia procesov, FCHPT STU, Radlinského 9, 812 37 Bratislava, Slovenská republika, 10.1.2007 2007.

D. G. Hull. *Optimal Control Theory for Applications*. Mechanical Engineering Series. Springer-Verlag New York, 2003.

M.M. Edvall K.Holmstrom, A.O. Goran. User's guide for tomlabsnopt. pages 171–210, 2008.

D. E. Kirk. *Optimal Control Theory: An Introduction*. Prentice-Hall, London, 1970.

J. S. Logsdon and L. T. Biegler. Accurate solution of differential-algebraic optimization problem. *Ind. Eng. Chem. Res.*, 28:1628–1639, 1989.

R. Paulen. Global optimization of processes. Master's thesis, Faculty of Chemical and Food Technology, Radlinskeho 9, 812 37 Bratislava, Slovak Republic, 2008.

R. Paulen, M. Fikar, and M. A. Latifi. Dynamic optimization of a hybrid system emulsion polymerization reaction. *Journal of Cybernetics and Informatics*, (9):71–78, 2010.

L. Petzold, S. Li, Y. Cao, and R. Serban. Sensitivity analysis of differential-algebraic equations and partial differential equations. 2006.

J. Rajesh, K. Gupta, H. S. Kusumakar, V. K. Jayaraman, and B. D. Kulkarni. Dynamic optimization of chemical processes using ant colony framework. *Computers and Chemistry*, 25:583–595, 2001.

M. Čižniar. Dynamic optimisation of processes. Master's thesis, Faculty of Chemical and Food Technology, Radlinskeho 9, 812 37 Bratislava, Slovak Republic, 2005.

Š. Ševčík. Optimálne riadenie dynamických systémov. Master's thesis, Faculty of Chemical and Food Technology, Radlinskeho 9, 812 37 Bratislava, Slovak Republic, 2005.