CHALMERS

Comparative Study of Numerical Methods for Optimal Control of a Biomechanical System

Controlled Motion of a Human Leg during Swing Phase

International Master's Programme Solid and Fluid Mechanics

ANDREAS DRAGANIS, CARL SANDSTRÖM

Department of Applied Mechanics
Division of Dynamics, Division of Material and Computational Mechanics
CHALMERS UNIVERSITY OF TECHNOLOGY
Göteborg, Sweden, 2009
Master's Thesis 2009:24

MASTER'S THESIS 2009:24

Comparative Study of Numerical Methods for Optimal Control of a Biomechanical System

Controlled Motion of a Human Leg during Swing Phase

ANDREAS DRAGANIS, CARL SANDSTRÖM

Department of Applied Mechanics
Division of Dynamics, Division of Material and Computational Mechanics
Göteborg, Sweden 2009

Comparative Study of Numerical Methods for Optimal Control of a Biomechanical System
Controlled Motion of a Human Leg during Swing Phase

ANDREAS DRAGANIS, CARL SANDSTRÖM

Master's Thesis 2009:24
ISSN 1652-8557
Department of Applied Mechanics
Division of Dynamics, Division of Material and Computational Mechanics
Chalmers University of Technology
SE-412 96 Göteborg
Sweden
Telephone: +46 (0)31-772 1000

Chalmers Reproservice
Göteborg, Sweden 2009

Abstract

One type of optimal control problem for a mechanical system is the problem of steering the system from an initial state to a target state while minimizing a chosen objective function, which measures a certain feature of the system. A typical example of this is minimizing the energy and/or time consumption of an industrial assembly robot as it performs a certain task.

In this contribution we investigate three methods for finding the optimal control of a biomechanical system relevant to human walking. The system at hand is a simplified model of a human leg during the process of walking. The leg is modeled as a double pendulum with control moments applied at the hip and knee joints. As the objective function pertinent to the optimization problem, a combination of two different measures of energy consumption are considered, one of which smooth and the other non-smooth. The considered optimization parameters, which the objective function is minimized with respect to, are the parameters involved in the discretizations of the free variables of the optimal control problem.

Three different numerical methods are employed for the discretization and solution of the two-point boundary value problem for the dynamic system in question; a temporal finite element based approach, one based on Fourier series approximations of the generalized coordinates and inverse dynamics and one based on Matlabs built-in functions for numerical solution of ordinary differential equations: `ode45`. For the considered biomechanical system, several optimal control problems are solved using the aforementioned numerical methods together with a general-purpose constrained optimization tool (Matlabs subroutine `fmincon`). In this way, the temporal finite element method and `ode45`, actually being initial value problem solvers, solve the two point boundary value problem by way of a kind of "shooting method". The results obtained using the three methods are analyzed and subsequently compared with respect to both computational efficiency and kinematic, dynamic and energetic characteristics of the optimal motion.

The numbers of optimization parameters deemed necessary for a sufficiently good solution of the optimal control problem for the tolerance settings used were 18 for both the `ode45` method and the temporal finite element method and 14 for the Fourier method. Even for these settings, the Fourier method produced better results: solutions corresponding to lower energies. The characteristics of the solutions thus obtained were very similar between the methods.

Keywords: Optimal control, Trajectory planning, Biomechanics, Direct dynamics, Inverse dynamics.

Contents

Preface

This Master's Thesis was carried out as a cross divisional project between the Division of Dynamics and the Division of Material and Computational Mechanics at the Department of Applied Mechanics as the final part of the Master's programme Solid and Fluid Mechanics on Chalmers University of Technology during the spring of 2009.

The work presented herein implements and evaluates three numerical approaches for solving an optimal control problem involving a biomechanical system. Specifically, the considered system is a model of a human leg during the swing phase of a step. No particular division of the work load required for the project was made so both authors were equally involved in every part of the process.

Since this is a cross divisional project, one supervisor and examiner from each division was involved. From the Division of Dynamics, Professor Viktor Berbyuk participated as the examiner of Andreas Draganis, while Håkan Johansson from the Division of Material and Computational Mechanics participated as the examiner of Carl Sandström. We would like to thank both of them for their support and valuable guidance throughout. We would further like to thank the Department of Applied Mechanics for financial support.

Göteborg, June 2009.
Andreas Draganis and Carl Sandström.

Chapter 1

Introduction

In this contribution, a number of approaches to finding the optimal control of a biomechanical system with respect to some measure of energy consumption are investigated. The considered biomechanical system is a simple model of a human leg in the process of walking, specifically: in the swing-phase of a step. The three methods for solving the optimal control problem are a temporal finite element based approach [7], one based on Fourier series approximations of the generalized coordinates and inverse dynamics [13] and one based on Matlabs built-in functions for numerical solution of ordinary differential equations (ODEs): `ode45` [3]. The latter method is included more for reference than for its theoretical interest. The discrete optimization problem connected to each of these methods is solved using a general-purpose optimization tool (Matlabs subroutine `fmincon`) [2].

The numerical solution of an optimal control problem requires some means of translating the continuous problem into a discrete optimization problem. Previous choices of methods used in this context for discretizing the free variables of the system include a spline-GA (genetic algorithm) method [8] and a Fourier series based method similar to the one considered in this thesis [10], [14], [13]. The temporal finite element method has previously been explored in an optimal control context, for instance in [4], [5], [6], but not quite, to the authors knowledge, in the particular manner considered in this project.

The purpose of the project is chiefly to investigate whether the temporal finite element method and the Fourier method are suitable and efficient for solving a problem of the given type. The temporal finite element method is interesting due to the fact that the control space and the state space are discretized separately. This enables implementation of schemes for local mesh refinement and facilitates effective error analysis. The Fourier method, its way of discretizing the variables of the system however not being as intuitively attractive, holds potential in that it enables, by inverse dynamics, an analytical solution of the equations of motion.

The considered mechanical system is shown in Figure 1.1. Lengths, masses and other properties of the model will later be chosen so as to simulate the physical characteristics of a human leg. The foot is modeled as a point mass attached to the end of the shank. The limb angles $\theta_1(t)$ and $\theta_2(t)$ are the free variables of the system, while the torques $u_1(t)$ and $u_2(t)$ applied at the hip joint and at the knee joint, respectively, constitute the control. Note that these torques are not externally applied, u_1 is acting between the upper body and the thigh and u_2 is acting between the two limbs of the leg. The swing phase consists of the part of the step between the instant of time when the foot leaves the ground and the instant it reaches the ground again. The motion of the hip will be prescribed.

H, K and A signify the hip, knee and ankle joints, respectively, while the thigh is referred to as Body 1 and the shank as Body 2. m_1, m_2, m_H and m_A represent the masses of the respective bodies. a_1 and a_2 represent the length of the limbs. r_1 is the distance from the hip joint to center of mass of the thigh and r_2 is the distance from the knee joint to the center of mass of the shank. $\bar{J}_1$ and $\bar{J}_2$ is the moment of inertia about the center of mass (hence the bar over the symbol) of the respective limbs. $t_0 = 0$ is the initial time of the motion and t_f is the final time. L is the total length of one step. $x(t)$ and $y(t)$ are the coordinates of the hip.

Boundary conditions corresponding to the swing phase of a step are imposed on the system (see Section 2.1.3 for more details):

- Initial and terminal thigh and shank angles are fixed
- Initial angular velocities of the thigh and shank are fixed.

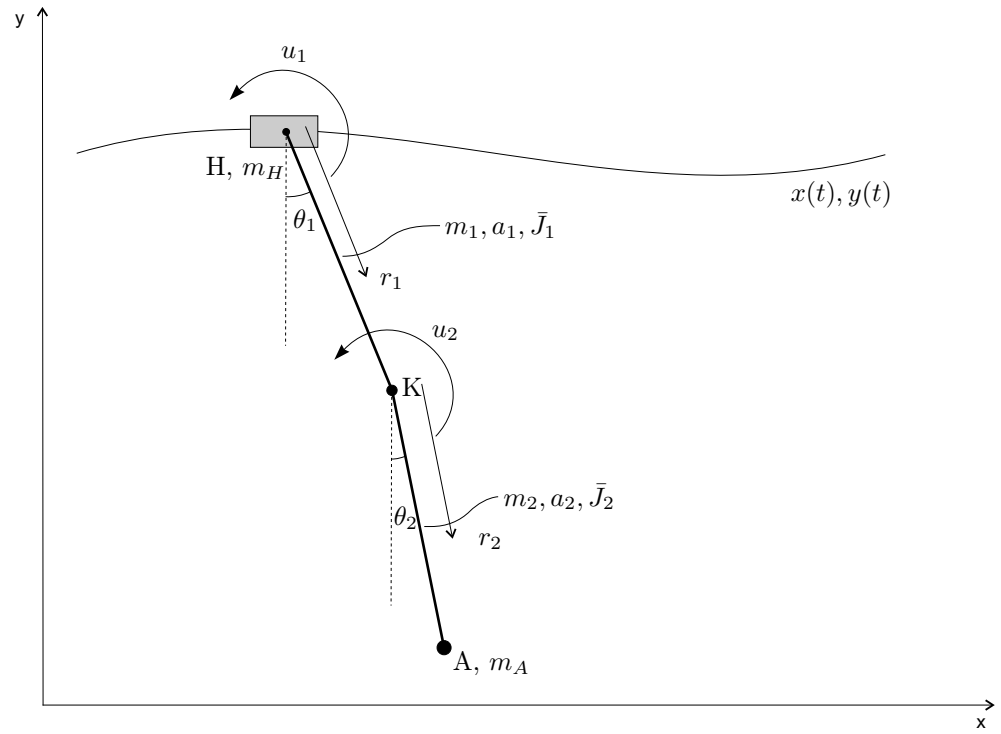

Figure 1.1: Schematic sketch of the considered model of a human leg.

Some constraints required for realistic human motion are also imposed (see Section 2.1.4 for more details):

- The knee can only be bent one way
- The angle of the shank has a lowest possible angle
- The foot must be above the ground at all times.

Chapter 2

Mathematical model

2.1 Statement of the optimal control problem

2.1.1 General statement of an optimal control problem

Consider a mechanical system whose motion can be described by the following system of equations:

$$\dot{\boldsymbol{x}}(t) = f(\boldsymbol{x}(t), \boldsymbol{u}(t)), \quad t \in [0,\ T], \tag{2.1}$$

where $\boldsymbol{x}(t) = (x_1, x_2, \ldots, x_n)$ is a state vector (which can contain both position coordinates and velocities), $\boldsymbol{u}(t) = (u_1, u_2, \ldots, u_m)$ is a vector of control stimuli (forces or torques) and $[0,\ T]$ is the time domain under consideration. It is further required that the state and control variables satisfy the following constraints:

$$\boldsymbol{h}(\boldsymbol{x}(t), \boldsymbol{u}(t)) = 0 \tag{2.2}$$

$$\boldsymbol{g}(\boldsymbol{x}(t), \boldsymbol{u}(t)) \leq 0. \tag{2.3}$$

The system of equations (2.1) and the constraints (2.2) – (2.3) constitute the mathematical model of the mechanical system. Note that the equality constraints (2.2) can include boundary conditions for the state and/or control variables.

Upon introducing the scalar objective functional

$$E = E[\boldsymbol{x}(t), \boldsymbol{u}(t)],$$

a function of the motion and of the control, the optimal control problem can be formulated as: given a mechanical system described by the mathematical model (2.1), find the motion $\boldsymbol{x}^*(t)$ and the control $\boldsymbol{u}^*(t)$ which satisfy the constraints (2.2) – (2.3) and also minimize the given objective functional $E\left[\boldsymbol{x}(t), \boldsymbol{u}(t)\right]$.

2.1.2 Equations of motion

In the present section, a mathematical model of the considered system (Figure 1.1) is derived using Lagrangian mechanics (the overall methodology of which is described in, for instance, Goldstein et. al. [1]). The symbolic mathematics software Mathematica has been used to perform the analytical manipulations. "(t)" is dropped for brevity in the following.

In the Lagrangian formulation of mechanics, the Lagrangian is first constructed. In this case, it is

$$L = T_1 + T_2 + T_H + T_A - (V_1 + V_2 + V_H + V_A), \tag{2.4}$$

where the first four terms represent the kinetic energies and the last four terms the potential energies of the respective bodies of the system. The generalized forces Q_q are then expressed, enabling the equations of motion to be set up as a system of equations, each one of the following form:

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}}\right) - \frac{\partial L}{\partial q} = Q_q, \tag{2.5}$$

where q is a generalized coordinate with corresponding generalized force Q_q.

The expressions for the kinetic energy of the different bodies of the system are

$$T_1 = \frac{1}{2} m_1 |\bar{\boldsymbol{v}}_1|^2 + \frac{1}{2} \bar{J}_1 \dot{\theta}_1^2 \tag{2.6}$$

$$T_2 = \frac{1}{2}m_2|\bar{\boldsymbol{v}}_2|^2 + \frac{1}{2}\bar{J}_2\dot{\theta}_2^2 \tag{2.7}$$

$$T_H = \frac{1}{2}m_H|\bar{\boldsymbol{v}}_H|^2 \tag{2.8}$$

$$T_A = \frac{1}{2}m_A|\bar{\boldsymbol{v}}_A|^2. \tag{2.9}$$

The expressions for the potential energy of the different bodies of the system are

$$V_1 = m_1 g\bar{\boldsymbol{r}}_1\hat{x} \tag{2.10}$$

$$V_2 = m_2 g\bar{\boldsymbol{r}}_2\hat{x} \tag{2.11}$$

$$V_H = m_H g\bar{\boldsymbol{r}}_H\hat{x} \tag{2.12}$$

$$V_A = m_A g\bar{\boldsymbol{r}}_A\hat{x}, \tag{2.13}$$

where $\hat{x}$ is the unit vector in the x-direction. The expressions for the position vectors of the centers of mass of the different bodies are shown in equations (2.14) – (2.17). The first equation of these shows how the parametrization of the hip motion is chosen.

$$\boldsymbol{r}_H = \begin{bmatrix} x(t) \\ y(t) \end{bmatrix} = \begin{bmatrix} V(t-t_0) + B_1\sin(2\omega(t-t_0)) \\ h_0 + B_2\sin(2\omega(t-t_0)) \end{bmatrix} \tag{2.14}$$

$$\bar{\boldsymbol{r}}_1 = \begin{bmatrix} x + \sin\theta_1 r_1 \\ y - \cos\theta_1 r_1 \end{bmatrix} \tag{2.15}$$

$$\bar{\boldsymbol{r}}_2 = \begin{bmatrix} x + \sin\theta_1 a_1 + \sin\theta_2 r_2 \\ y - \cos\theta_1 a_1 - \cos\theta_2 r_2 \end{bmatrix} \tag{2.16}$$

$$\bar{\boldsymbol{r}}_A = \begin{bmatrix} x + \sin\theta_1 a_1 + \sin\theta_2 a_2 \\ y - \cos\theta_1 a_1 - \cos\theta_2 a_2 \end{bmatrix}. \tag{2.17}$$

The velocity vectors are simply the time derivatives of the position vectors: $\boldsymbol{v} = \frac{d}{dt}\boldsymbol{r}$.

The generalized forces can be identified from the expression for the virtual work δW performed under virtual displacements of all generalized coordinates. Note that the angle associated with the torque u_2 is the difference between the two limb angles, since u_2 is acting between the two limbs.

$$\delta W = u_1\delta\theta_1 + u_2(\delta\theta_2 - \delta\theta_1) = (u_1 - u_2)\delta\theta_1 + u_2\delta\theta_2.$$

The following generalized forces are thus identified: $Q_1 = u_1 - u_2$, $Q_2 = u_2$. Performing the necessary symbolic manipulations, the resulting equations of motion are

$$\begin{gathered} \sin(\theta_1-\theta_2)a_1(a_2m_A + m_2r_2)(\dot{\theta}_2)^2 + g\sin\theta_1(a_1(m_2+m_A) + m_1r_1) + \\ \cos\theta_1(a_1(m_2+m_A) + m_1r_1)\ddot{x} + \sin\theta_1(a_1(m_2+m_A) + m_1r_1)\ddot{y} + J_1\ddot{\theta}_1 \\ +a_1(a_1(m_2+m_A)\ddot{\theta}_1 + \cos(\theta_1-\theta_2)(a_2m_A + m_2r_2)\ddot{\theta}_2) - u_1(t) + u_2(t) = 0 \end{gathered} \tag{2.18}$$

$$\begin{gathered} (a_2m_A + m_2r_2)(-\sin(\theta_1-\theta_2)a_1(\dot{\theta}_1)^2 + \cos\theta_2\ddot{x} + \\ \sin\theta_2(g+\ddot{y}) + \cos(\theta_1-\theta_2)a_1\ddot{\theta}_1) + (m_Aa_2^2 + J_2)\ddot{\theta}_2 - u_2(t) = 0. \end{gathered} \tag{2.19}$$

$\ddot{\theta}_1$ and $\ddot{\theta}_2$ appear as linear terms in the equations. Solving for these enables the system of equations to be rewritten in the form

$$\begin{aligned} \ddot{\theta}_1 &= g_1(\theta_1, \theta_2, \dot{\theta}_1, \dot{\theta}_2, u_1, u_2) \\ \ddot{\theta}_2 &= g_2(\theta_1, \theta_2, \dot{\theta}_1, \dot{\theta}_2, u_1, u_2). \end{aligned} \tag{2.20}$$

Performing the substitutions $\hat{\theta}_1 = \dot{\theta}_1$, $\hat{\theta}_2 = \dot{\theta}_2$ transforms the above equations into a system of four first order differential equations:

$$\begin{cases} \dot{\theta}_1 = \hat{\theta}_1 \\ \dot{\hat{\theta}}_1 = g_1(\theta_1, \theta_2, \hat{\theta}_1, \hat{\theta}_2, u_1, u_2) \\ \dot{\theta}_2 = \hat{\theta}_2 \\ \dot{\hat{\theta}}_2 = g_2(\theta_1, \theta_2, \hat{\theta}_1, \hat{\theta}_2, u_1, u_2). \end{cases} \tag{2.21}$$

2.1.3 Boundary conditions

The boundary conditions imposed on the system, representing initial and final values for the limb angles, mentioned in Chapter 1, are:

Table 2.1: Boundary conditions for the motion

Parameter	Value at $t = 0$	Value at $t = t_f$
$\theta_1(t)$	θ_{10}	θ_{1f}
$\theta_2(t)$	θ_{20}	θ_{2f}
$\dot{\theta}_1(t)$	$\dot{\theta}_{10} = 0$	$\dot{\theta}_{1f}$, free
$\dot{\theta}_2(t)$	$\dot{\theta}_{20} = 0$	$\dot{\theta}_{2f}$, free

2.1.4 Constraints

The constraints imposed on the system, mentioned in Chapter 1, are in order:

Table 2.2: Constraints on the motion

Constraint	Description
$\theta_2(t) - \theta_1(t) \leq 0$	The opening angle of the knee must be negative.
$\theta_1(t) \geq \theta_{10} - \delta$	The thigh is not allowed to swing too far back.
$y_A(t) \geq 0$	The foot must be above or at ground level at all times.

where δ is just a small number.

2.1.5 The optimal control problem

The optimal control problem is to find the control $u_1(t)$ and $u_2(t)$ satisfying the constraints and the boundary conditions imposed on the system, such that a chosen objective function E is minimized. The objective function E is a function of the motion and in the present case, the following is chosen:

$$E = \lambda_1 E_1 + \lambda_2 E_2, \tag{2.22}$$

where E_1 and E_2 are different measures of energy consumption, given below, and λ_1 and λ_2 are weighting parameters.

$$\begin{aligned} E_1 &= \frac{1}{L} \int_0^{t_f} \Big[\mid u_1(t)\dot{\theta}_1(t) \mid + \mid u_2(t)(\dot{\theta}_1(t) - \dot{\theta}_2(t)) \mid \Big] \, dt \\ E_2 &= \int_0^{t_f} \left[u_1^2(t) + u_2^2(t) \right] dt. \end{aligned} \tag{2.23}$$

E_1 is a non-smooth function of the motion and the control, measuring the mechanical work per meter performed by the control torques $u_1(t)$ and $u_2(t)$ [12]. E_2 is a smooth function of the control only, measuring heat energy loss due to torque generation [8]. It would be possible to keep it simple and just use E_2 as the objective function. However, it was desired to also include the non-smooth energy measure E_1 for the sake of added generality and complexity.

Chapter 3

Numerical solution of the optimal control problem

3.1 Discretization

In order to translate the continuous optimal control problem into a numerical optimization problem, the continuous functions of time representing the free variables of the corresponding mechanical system must be replaced by discrete representations. The set of unknowns of the problem are thus transformed from the values of these functions at each point in time in the considered time interval, to a set of parameters, finite in number.

The numerical optimization problem will be solved using Matlabs general-purpose constrained optimization tool, `fmincon` [2]. `fmincon` attempts to find the minimum of a scalar function of several parameters subject to a set of equality and/or inequality constraints, starting from an initial guess of the parameters. The user thus needs to supply a function that can calculate the value of the objective function E (from the optimization parameters), one that can calculate the values of the equality and/or inequality constraint functions (from the optimization parameters) as well as the initial guess. The optimization method used within `fmincon` is either a "Trust-Region-Reflective" method or an "Active-Set" method, based on the type of constraints used. In the present case, where nonlinear equality and inequality constraints are present, the Active-Set method is used. This method is a sequential quadratic programming (SQP) method, which means solving a quadratic programming (QP) subproblem at each iteration. An updated estimate of the Hessian is computed in each iteration using the Broyden-Fletcher-Goldfarb-Shanno (BFGS) formula. By default, the gradients of the objective function and of equality and inequality constraint functions are computed using finite differences. There is an option, however, to supply `fmincon` with expressions for these gradients and thus improve the speed of the optimization process. This feature will be utilized for the temporal finite element method.

The following three sections describe the solution of the equations of motion for the three respective methods. In each iteration step of the optimizer for the `ode45` based method and for the temporal finite element based method, the time history of the control is given while the limb angles are sought (direct dynamics). In both cases, the control variables are represented by piecewise linear functions, their node values being the optimization parameters, that is the free parameters of the optimization problem. The discretization which is used for the control variables is based on the evenly spaced time mesh having the nodes $t_i,\ i = 0, 1, 2, \ldots, M;\ 0 = t_0 < t_1 < \ldots < t_M = t_f$ and is defined as:

$$u(t) \approx u_h(t) = \sum_{i=0}^{M} u_i Y_i(t) \tag{3.1}$$

where $u_i,\ i = 0, 1, 2, \ldots, M$ are the node values of the discretization and $Y_i(t)$ are shape functions. These are piecewise linear and satisfy the following:

$$Y_i(t) = \begin{cases} 1 & t = t_i \\ 0 & t = t_j, j \neq i \end{cases} \tag{3.2}$$

An example of a function in the form of equation (3.1) is shown in Figure 3.1, together with the shape functions that exist in the considered time domain.

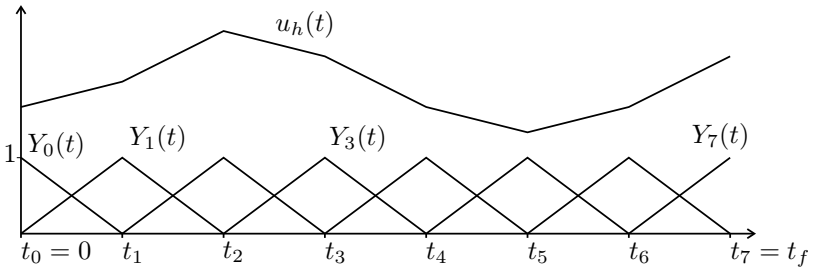

Figure 3.1: An example of a function in the form of equation (3.1) and all shape functions $Y_i(t)$ on an example mesh for which $M = 7$.

Note that, since two control variables are present in the system, the number of optimization parameters in the optimization problem is $2M$.

In each iteration step of the optimizer for the Fourier based method, the limb angles are given and the control is sought (inverse dynamics). The optimization parameters in this case are either the ones involved in the discretization of the time histories of the limb angles or a reduced set, see Section 3.4.

3.2 The `ode45` method

Matlabs `ode45` [3] is a function used for solving initial value problems involving ordinary differential equations. It integrates the system of differential equations $y' = f(y, t)$, given bounds for the time domain and initial conditions for the free variables and their derivatives. `ode45` is based on an explicit Runge-Kutta(4,5) formula and automatically generates a mesh for the independent variable t, the maximum spacing of which can be controlled. The function uses precompiled code, which significantly increases the efficiency of its execution.

When the `ode45` method is used, the function is called in each iteration of the optimizer with information about the time histories of the control. This information is derived from the optimization parameters, which are the node values of the piecewise linear functions used to represent the control variables. These have the form of equation (3.1), as mentioned. The function returns information about the time histories of the limb angles, enabling the computation of the objective function and the constraint functions, which are needed for the optimization process.

Since only initial conditions can be imposed when solving equations using `ode45` and because both initial and terminal conditions are present in the considered problem, the latter need to be imposed elsewhere, necessitating the introduction of these as additional equality constraints in the constrained optimization process performed by `fmincon`. This means that the `ode45` based optimization routine fulfills the terminal conditions by way of a type of *shooting method*.

3.3 The temporal finite element method

The temporal finite element method for optimal control is based on separate discretizations of the state space and the control space. This enables the implementation of efficient local mesh refinement routines. Another promising feature that will be explored later is the possibility to obtain analytical expressions for the gradients of the discretized state variables with respect to the node values of the discretized control variables. This, in turn, enables analytical expressions for the gradients of the objective function and the constraint functions with respect to these node values. These expressions can be used in the optimization process, voiding the need for numerical calculations of these gradients and greatly speeding up the process.

The problem considered here is the state problem arrived at in Section 2.1 minus the terminal conditions for the limb angles: find $\theta_1(t)$, $\theta_2(t)$, $\hat{\theta}_1(t)$ and $\hat{\theta}_2(t)$ for $t \in [0,\ t_f]$ such that:

$$\begin{cases} \dot{\theta}_1 = \hat{\theta}_1 \\ \dot{\hat{\theta}}_1 = g_1(\theta_1, \theta_2, \hat{\theta}_1, \hat{\theta}_2, u_1, u_2) \\ \dot{\theta}_2 = \hat{\theta}_2 \\ \dot{\hat{\theta}}_2 = g_2(\theta_1, \theta_2, \hat{\theta}_1, \hat{\theta}_2, u_1, u_2) \end{cases} \tag{3.3}$$

$$\begin{aligned} \theta_i(0) &= \theta_{i0} \\ \dot{\theta}_i(0) &= \dot{\theta}_{i0} \end{aligned} \tag{3.4}$$

for $i = 1, 2$. Note that it might be that only some of the boundary conditions above are imposed. "(t)" is dropped in the equations above and in what follows.

In the present section, the finite element formulation of the given problem is derived according to the continuous Galerkin method of order 1 (cG(1)) [7]. The cG(1) method means using continuous shape functions of degree 1 and discontinuous test functions of degree 0. We denote the space of piecewise constant functions defined on the interval $[a,\ b]$ by $\mathbb{V}([a,\ b])$.

3.3.1 Weak formulation

Using the evenly spaced times $t_n,\ n = 0, 1, 2, \ldots, N;\ 0 = t_0 < t_1 < \ldots < t_N = t_f$, a division of the total time domain into sub-intervals $t_{n-1} < t < t_n,\ n = 1, 2, \ldots, N$ is possible. In order to derive the variational (weak) formulation of the problem, the equations in (3.3) are first multiplied by test functions: $\psi(t)$, and the result is integrated over the whole temporal domain $[0,\ t_f]$. The integrals are then rewritten as sums of integrals over the temporal sub-intervals. The resulting problem is: find $\theta_1(t)$, $\theta_2(t)$, $\hat{\theta}_1(t)$ and $\hat{\theta}_2(t)$ for $t \in [0,\ t_f]$ such that (3.4) are fulfilled and such that:

$$\begin{cases} \sum_{n=1}^{N} \left\{ \int_{t_{n-1}}^{t_n} (\dot{\theta}_1 - \hat{\theta}_1)\psi dt \right\} = 0 \\ \sum_{n=1}^{N} \left\{ \int_{t_{n-1}}^{t_n} (\dot{\hat{\theta}}_1 - g_1(\theta_1, \theta_2, \hat{\theta}_1, \hat{\theta}_2, u_1, u_2))\psi dt \right\} = 0 \\ \sum_{n=1}^{N} \left\{ \int_{t_{n-1}}^{t_n} (\dot{\theta}_2 - \hat{\theta}_2)\psi dt \right\} = 0 \\ \sum_{n=1}^{N} \left\{ \int_{t_{n-1}}^{t_n} (\dot{\hat{\theta}}_2 - g_2(\theta_1, \theta_2, \hat{\theta}_1, \hat{\theta}_2, u_1, u_2))\psi dt \right\} = 0 \end{cases}$$

$\forall \psi(t) \in \mathbb{V}([t_0,\ t_f])$. The arbitrariness of the test functions means that the above problem is equivalent to the following final weak formulation, which requires the terms in the sums above to be zero individually: find $\theta_1(t)$, $\theta_2(t)$, $\hat{\theta}_1(t)$ and $\hat{\theta}_2(t)$ for $t \in [0,\ t_f]$ such that (3.4) are fulfilled and such that:

$$\begin{cases} \int_{t_{n-1}}^{t_n} (\dot{\theta}_1 - \hat{\theta}_1)\psi dt = 0 \\ \int_{t_{n-1}}^{t_n} (\dot{\hat{\theta}}_1 - g_1(\theta_1, \theta_2, \hat{\theta}_1, \hat{\theta}_2, u_1, u_2))\psi dt = 0 \\ \int_{t_{n-1}}^{t_n} (\dot{\theta}_2 - \hat{\theta}_2)\psi dt = 0 \\ \int_{t_{n-1}}^{t_n} (\dot{\hat{\theta}}_2 - g_2(\theta_1, \theta_2, \hat{\theta}_1, \hat{\theta}_2, u_1, u_2))\psi dt = 0 \end{cases} \tag{3.5}$$

$\forall \psi(t) \in \mathbb{V}([t_{n-1},\ t_n])$ and for $n = 1, 2, \ldots, N$.

3.3.2 Finite element formulation

In order to derive the finite element formulation of the problem, the following finite element discretizations are introduced:

$$\theta_1(t) \approx \theta_{1h}(t) = \sum_{i=0}^{N} \theta_{1,i} X_i(t) \tag{3.6}$$

$$\theta_2(t) \approx \theta_{2h}(t) = \sum_{i=0}^{N} \theta_{2,i} X_i(t) \tag{3.7}$$

$$\hat{\theta}_1(t) \approx \hat{\theta}_{1h}(t) = \sum_{i=0}^{N} \hat{\theta}_{1,i} X_i(t) \tag{3.8}$$

$$\hat{\theta}_2(t) \approx \hat{\theta}_{2h}(t) = \sum_{i=0}^{N} \hat{\theta}_{2,i} X_i(t), \tag{3.9}$$

where $\theta_{1,i},\ \theta_{2,i}$, $\hat{\theta}_{1,i}$ and $\hat{\theta}_{2,i}$, $i = 0, 1, 2, \ldots, N$ are the node values of the respective discretizations and $X_i(t)$ are the shape functions. These are piecewise linear, connected to the same time mesh used for the division of the integration domain and satisfy the following.

$$X_i(t) = \begin{cases} 1 & t = t_i \\ 0 & t = t_j,\ j \neq i \end{cases} \tag{3.10}$$

Furthermore, the following type of function is chosen for the test functions:

$$\psi_i(t) = \begin{cases} 1 & t_{i-1} < t \leq t_i \\ 0 & \text{elsewhere.} \end{cases} \tag{3.11}$$

Figure 3.2 shows examples of the shape functions $X_i(t)$ and the test functions $\psi_i(t)$:

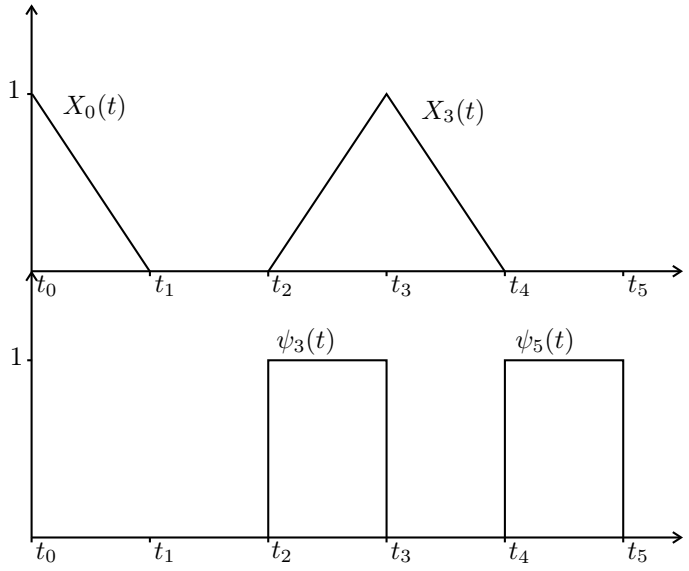

Figure 3.2: Some examples of the shape functions $X_i(t)$ and the test functions $\psi_i(t)$ in an example mesh for which $N = 5$.

Introducing the following set of vectors, containing the node values of the approximations defined above: $\boldsymbol{\theta}_n = (\theta_{1,n}, \theta_{2,n}, \hat{\theta}_{1,n}, \hat{\theta}_{2,n}),\ n = 1, 2, \ldots, N$ and introducing the mentioned choices for the approximations and the test functions into equation (3.5) results in the following problem: Find $\boldsymbol{\theta}_n,\ n = 1, 2, \ldots, N$ such that:

$$\begin{cases} \int_{t_{n-1}}^{t_n} (\dot{\theta}_{1h} - \hat{\theta}_{1h})\psi_n dt = 0 \\ \int_{t_{n-1}}^{t_n} (\dot{\hat{\theta}}_{1h} - g_1(\theta_{1h}, \theta_{2h}, \hat{\theta}_{1h}, \hat{\theta}_{2h}, u_1, u_2))\psi_n dt = 0 \\ \int_{t_{n-1}}^{t_n} (\dot{\theta}_{2h} - \hat{\theta}_{2h})\psi_n dt = 0 \\ \int_{t_{n-1}}^{t_n} (\dot{\hat{\theta}}_{2h} - g_2(\theta_{1h}, \theta_{2h}, \hat{\theta}_{1h}, \hat{\theta}_{2h}, u_1, u_2))\psi_n dt = 0 \end{cases} \tag{3.12}$$

$$\begin{aligned} \theta_{ih}(0) &= \theta_{i0} \\ \dot{\theta}_{ih}(0) &= \dot{\theta}_{i0} \end{aligned} \tag{3.13}$$

for $i = 1, 2$ and $n = 1, 2, \ldots, N$.

Using the definitions (3.6) – (3.9) and the fact that it is only the shape functions $X_{n-1}(t)$ and $X_n(t)$ that do not vanish on the interval $[t_{n-1},\ t_n]$, we can reformulate the above as: Find $\boldsymbol{\theta}_n,\ n = 1, 2, \ldots, N$ such that (3.13) are fulfilled and such that:

$$\begin{cases} \int_{t_{n-1}}^{t_n} \left[\left(\theta_{1,n-1}\dot{X}_{n-1} + \theta_{1,n}\dot{X}_n \right) - \left(\hat{\theta}_{1,n-1}X_{n-1} + \hat{\theta}_{1,n}X_n \right) \right] \psi_n dt = 0 \\ \int_{t_{n-1}}^{t_n} \left[\left(\hat{\theta}_{1,n-1}\dot{X}_{n-1} + \hat{\theta}_{1,n}\dot{X}_n \right) - g_1(\theta_{1h}, \theta_{2h}, \hat{\theta}_{1h}, \hat{\theta}_{2h}, u_1, u_2) \right] \psi_n dt = 0 \\ \int_{t_{n-1}}^{t_n} \left[\left(\theta_{2,n-1}\dot{X}_{n-1} + \theta_{2,n}\dot{X}_n \right) - \left(\hat{\theta}_{2,n-1}X_{n-1} + \hat{\theta}_{2,n}X_n \right) \right] \psi_n dt = 0 \\ \int_{t_{n-1}}^{t_n} \left[\left(\hat{\theta}_{2,n-1}\dot{X}_{n-1} + \hat{\theta}_{2,n}\dot{X}_n \right) - g_2(\theta_{1h}, \theta_{2h}, \hat{\theta}_{1h}, \hat{\theta}_{2h}, u_1, u_2) \right] \psi_n dt = 0 \end{cases} \tag{3.14}$$

$n = 1, 2, \ldots, N$.

These equations enable a sequential solution procedure for finding the node values $\boldsymbol{\theta}_n$: in the equation system corresponding to $n = 1$, the node values $\boldsymbol{\theta}_1$ are solved for using the initial conditions $\boldsymbol{\theta}_0$. Those values are then used in the equation system corresponding to $n = 2$ to find $\boldsymbol{\theta}_2$ and so on.

A set of residual functions are now defined:

$$\boldsymbol{R}_n(\boldsymbol{\theta}_n) = \begin{bmatrix} \int_{t_{n-1}}^{t_n} \left[\left(\theta_{1,n-1}\dot{X}_{n-1} + \theta_{1,n}\dot{X}_n \right) - \left(\hat{\theta}_{1,n-1}X_{n-1} + \hat{\theta}_{1,n}X_n \right) \right] \psi_n dt \\ \int_{t_{n-1}}^{t_n} \left[\left(\hat{\theta}_{1,n-1}\dot{X}_{n-1} + \hat{\theta}_{1,n}\dot{X}_n \right) - g_1(\theta_{1,n}, \theta_{2,n}, \hat{\theta}_{1,n}, \hat{\theta}_{2,n}) \right] \psi_n dt \\ \int_{t_{n-1}}^{t_n} \left[\left(\theta_{2,n-1}\dot{X}_{n-1} + \theta_{2,n}\dot{X}_n \right) - \left(\hat{\theta}_{2,n-1}X_{n-1} + \hat{\theta}_{2,n}X_n \right) \right] \psi_n dt \\ \int_{t_{n-1}}^{t_n} \left[\left(\hat{\theta}_{2,n-1}\dot{X}_{n-1} + \hat{\theta}_{2,n}\dot{X}_n \right) - g_2(\theta_{1,n}, \theta_{2,n}, \hat{\theta}_{1,n}, \hat{\theta}_{2,n}) \right] \psi_n dt \end{bmatrix} \tag{3.15}$$

$n = 1, 2, \ldots, N$. It was here emphasized, in recognition of the fact that the above equations will be solved with only $\boldsymbol{\theta}_n$ as unknowns, that the functions g_1 and g_2 depend on these node values, which they do via the respective approximations.

The discretized problem can then be written as: Find $\boldsymbol{\theta}_n,\ n = 1, 2, \ldots, N$ such that (3.4) are fulfilled and such that:

$$\boldsymbol{R}_n(\boldsymbol{\theta}_n) = \mathbf{0},\ n = 1, 2, \ldots, N. \tag{3.16}$$

As mentioned above, the above set of equations, together with the initial conditions in (3.4) – which provide $\boldsymbol{\theta}_0$ – enable a sequential procedure for successively solving for $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots, \boldsymbol{\theta}_N$. Note that the terminal conditions are not used in this procedure. They are instead used as constraints in the optimization process, just as for the `ode45` method (Section 3.2). This means that the optimization routine based on the temporal finite element method also employs a type of shooting method to satisfy the terminal conditions.

3.3.3 Solving the equations in a given time increment

The Newton method is suitable for solving the equations (3.16) in each time interval. The Newton iteration scheme for time step n is the following:

$$\begin{aligned} \boldsymbol{R}_n(\boldsymbol{\theta}_n^{(k)}) + \mathcal{D}\boldsymbol{R}_n(\boldsymbol{\theta}_n^{(k)})[\Delta\boldsymbol{\theta}] &= \mathbf{0}, \\ \boldsymbol{\theta}_n^{(k+1)} &= \boldsymbol{\theta}_n^{(k)} + \Delta\boldsymbol{\theta}, \end{aligned} \tag{3.17}$$

where $\mathcal{D}\boldsymbol{R}_n(\boldsymbol{\theta}_n^{(k)})[\Delta\boldsymbol{\theta}]$ is the directional derivative of $\boldsymbol{R}_n$ at $\boldsymbol{\theta}_n^{(k)}$ in the direction of the increment $\Delta\boldsymbol{\theta} = [\Delta\theta_1\ \Delta\theta_2\ \Delta\hat{\theta}_1\ \Delta\hat{\theta}_2]$. The definition is: $\mathcal{D}\boldsymbol{R}_n(\boldsymbol{\theta}_n^{(k)})[\Delta\boldsymbol{\theta}] = \frac{d}{d\epsilon}\left[\boldsymbol{R}_n(\boldsymbol{\theta}_n^{(k)} + \epsilon\Delta\boldsymbol{\theta})\right]_{\epsilon=0}$, which in our case becomes (dropping the superscript "(k)" for brevity):

$$\mathcal{D}\boldsymbol{R}_n(\boldsymbol{\theta}_n)[\Delta\boldsymbol{\theta}] =$$

$$\left[\begin{array}{l} \frac{d}{d\epsilon}\int_{t_{n-1}}^{t_n}\Big[\Big(\theta_{1,n-1}\dot{X}_{n-1} + (\theta_{1,n} + \epsilon\Delta\theta_1)\dot{X}_n\Big) - \\ \Big(\hat{\theta}_{1,n-1}X_{n-1} + (\hat{\theta}_{1,n} + \epsilon\Delta\hat{\theta}_1)X_n\Big)\Big]\psi_n dt \\ \frac{d}{d\epsilon}\int_{t_{n-1}}^{t_n}\Big[\Big(\hat{\theta}_{1,n-1}\dot{X}_{n-1} + (\hat{\theta}_{1,n} + \epsilon\Delta\hat{\theta}_1)\dot{X}_n\Big) - \\ g_1(\theta_{1h}(\theta_{1,n} + \epsilon\Delta\theta_1), \theta_{2h}(\theta_{2,n} + \epsilon\Delta\theta_2), \hat{\theta}_{1h}(\hat{\theta}_{1,n} + \epsilon\Delta\hat{\theta}_1), \hat{\theta}_{2h}(\hat{\theta}_{2,n} + \epsilon\Delta\hat{\theta}_2), u_1, u_2)\Big]\psi_n dt \\ \frac{d}{d\epsilon}\int_{t_{n-1}}^{t_n}\Big[\Big(\theta_{2,n-1}\dot{X}_{n-1} + (\theta_{2,n} + \epsilon\Delta\theta_2)\dot{X}_n\Big) - \\ \Big(\hat{\theta}_{2,n-1}X_{n-1} + (\hat{\theta}_{2,n} + \epsilon\Delta\hat{\theta}_2)X_n\Big)\Big]\psi_n dt \\ \frac{d}{d\epsilon}\int_{t_{n-1}}^{t_n}\Big[\Big(\hat{\theta}_{2,n-1}\dot{X}_{n-1} + (\hat{\theta}_{2,n} + \epsilon\Delta\hat{\theta}_2)\dot{X}_n\Big) - \\ g_2(\theta_{1h}(\theta_{1,n} + \epsilon\Delta\theta_1), \theta_{2h}(\theta_{2,n} + \epsilon\Delta\theta_2), \hat{\theta}_{1h}(\hat{\theta}_{1,n} + \epsilon\Delta\hat{\theta}_1), \hat{\theta}_{2h}(\hat{\theta}_{2,n} + \epsilon\Delta\hat{\theta}_2), u_1, u_2)\Big]\psi_n dt \end{array}\right]_{\epsilon=0} =$$

$$= \left[\begin{array}{l} \int_{t_{n-1}}^{t_n}\Big[\dot{X}_n\Delta\theta_1 - X_n\Delta\hat{\theta}_1\Big]\psi_n dt \\ \int_{t_{n-1}}^{t_n}\Big[\dot{X}_n\Delta\hat{\theta}_1 - (g_{1\theta_1}'X_n\Delta\theta_1 + g_{1\theta_2}'X_n\Delta\theta_2 + g_{1\hat{\theta}_1}'X_n\Delta\hat{\theta}_1 + g_{1\hat{\theta}_2}'X_n\Delta\hat{\theta}_2)\Big]\psi_n dt \\ \int_{t_{n-1}}^{t_n}\Big[\dot{X}_n\Delta\theta_2 - X_n\Delta\hat{\theta}_2\Big]\psi_n dt \\ \int_{t_{n-1}}^{t_n}\Big[\dot{X}_n\Delta\hat{\theta}_2 - (g_{2\theta_1}'X_n\Delta\theta_1 + g_{2\theta_2}'X_n\Delta\theta_2 + g_{2\hat{\theta}_1}'X_n\Delta\hat{\theta}_1 + g_{2\hat{\theta}_2}'X_n\Delta\hat{\theta}_2)\Big]\psi_n dt \end{array}\right] =$$

$$= \boldsymbol{J}_n(\boldsymbol{\theta}_n^{(k)})\Delta\boldsymbol{\theta} \tag{3.18}$$

where

$$\boldsymbol{J}_n = \int_{t_{n-1}}^{t_n}\left[\begin{array}{cccc} \dot{X}_n & 0 & -X_n & 0 \\ -g_{1\theta_1}'X_n & -g_{1\theta_2}'X_n & (\dot{X}_n - g_{1\hat{\theta}_1}'X_n) & -g_{1\hat{\theta}_2}'X_n \\ 0 & \dot{X}_n & 0 & -X_n \\ -g_{2\theta_1}'X_n & -g_{2\theta_2}'X_n & -g_{2\hat{\theta}_1}'X_n & (\dot{X}_n - g_{2\hat{\theta}_2}'X_n) \end{array}\right]\psi_n dt \tag{3.19}$$

3.3.4 The gradient of the objective function E

The value of the gradient of the objective function with respect to the controls at a specific point can be calculated and returned to `fmincon` after the evaluation of E. Relying in this way on algebraic expressions rather than numerical calculations (as mentioned, `fmincon` by default uses finite differences to calculate the gradient) to evaluate the gradient is likely to greatly increase the speed of the optimization process and is, as mentioned, one of the most interesting features of the temporal finite element method.

We introduce discretizations of the control variables of the form shown in equation (3.1):

$$\begin{aligned} u_1(t) \approx u_{1h}(t) = \sum_{i=0}^{M} u_{1,i} Y_i(t) \\ u_2(t) \approx u_{2h}(t) = \sum_{i=0}^{M} u_{2,i} Y_i(t) \end{aligned} \tag{3.20}$$

Using these discretizations of the control variables as well as equations (3.8) and (3.9) (but with $\hat{\theta}$ replaced by $\dot{\theta}$) in equation (2.23) gives the following discretization of the two components of the objective function (for brevity, "(t)" is dropped in the following equations):

$$\begin{aligned} E_1 \approx E_{1h} = \frac{1}{L} \int_0^{t_f} \left[\mid u_{1h}\dot{\theta}_{1h} \mid + \mid u_{2h}(\dot{\theta}_{1h} - \dot{\theta}_{2h}) \mid \right] dt \\ E_2 \approx E_{2h} = \int_0^{t_f} \left[u_{1h}^2 + u_{2h}^2 \right] dt \end{aligned} \tag{3.21}$$

The gradient of the discretized objective function $E_h = \lambda_1 E_{1h} + \lambda_2 E_{2h}$ with respect to the node values of the approximations of the control variables: $\{u_{1,j},\ u_{2,j}\}$, $j = 1, 2, \ldots, M$, is then:

$$\begin{aligned} \nabla E_h = \left(\frac{\mathrm{d}E_h}{\mathrm{d}u_{1,1}}, \frac{\mathrm{d}E_h}{\mathrm{d}u_{1,2}}, \ldots, \frac{\mathrm{d}E_h}{\mathrm{d}u_{1,M}} \right) = \\ \left(\lambda_1 \frac{\mathrm{d}E_{1h}}{\mathrm{d}u_{1,1}} + \lambda_2 \frac{\mathrm{d}E_{2h}}{\mathrm{d}u_{1,1}}, \lambda_1 \frac{\mathrm{d}E_{1h}}{\mathrm{d}u_{1,2}} + \lambda_2 \frac{\mathrm{d}E_{2h}}{\mathrm{d}u_{1,2}}, \ldots, \lambda_1 \frac{\mathrm{d}E_{1h}}{\mathrm{d}u_{1,M}} + \lambda_2 \frac{\mathrm{d}E_{2h}}{\mathrm{d}u_{1,M}} \right) \end{aligned} \tag{3.22}$$

The derivative of E_{1h} with respect to the node values $u_{1,i}$ and $u_{2,i}$ are:

$$\begin{aligned} \frac{\mathrm{d}E_{1h}}{\mathrm{d}u_{1,i}} = \frac{1}{L} \int_0^{t_f} \left[\mathrm{sgn}\left(u_{1h}\dot{\theta}_{1h} \right) \left(Y_i \dot{\theta}_{1h} + u_{1h} \frac{\mathrm{d}\dot{\theta}_{1h}}{\mathrm{d}u_{1,i}} \right) \right. \\ \left. + \mathrm{sgn}\left(u_{2h}\left(\dot{\theta}_{1h} - \dot{\theta}_{2h} \right) \right) u_{2h} \left(\frac{\mathrm{d}\dot{\theta}_{1h}}{\mathrm{d}u_{1,i}} - \frac{\mathrm{d}\dot{\theta}_{2h}}{\mathrm{d}u_{1,i}} \right) \right] dt \end{aligned} \tag{3.23}$$

$$\begin{aligned} \frac{\mathrm{d}E_{1h}}{\mathrm{d}u_{2,i}} = \frac{1}{L} \int_0^{t_f} \left[\mathrm{sgn}\left(u_{1h}\dot{\theta}_{1h} \right) u_{1h} \frac{\mathrm{d}\dot{\theta}_{1h}}{\mathrm{d}u_{2,i}} \right. \\ \left. + \mathrm{sgn}\left(u_{2h}\left(\dot{\theta}_{1h} - \dot{\theta}_{2h} \right) \right) \left(Y_i \left(\dot{\theta}_{1h} - \dot{\theta}_{2h} \right) + u_{2h} \left(\frac{\mathrm{d}\dot{\theta}_{1h}}{\mathrm{d}u_{2,i}} - \frac{\mathrm{d}\dot{\theta}_{2h}}{\mathrm{d}u_{2,i}} \right) \right) \right] dt. \end{aligned}$$

Note that the generalized coordinates $\theta_1(t)$ and $\theta_2(t)$ are functions of the control variables $u_1(t)$ and $u_2(t)$ and thus, in the discretized problem, $\theta_{1h}(t)$ and $\theta_{2h}(t)$ are functions of the node values of the approximations of the control variables: $\{u_{1,j},\ u_{2,j}\}$, $j = 1, 2, \ldots, M$. The calculation of the derivatives $\frac{\mathrm{d}\dot{\theta}_{jh}}{\mathrm{d}u_{k,i}}$ is described below. The derivatives of E_{2h} with respect to the node values are:

$$\begin{aligned} \frac{\mathrm{d}E_{2h}}{\mathrm{d}u_{1,i}} = \int_0^{t_f} 2u_{1h} Y_i dt \\ \frac{\mathrm{d}E_{2h}}{\mathrm{d}u_{2,i}} = \int_0^{t_f} 2u_{2h} Y_i dt. \end{aligned} \tag{3.24}$$

3.3.5 Gradients of inequality constraints

The inequality constraints, given in Table 2.2, are collected in a vector as follows:

$$\boldsymbol{c}_{ineq} = \begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} \theta_2 - \theta_1 \\ -\theta_1 - 10^\circ \\ -y_{hip}(t_f) + a_1 \cos\theta_1 + a_2 \cos\theta_2 \end{bmatrix} \leq \mathbf{0} \tag{3.25}$$

Introducing the discretized inequality constraint vector, $\boldsymbol{c}_{ineq,h} = (c_{1h}, c_{2h}, c_{3h})$, defined as the one resulting from replacing $\theta_1(t)$ and $\theta_2(t)$ in $\boldsymbol{c}_{ineq}$ by their respective discretizations, we can calculate the gradient with respect to the node values of the discretized control moments:

$$\nabla \boldsymbol{c}_{ineq,h} = \begin{bmatrix} \frac{dc_{1h}}{du_{1,1}} & \frac{dc_{1h}}{du_{1,2}} & \cdots & \frac{dc_{1h}}{du_{1,M}} & \frac{dc_{1h}}{du_{2,1}} & \cdots & \frac{dc_{1h}}{du_{2,M}} \\ \frac{dc_{2h}}{du_{1,1}} & \frac{dc_{2h}}{du_{1,2}} & \cdots & \frac{dc_{2h}}{du_{1,M}} & \frac{dc_{2h}}{du_{2,1}} & \cdots & \frac{dc_{2h}}{du_{2,M}} \\ \frac{dc_{3h}}{du_{1,1}} & \frac{dc_{3h}}{du_{1,2}} & \cdots & \frac{dc_{3h}}{du_{1,M}} & \frac{dc_{3h}}{du_{2,1}} & \cdots & \frac{dc_{3h}}{du_{2,M}} \end{bmatrix} \tag{3.26}$$

The derivative $\frac{dc_{ih}}{du_{k,j}}$ is investigated:

$$\frac{dc_{ih}}{du_{k,j}} = \frac{\partial c_{ih}}{\partial u_{k,j}} + \frac{\partial c_{ih}}{\partial \theta_{1h}} \frac{\mathrm{d}\theta_{1h}}{\mathrm{d}u_{k,j}} + \frac{\partial c_{ih}}{\partial \theta_{2h}} \frac{\mathrm{d}\theta_{2h}}{\mathrm{d}u_{k,j}} + \frac{\partial c_{ih}}{\partial \hat{\theta}_{1h}} \frac{\mathrm{d}\hat{\theta}_{1h}}{\mathrm{d}u_{k,j}} + \frac{\partial c_{ih}}{\partial \hat{\theta}_{2h}} \frac{\mathrm{d}\hat{\theta}_{2h}}{\mathrm{d}u_{k,j}} \tag{3.27}$$

The derivatives $\frac{\partial c_{ih}}{\partial u_{k,j}}$, $\frac{\partial c_{ih}}{\partial \hat{\theta}_{1h}}$ and $\frac{\partial c_{ih}}{\partial \hat{\theta}_{2h}}$ are all zero and the derivatives $\frac{\partial c_i}{\partial \theta_{1h}}$ and $\frac{\partial c_i}{\partial \theta_{2h}}$ are given in equations (3.28) thru (3.33).

$$\frac{\partial c_{1h}}{\partial \theta_{1h}} = -1 \tag{3.28}$$

$$\frac{\partial c_{1h}}{\partial \theta_{2h}} = 1 \tag{3.29}$$

$$\frac{\partial c_{2h}}{\partial \theta_{1h}} = -1 \tag{3.30}$$

$$\frac{\partial c_{2h}}{\partial \theta_{2h}} = 0 \tag{3.31}$$

$$\frac{\partial c_{3h}}{\partial \theta_{1h}} = -a_1 \sin\theta_{1h} \tag{3.32}$$

$$\frac{\partial c_{3h}}{\partial \theta_{2h}} = -a_2 \sin\theta_{2h} \tag{3.33}$$

In order to produce the derivatives $\frac{\mathrm{d}\theta_{1h}}{\mathrm{d}u_{k,j}}$ and $\frac{\mathrm{d}\theta_{2h}}{\mathrm{d}u_{k,j}}$, the directional derivative $\mathcal{D}\bullet(u_k(t))[\delta u(t)]$ of both sides in each of the equations in (2.21) is first taken. The idea is that the sought derivatives will appear when the increment function $\delta u(t)$ is chosen as the basis function $Y_j(t)$, since the effect of increasing the node value $u_{k,j}$ for the discretized control variable u_{kh} by some value a is exactly that achieved by adding the function $aY_j(t)$.

The derivation for the derivatives $\frac{\mathrm{d}\theta_{1h}}{\mathrm{d}u_{1,j}}$ and $\frac{\mathrm{d}\theta_{2h}}{\mathrm{d}u_{1,j}}$ are chosen for the following demonstration ("(t)" is dropped in the following). We first introduce the definitions:

$$\begin{cases} \theta_{1,u_1} \equiv \mathcal{D}\theta_1(u_1)[\delta u] \equiv \left.\dfrac{\mathrm{d}}{\mathrm{d}\epsilon}\right|_{\epsilon=0} \theta_1(u_1 + \epsilon\delta u, u_2) = \dfrac{\partial \theta_1}{\partial u_1}\delta u \\ \hat{\theta}_{1,u_1} \equiv \mathcal{D}\hat{\theta}_1(u_1)[\delta u] \equiv \left.\dfrac{\mathrm{d}}{\mathrm{d}\epsilon}\right|_{\epsilon=0} \hat{\theta}_1(u_1 + \epsilon\delta u, u_2) = \dfrac{\partial \hat{\theta}_1}{\partial u_1}\delta u \\ \theta_{2,u_1} \equiv \mathcal{D}\theta_2(u_1)[\delta u] \equiv \left.\dfrac{\mathrm{d}}{\mathrm{d}\epsilon}\right|_{\epsilon=0} \theta_2(u_1 + \epsilon\delta u, u_2) = \dfrac{\partial \theta_2}{\partial u_1}\delta u \\ \hat{\theta}_{2,u_1} \equiv \mathcal{D}\hat{\theta}_2(u_1)[\delta u] \equiv \left.\dfrac{\mathrm{d}}{\mathrm{d}\epsilon}\right|_{\epsilon=0} \hat{\theta}_2(u_1 + \epsilon\delta u, u_2) = \dfrac{\partial \hat{\theta}_2}{\partial u_1}\delta u \end{cases} \tag{3.34}$$

Furthermore, we see that:

$$\begin{aligned} \mathcal{D}g_1(u_1)[\delta u] &= \left.\frac{\mathrm{d}}{\mathrm{d}\epsilon}\right|_{\epsilon=0} \Big[g_1\Big(\theta_1(u_1+\epsilon\delta u, u_2), \theta_2(u_1+\epsilon\delta u, u_2), \hat{\theta}_1(u_1+\epsilon\delta u, u_2), \\ &\qquad \hat{\theta}_2(u_1+\epsilon\delta u, u_2), u_1+\epsilon\delta u, u_2\Big)\Big] \\ &= \frac{\partial g_1}{\partial \theta_1}\frac{\partial \theta_1}{\partial u_1}\delta u + \frac{\partial g_1}{\partial \theta_2}\frac{\partial \theta_2}{\partial u_1}\delta u + \frac{\partial g_1}{\partial \hat{\theta}_1}\frac{\partial \hat{\theta}_1}{\partial u_1}\delta u + \frac{\partial g_1}{\partial \hat{\theta}_2}\frac{\partial \hat{\theta}_2}{\partial u_1}\delta u + \frac{\partial g_1}{\partial u_1}\delta u \\ &= \frac{\partial g_1}{\partial \theta_1}\theta_{1,u_1} + \frac{\partial g_1}{\partial \theta_2}\theta_{2,u_1} + \frac{\partial g_1}{\partial \hat{\theta}_1}\hat{\theta}_{1,u_1} + \frac{\partial g_1}{\partial \hat{\theta}_2}\hat{\theta}_{2,u_1} + \frac{\partial g_1}{\partial u_1}\delta u \end{aligned} \tag{3.35}$$

and, similarly:

$$\mathcal{D}g_2(u_1)\left[\delta u\right] = \frac{\partial g_2}{\partial \theta_1}\theta_{1,u_1} + \frac{\partial g_2}{\partial \theta_2}\theta_{2,u_1} + \frac{\partial g_2}{\partial \hat{\theta}_1}\hat{\theta}_{1,u_1} + \frac{\partial g_2}{\partial \hat{\theta}_2}\hat{\theta}_{2,u_1} + \frac{\partial g_2}{\partial u_1}\delta u. \tag{3.36}$$

Recognizing that the directional derivative and the time derivative are interchangeable, it is seen that taking the directional derivative of both sides in each of the four equations in (2.21) yields the linear equation

$$\begin{cases} \dot{\theta}_{1,u_1} = \hat{\theta}_{1,u_1} \\ \dot{\hat{\theta}}_{1,u_1} = g_1{}'_{\theta_1}\theta_{1,u_1} + g_1{}'_{\theta_2}\theta_{2,u_1} + g_1{}'_{\hat{\theta}_1}\hat{\theta}_{1,u_1} + g_1{}'_{\hat{\theta}_2}\hat{\theta}_{2,u_1} + g_1{}'_{u_1}\delta u \\ \dot{\theta}_{2,u_1} = \hat{\theta}_{2,u_1} \\ \dot{\hat{\theta}}_{2,u_1} = g_2{}'_{\theta_1}\theta_{1,u_1} + g_2{}'_{\theta_2}\theta_{2,u_1} + g_2{}'_{\hat{\theta}_1}\hat{\theta}_{1,u_1} + g_2{}'_{\hat{\theta}_2}\hat{\theta}_{2,u_1} + g_2{}'_{u_1}\delta u \end{cases} \tag{3.37}$$

The cG(1) temporal finite element formulation of the given problem is now derived in the same manner as above: multiplying the equations by test functions, integrating over the whole temporal interval, splitting the integrals into sums of integrals over subintervals, introducing discretizations analogous to the ones in equations (3.6) – (3.9):

$$\theta_{1,u_1}(t) \approx \theta_{1,u_1,h}(t) = \sum_{i=0}^{N} \theta_{1,u_1,i} X_i(t) \tag{3.38}$$

$$\theta_{2,u_1}(t) \approx \theta_{2,u_1,h}(t) = \sum_{i=0}^{N} \theta_{2,u_1,i} X_i(t) \tag{3.39}$$

$$\hat{\theta}_{1,u_1}(t) \approx \hat{\theta}_{1,u_1,h}(t) = \sum_{i=0}^{N} \hat{\theta}_{1,u_1,i} X_i(t) \tag{3.40}$$

$$\hat{\theta}_{2,u_1}(t) \approx \hat{\theta}_{2,u_1,h}(t) = \sum_{i=0}^{N} \hat{\theta}_{2,u_1,i} X_i(t) \tag{3.41}$$

and removing vanishing terms in the sums. The result is the following problem (cf. equation (3.14)): Find the node values $\theta_{1,u_1,n}$, $\theta_{2,u_1,n}$, $\hat{\theta}_{1,u_1,n}$, $\hat{\theta}_{1,u_1,n}$ such that the initial conditions described below are fulfilled and such that:

$$\begin{cases} \int_{t_{n-1}}^{t_n} \left[(\theta_{1,u_1,n-1}\dot{X}_{n-1} + \theta_{1,u_1,n}\dot{X}_n) - (\hat{\theta}_{1,u_1,n-1}X_{n-1} + \hat{\theta}_{1,u_1,n}X_n) \right] \psi_n dt = 0 \\ \int_{t_{n-1}}^{t_n} \Big[-(\hat{\theta}_{1,u_1,n-1}\dot{X}_{n-1} + \hat{\theta}_{1,u_1,n}\dot{X}_n) + g_1{}'_{\theta_1}(\theta_{1,u_1,n-1}X_{n-1} + \theta_{1,u_1,n}X_n) \\ \quad + g_1{}'_{\theta_2}(\theta_{2,u_1,n-1}X_{n-1} + \theta_{2,u_1,n}X_n) + g_1{}'_{\hat{\theta}_1}(\hat{\theta}_{1,u_1,n-1}X_{n-1} + \hat{\theta}_{1,u_1,n}X_n) \\ \quad + g_1{}'_{\hat{\theta}_2}(\hat{\theta}_{2,u_1,n-1}X_{n-1} + \hat{\theta}_{2,u_1,n}X_n) + g'_{1,u_1}\delta u_h \Big] \psi_n dt = 0 \\ \int_{t_{n-1}}^{t_n} \left[(\theta_{2,u_1,n-1}\dot{X}_{n-1} + \theta_{2,u_1,n}\dot{X}_n) - (\hat{\theta}_{2,u_1,n-1}X_{n-1} + \hat{\theta}_{2,u_1,n}X_n) \right] \psi_n dt = 0 \\ \int_{t_{n-1}}^{t_n} \Big[-(\hat{\theta}_{2,u_1,n-1}\dot{X}_{n-1} + \hat{\theta}_{2,u_1,n}\dot{X}_n) + g_2{}'_{\theta_1}(\theta_{1,u_1,n-1}X_{n-1} + \theta_{1,u_1,n}X_n) \\ \quad + g_2{}'_{\theta_2}(\theta_{2,u_1,n-1}X_{n-1} + \theta_{2,u_1,n}X_n) + g_2{}'_{\hat{\theta}_1}(\hat{\theta}_{1,u_1,n-1}X_{n-1} + \hat{\theta}_{1,u_1,n}X_n) \\ \quad + g_2{}'_{\hat{\theta}_2}(\hat{\theta}_{2,u_1,n-1}X_{n-1} + \hat{\theta}_{2,u_1,n}X_n) + g'_{2,u_1}\delta u_h \Big] \psi_n dt = 0, \end{cases} \tag{3.42}$$

$n = 1, 2, \ldots, N$. As was the case for the equations (3.14), these equations enable a sequential solution procedure in which the unknowns in each time interval $[t_{n-1},\ t_n]$ are $\theta_{1,u_1,n}$, $\theta_{2,u_1,n}$, $\hat{\theta}_{1,u_1,n}$ and $\hat{\theta}_{2,u_1,n}$. The initial conditions used for this purpose are: $\theta_{1,u_1,0} = 0$, $\theta_{2,u_1,0} = 0$, $\hat{\theta}_{1,u_1,0} = 0$ and $\hat{\theta}_{2,u_1,0} = 0$, since we need to have $\theta_{1,u_1}(0) = 0$, $\theta_{2,u_1}(0) = 0$, $\hat{\theta}_{1,u_1}(0) = 0$ and $\hat{\theta}_{2,u_1}(0) = 0$. The reason is that applied forces directly affect only the accelerations in a system, and thus can not change the state of the system instantaneously, so positions and velocities at $t = 0$ can not be affected for any change in the time history of the control.

Rearranging the equations so that all known quantities are on the right hand side yields right and left hand sides as shown in equations (3.43) and (3.44), respectively:

$$RHS = \begin{cases} \int_{t_{n-1}}^{t_n} \left[-\theta_{1,u_1,n-1}\dot{X}_{n-1} + \hat{\theta}_{1,u_1,n-1}X_{n-1} \right] \psi_n dt \\ \\ \int_{t_{n-1}}^{t_n} \Big[-\hat{\theta}_{1,u_1,n-1}\dot{X}_{n-1} + g1'_{\theta_1}\theta_{1,u_1,n-1}X_{n-1} \\ +g1'_{\theta_2}\theta_{2,u_1,n-1}X_{n-1} + g1'_{\hat{\theta}_1}\hat{\theta}_{1,u_1,n-1}X_{n-1} \\ +g1'_{\hat{\theta}_2}\hat{\theta}_{2,u_1,n-1}X_{n-1} + g'_{1,u_1}\delta u_h \Big] \psi_n dt \\ \\ \int_{t_{n-1}}^{t_n} \left[-\theta_{2,u_1,n-1}\dot{X}_{n-1} + \hat{\theta}_{2,u_1,n-1}X_{n-1} \right] \psi_n dt \\ \\ \int_{t_{n-1}}^{t_n} \Big[-\hat{\theta}_{2,u_1,n-1}\dot{X}_{n-1} + g2'_{\theta_1}\theta_{1,u_1,n-1}X_{n-1} \\ +g2'_{\theta_2}\theta_{2,u_1,n-1}X_{n-1} + g2'_{\hat{\theta}_1}\hat{\theta}_{1,u_1,n-1}X_{n-1} \\ +g2'_{\hat{\theta}_2}\hat{\theta}_{2,u_1,n-1}X_{n-1} + g'_{2,u_1}\delta u_h \Big] \psi_n dt \end{cases} \tag{3.43}$$

$$LHS = \begin{cases} \int_{t_{n-1}}^{t_n} \left[(\theta_{1,u_1,n}\dot{X}_n - \hat{\theta}_{1,u_1,n}X_n \right] \psi_n dt \\ \\ \int_{t_{n-1}}^{t_n} \Big[\hat{\theta}_{1,u_1,n}\dot{X}_n - g1'_{\theta_1}\theta_{1,u_1,n}X_n \\ -g1'_{\theta_2}\theta_{2,u_1,n}X_n - g1'_{\hat{\theta}_1}\hat{\theta}_{1,u_1,n}X_n \\ -g1'_{\hat{\theta}_2}\hat{\theta}_{2,u_1,n}X_n \Big] \psi_n dt \\ \\ \int_{t_{n-1}}^{t_n} \left[(\theta_{2,u_1,n}\dot{X}_n - \hat{\theta}_{2,u_1,n}X_n \right] \psi_n dt \\ \\ \int_{t_{n-1}}^{t_n} \Big[\hat{\theta}_{1,u_1,n}\dot{X}_n - g2'_{\theta_1}\theta_{1,u_1,n}X_n \\ -g2'_{\theta_2}\theta_{2,u_1,n}X_n - g2'_{\hat{\theta}_1}\hat{\theta}_{1,u_1,n}X_n \\ -g2'_{\hat{\theta}_2}\hat{\theta}_{2,u_1,n}X_n \Big] \psi_n dt \end{cases} \tag{3.44}$$

Since $\theta_{1,u_1,n}$, $\theta_{2,u_1,n}$, $\hat{\theta}_{1,u_1,n}$ and $\hat{\theta}_{2,u_1,n}$ are just node values and thus independent of time, the left hand side can be written as follows:

$$\int_{t_{n-1}}^{t_n} \begin{bmatrix} \dot{X}_n & 0 & -X_n & 0 \\ -g1'_{\theta_1}X_n & -g1'_{\theta_2}X_n & (\dot{X}_n - g1'_{\hat{\theta}_1}X_n) & -g1'_{\hat{\theta}_2}X_n \\ 0 & \dot{X}_n & 0 & -X_n \\ -g2'_{\theta_1}X_n & -g2'_{\theta_2}X_n & -g2'_{\hat{\theta}_1}X_n & (\dot{X}_n - g2'_{\hat{\theta}_2}X_n) \end{bmatrix} \psi_n dt \begin{bmatrix} \theta_{1,u_1,n} \\ \theta_{2,u_1,n} \\ \hat{\theta}_{1,u_1,n} \\ \hat{\theta}_{2,u_1,n} \end{bmatrix} \tag{3.45}$$

where it is noted that the first matrix is the same as the Jacobian found above: equation (3.19). This is another convenient feature of the temporal finite element method.

As mentioned above, choosing $\delta u(t) = Y_j(t)$ will make the sought derivatives $\frac{\mathrm{d}\theta_{1h}}{\mathrm{d}u_{1,j}}$ and $\frac{\mathrm{d}\theta_{2h}}{\mathrm{d}u_{1,j}}$ appear. In fact, if $\delta u(t) = Y_j(t)$, then $\theta_{1,u_1,h}(t) \approx \theta_{1,u_1}(t) = \frac{\mathrm{d}}{\mathrm{d}\epsilon}\big|_{\epsilon=0} \theta_1(t)\,(u_1(t) + \epsilon Y_j(t), u_2(t)) = \frac{\mathrm{d}\theta_{1h}}{\mathrm{d}u_{1,j}}(t)$ and $\theta_{2,u_1,h}(t) = \frac{\mathrm{d}}{\mathrm{d}\epsilon}\big|_{\epsilon=0} \theta_2(t)\,(u_1(t) + \epsilon Y_j(t), u_2(t)) = \frac{\mathrm{d}\theta_{2h}}{\mathrm{d}u_{1,j}}(t)$. The last equalities are understood by remembering the above discussion about the meaning of incrementing the control variables by a constant times one of the shape functions. Solving the system of equations incrementally for all time intervals then gives the node values required to evaluate the sought functions using equations (3.38) and (3.39).

3.3.6 Gradients of equality constraints

The four initial conditions (see Table 2.1) are used in the equation system corresponding to the first time interval in (3.16). The two remaining terminal conditions also need to be imposed and this is done in the form of equality constraints in the optimization problem:

$$\boldsymbol{c}_{eq} = \begin{bmatrix} c_4 \\ c_5 \end{bmatrix} = \begin{bmatrix} \theta_1(t_f) - \theta_{1f} \\ \theta_2(t_f) - \theta_{2f} \end{bmatrix} = \mathbf{0} \tag{3.46}$$

Like for the inequality constraint vector, we define the discretized equality constraint vector $\boldsymbol{c}_{eq,h} = (c_{4h}, c_{5h})$ as the vector resulting from replacing $\theta_1(t)$ and $\theta_2(t)$ by $\theta_{1h}(t)$ and $\theta_{2h}(t)$, respectively.

The gradient of the discretized equality constraint vector with respect to the node values of the discretized control moments is then:

$$\nabla \boldsymbol{c}_{eq,h} = \begin{bmatrix} \frac{dc_{4h}}{du_{1,1}} & \frac{dc_{4h}}{du_{1,2}} & \cdots & \frac{dc_{4h}}{du_{1,M}} & \frac{dc_{4h}}{du_{2,1}} & \cdots & \frac{dc_{4h}}{du_{2,M}} \\ \frac{dc_{5h}}{du_{1,1}} & \frac{dc_{5h}}{du_{1,2}} & \cdots & \frac{dc_{5h}}{du_{1,M}} & \frac{dc_{5h}}{du_{2,1}} & \cdots & \frac{dc_{5h}}{du_{2,M}} \end{bmatrix} \tag{3.47}$$

The derivatives of c_{4h} and c_{5h} with respect to the node values are:

$$\begin{aligned} \frac{\mathrm{d}c_{4h}}{\mathrm{d}u_{1,j}} &= \left.\frac{\mathrm{d}\theta_{1h}}{\mathrm{d}u_{1,j}}\right|_{t=t_f} \\ \frac{\mathrm{d}c_{4h}}{\mathrm{d}u_{2,j}} &= \left.\frac{\mathrm{d}\theta_{1h}}{\mathrm{d}u_{2,j}}\right|_{t=t_f} \\ \frac{\mathrm{d}c_{5h}}{\mathrm{d}u_{1,j}} &= \left.\frac{\mathrm{d}\theta_{2h}}{\mathrm{d}u_{1,j}}\right|_{t=t_f} \\ \frac{\mathrm{d}c_{5h}}{\mathrm{d}u_{2,j}} &= \left.\frac{\mathrm{d}\theta_{2h}}{\mathrm{d}u_{2,j}}\right|_{t=t_f} \end{aligned} \tag{3.48}$$

where the derivatives are computed as described above.

3.4 The Fourier method

The Fourier method for optimal control is based on *inverse dynamics*, meaning that in each iteration of the optimizer, the equations of motion are solved for the control given expressions for the state variables. For a system of the given type, which has the control variables appearing as linear terms in the equations of motion, the control is explicitly expressible in terms of the state variables, which means that it can be evaluated exactly and without need for numerical solution methods. Another interesting feature, which will be shown below, of the Fourier method for optimal control is the possibility to incorporate the boundary conditions into the solver so that they are satisfied automatically and accurately.

The idea central to the Fourier series method is the discretization of the considered system of equations by approximation of the time histories of the state variables by the sum of a polynomial and a truncated Fourier series expression. The unknowns of the optimization problem are then the coefficients of this approximation. The optimization process will involve solving the equations of motion for the control, which is what characterizes an inverse dynamics approach. The contents of this section will be largely based on [13].

Just as with the temporal finite element-approach, the Fourier series method will be applied to the first order version of the equations of motion: equation (2.21). The possible boundary conditions for coordinate i are (from equation (3.4)):

$$\begin{aligned} \theta_i(0) &= \theta_{i0}, \quad \theta_i(t_f) = \theta_{if} \\ \dot{\theta}_i(0) &= \dot{\theta}_{i0}, \quad \dot{\theta}_i(t_f) = \dot{\theta}_{if}. \end{aligned} \tag{3.49}$$

In recognition of the inverse dynamics-nature of the Fourier series method, the equations of motion (2.21) are rewritten as follows:

$$\begin{cases} \dot{\theta}_1 = \hat{\theta}_1 \\ \dot{\theta}_2 = \hat{\theta}_2 \\ u_1 = h_1(\theta_1, \theta_2, \hat{\theta}_1, \hat{\theta}_2, \dot{\hat{\theta}}_1, \dot{\hat{\theta}}_2) \\ u_2 = h_2(\theta_1, \theta_2, \hat{\theta}_1, \hat{\theta}_2, \dot{\hat{\theta}}_1, \dot{\hat{\theta}}_2) \end{cases} \tag{3.50}$$

Since the considered system has fewer control variables (two) than state variables (four), it is sufficient to replace only two of the state variables with approximations and calculate the others from the equations of motion [13]. Since simple differentiation of the variables θ_i yields the variables $\hat{\theta}_i$, the most suitable choice in the present case is to choose θ_1 and θ_2 to replace by approximations. $\hat{\theta}_1$, $\hat{\theta}_2$, u_1 and u_2 are then calculated using the equations of motion (3.50).

For a first order system, the following type of approximation of the chosen state variables is suitable [13]:

$$\theta_i(t) = P_i(t) + \beta_i(t) \tag{3.51}$$

$$P_i(t) = \sum_{j=0}^{3} p_{ij} t^j \tag{3.52}$$

$$\beta_i(t) = \sum_{m=1}^{K} a_{im} \cos \frac{2m\pi t}{t_f} + \sum_{m=1}^{K} b_{im} \sin \frac{2m\pi t}{t_f} \tag{3.53}$$

(where p_{i0} is actually the constant term of the truncated Fourier series). We have here and will in the following assume that $t_0 = 0$.

3.4.1 Optimization

The natural choice for the set of unknowns pertaining to the optimization problem is the coefficients of the approximation: $\{p_{10}, p_{11}, p_{12}, p_{13}, a_{11}, a_{12}, \ldots, a_{1K}, b_{11}, b_{12}, \ldots, b_{1K}, p_{20}, \ldots\}$. The optimization process may then proceed as defined by the following description of the process of one iteration:

- Obtain the time histories of the variables chosen for approximation (in the present case: θ_1 and θ_2) from the coefficient vector that was the result of the last iteration
- Calculate the time histories of the rest of the state variables and of the control moments using the equations of motion
- Calculate the value of the objective function and the values of the constraint functions
- Generate the next update of the coefficient vector.

There is a possibility, however, to incorporate the boundary values of the problem in such a way that the number of unknowns in the optimization problem are reduced. Inserting the boundary values (3.49) into the expression for the approximation; equation (3.51), gives:

$$\begin{aligned}
\theta_i(0) &= P_i(0) + \beta_i(0) = \theta_{i0} \\
\theta_i(t_f) &= P_i(t_f) + \beta_i(t_f) = \theta_{if} \\
\dot{\theta}_i(0) &= \dot{P}_i(0) + \dot{\beta}_i(0) = \dot{\theta}_{i0} \\
\dot{\theta}_i(t_f) &= \dot{P}_i(t_f) + \dot{\beta}_i(t_f) = \dot{\theta}_{if}
\end{aligned} \tag{3.54}$$

Solving for the four coefficients of the polynomial, we obtain:

$$\begin{aligned}
p_{i0} &= -\sum_{m=1}^{K} a_{im} + \theta_{i0} \\
p_{i1} &= -\frac{2\pi}{t_f} \sum_{m=1}^{K} m b_{im} + \dot{\theta}_{i0} \\
p_{i2} &= \frac{6\pi}{t_f^2} \sum_{m=1}^{K} m b_{im} + \frac{3(\theta_{if} - \theta_{i0})}{t_f^2} - \frac{2\dot{\theta}_{i0} + \dot{\theta}_{if}}{t_f} \\
p_{i3} &= -\frac{4\pi}{t_f^3} \sum_{m=1}^{K} m b_{im} + \frac{2(\theta_{i0} - \theta_{if})}{t_f^3} + \frac{\dot{\theta}_{i0} + \dot{\theta}_{if}}{t_f^2}
\end{aligned} \tag{3.55}$$

The set of unknowns pertaining to the optimization problem can therefore be chosen to be those of the *free* initial and terminal values as well as the Fourier coefficients. For instance, for a problem containing only one state variable, whose initial and terminal positions are known but whose initial and terminal velocities are unknown, the vector of unknowns is: $\{\dot{\theta}_0, \dot{\theta}_f, a_1, a_2, \ldots, a_K, b_1, b_2, \ldots, b_K\}$. For this case, in each iteration of the optimization problem, the four coefficients of the polynomial $P(t)$ are calculated using $\dot{\theta}_0$ and $\dot{\theta}_f$ from the last update of the vector of optimization parameters as well as the two known boundary conditions. For the considered system, having two generalized coordinates, a procedure like the one described above is performed for each coordinate except that, for each one, only the end velocity is free (see Table 2.1). The considered system thus has two free boundary conditions and six fixed. The number of optimization parameters in the optimization problem is therefore $4K + 2$.

3.5 Discretization errors

The discretization errors introduced for each of the three methods can be split into three subcategories:

1. Errors due to the discretization of the control, or, in the inverse dynamics case (the Fourier method), the limb angles. Defining parameter:
   - `ode45`: M (number of discretization points for each control variable).
   - Temporal FEM: M (number of discretization points for each control variable).
   - Fourier method: K (number of Fourier coefficients in the discretization of each limb angle).
2. Errors from the numerical solution of the equations of motion. Defining parameter:
   - `ode45`: `RelTol`, `AbsTol` and `MaxStep`.
   - Temporal FEM: N (number of points (minus one) used for the discretization of the time domain), tolerance used for the Newton method based solver of the equations of motion.
   - Fourier method: No error, explicit evaluation of the control is possible.
3. Errors from the numerical minimization process. Defining parameters: `TolCon` and `TolFun`.

For our studies of convergence of the objective function with increasing fineness of the discretizations of the free variables (see Sections 4.3 and 5.2), it is preferable that the error corresponding to the discretization fineness (type 1 above) is the predominant one. This is necessary for the values of the objective function for varying finenesses to be justly comparable. An error analysis investigating the effect of the different types of errors mentioned above on certain features of the solution is presented in Section 5.4.1.

Chapter 4

Method

4.1 Implementation

To solve the numerical optimization problems connected to the three respective methods considered, Matlab is used. Mathematica was used for all algebraic manipulations involved in the derivation of the equations of motion. The results were then exported for use in Matlab.

4.1.1 Optimization

For finding the minimum of the objective function, Matlabs function `fmincon` is used. `fmincon` calls the objective function with the optimization parameters as input, which returns the value of the function. `fmincon` finds the gradient of the objective function either by numerical means or by evaluating an analytic expression if one such is supplied by the user. To check if the current optimization parameters yield a feasible solution, functions returning the values of nonlinear equality and/or inequality constraint functions are also called. In the same manner as for the objective function, the gradients of the nonlinear constraint functions are either calculated or supplied by the user. `fmincon` has a feature that enables checking the supplied gradients against ones calculated by numerical differentiation to ascertain the validity of the former.

If the gradient of the objective function is sufficiently small and the solution is feasible, a minimum is considered as having been found. `fmincon` has associated with it the parameters `TolFun` and `TolCon`, which specify the tolerance of the termination conditions connected to the objective function and the nonlinear constraint functions, respectively.

4.1.2 `ode45` method

Matlabs function `ode45` is used as one of the methods for solving the differential equations. This solver uses a callback function [3] which returns the value of the function and its derivative. The error tolerance can be set by using the two parameters `RelTol` and `AbsTol`. The solution at point i is sufficiently accurate if equation (4.1) is fulfilled.

$$\left|e(i)\right| \leq \max(\texttt{RelTol} \times y(i), \texttt{AbsTol}(i)) \tag{4.1}$$

$e(i)$ is the estimated error and $y(i)$ is the value of the function at point i. Furthermore, the maximum step size can be set using the `MaxStep` parameter. Suitable values for `RelTol`, `AbsTol` and `MaxStep` are found from the convergence study described in Section 4.3.1. When using the `ode45` method, the gradient of the objective function is found by numerical differentiation. Since the `ode45` method uses direct dynamics, the unknowns in the optimal control problem are the control moments. Thus, the optimization variables are the node values of the discretized moments.

4.1.3 Temporal finite element method

Analytical expressions for the gradients of the objective function and the nonlinear constraint functions are available when using the temporal finite element method (see Section 3.3). The accuracy in the solution of the equations of motion depends on the number of nodes in the time mesh used: N. As the `ode45` method, the temporal finite element method is a direct dynamics approach, so the optimization variables are the node values of the discretized moments.

4.1.4 Fourier method

The gradients of the objective function and the nonlinear constraint functions are not supplied as analytical expressions for the Fourier method and are thus calculated using numerical differentiation. The Fourier method uses inverse dynamics, which means that the unknowns of the numerical optimization problem are the coefficients of the parametrization of the limb angles (or a reduced set, see Section 3.4).

4.2 Initial guesses

For the optimization routine to successfully find the minimum of the objective function, a sufficiently good initial guess has to be made. If the initial guess is too far off, the optimization routine might diverge or a local (not being the global) minimum might be found. One possible way of avoiding these erroneous local minima is to apply some additional constraints (constraint number two in Table 2.2, although physically valid, was added for this purpose), but that usually means a higher computational cost in each iteration, so it is preferable to improve the initial guess instead. This is a process involving ad hoc measures and much trial and error.

4.2.1 Temporal finite element method

To create an initial guess for use with the temporal finite element method, the steps below were found to be suitable. The number of nodes in the discretizations of the applied moments was set to 4.

1. With no movement of the hip, find the moments that moves the foot from the initial angles to the final angles.
2. With the result from the previous step as an initial guess, increase the velocity by a small amount ($\approx$10%).
3. Repeat step 2 with the previous result as an initial guess until the final velocity is reached.

As the number of nodes in the discretizations of the applied moments are then increased, a new guess is generated by interpolating over the previous guess.

4.2.2 `ode45` method

The initial guess generated for the temporal finite element method can be used with the `ode45` based solver or a guess can be generated following a similar scheme.

4.2.3 Fourier method

The initial guess for the optimization coefficients for the Fourier method is generated using the limb angles from a previous solution found using the temporal finite element method. The coefficients in the expressions for the limb angles used in the Fourier method are found by fitting the respective expressions to the available temporal finite element solutions using a least squares based procedure. The final velocities, also included in the reduced version of the optimization parameter vector for the Fourier method (see Section 3.4), are simply taken from the temporal FEM solution.

4.3 Convergence study

The parameters controlling the fineness of the discretizations of the free variables are M for the `ode45` method and the finite element method and K for the Fourier method, as mentioned in Section 3.5. It is of interest to find values of these parameters with respect to which the value of the objective function has converged, that is, parameter values corresponding to values of the objective function that change little with additional increases in the parameters. To this end, the first step is to tune the tolerances associated with `fmincon` and, for the associated method, `ode45`, so that the solution is stable enough with respect to the value of the objective function. For the temporal finite element method and the Fourier method, this is achieved by simply lowering the tolerances (`TolFun` and `TolCon`) in `fmincon` to a sufficiently low level such that the solution is independent with respect to additional lowering of the tolerances. `ode45` produces quite unstable results with its default options. Therefore, both the tolerances in `fmincon` and the tolerance and maximum step size for `ode45` have to be lowered to generate a stable solution.

4.3.1 `ode45` method

The parameters that are varied during the convergence study of `ode45` are `AbsTol`, `RelTol`, `MaxStep` and `fmincon`'s `TolFun`. To reduce the number of parameters, `AbsTol` and `RelTol` are set equal at all times.

1. Let `TolFun` decrease in steps of a factor 10 while all other parameters are fixed at their default values. When the changes in the solution are small, a starting point is found.
2. To test `MaxStep`, let `MaxStep` vary such that the minimum number of time intervals in the solution varies from 10 to 100. Ie, `MaxStep` $= \{\Delta t/10 \quad \Delta t/20 \quad \Delta t/30 \quad ... \quad \Delta t/100\}$ where $\Delta t = t_f - t_0$. The value for `MaxStep` giving a stable/smooth enough E-curve is chosen as the preferred one.
3. Repeat point number 2 when decreasing `TolFun` by a factor 10. When the change in the solution is less than 0.1%, that value of `TolFun` is chosen as the preferred one.
4. Repeat point number 3 when decreasing `AbsTol` by a factor 10. When the change in the solution is less than 0.1%, that value of `AbsTol` is chosen as the preferred one.
5. The preferred number of node points in the discretizations of the moments, M^*, is chosen as the one for which its double value means a difference in the value of the objective function of at most 0.5%.

4.3.2 Temporal finite element method

Since both the number of nodes in the discretization of the limb angles, N, and the number of nodes in the discretization of the applied moments, M, affect the error, sufficiently large values for these parameters must be found. First, a sufficiently large value for N is found by solving for increasing N ($N = 25, 50, 75, 100, 150$) and for each N solve for increasing numbers of node points in the discretization of the applied moments ($M = 4, 5, 6...$). When the value of E changes with a maximum of 0.1% when N is doubled, that value of N is denoted N^*. M^* is found when the doubled number of node points yields an E-value within an 0.5% range.

4.3.3 Fourier method

The Fourier method uses the trapezoidal rule to calculate the integral in the objective function E. The suitable number of node points for this procedure is chosen as the one for which an increase by a factor of 2 keeps the value of E within 0.1%.

Chapter 5

Results and discussion

5.1 Problem specification

In the present section, the parameter values used in all of the simulations, unless otherwise indicated, are presented. The parameters are taken from [9] unless otherwise indicated.

Table 5.1: Notation and parameter values.

Symbol	Value	Description
m_1	6.86 kg	mass of thigh
m_2	3.28 kg	mass of shank
m_H	-	mass of hip platform (doesn't enter in equations of motion)
m_A	0 kg	mass of ankle/foot
a_1	0.460 m	length of thigh
a_2	0.430 m	length of shank
r_1	0.180 m	distance from hip joint to center of mass of thigh
r_2	0.181 m	distance from knee joint to center of mass of shank
$\bar{J}_1$	0.1188 kgm^2	moment of inertia about center of mass of thigh
$\bar{J}_2$	0.0504 kgm^2	moment of inertia about center of mass of shank
t_0	0 s	initial time of motion
t_f	0.42 s, see below	final time of motion
L	1.42 m, see below	total length of one step
λ_1	1	objective function weighting factor
λ_2	0.4	objective function weighting factor

The values of λ_1 and λ_2 were decided upon independently of external references. A comparison when λ_2 is varied is later made (Section 5.3.4). The motion of the hip platform is parameterized as shown in equation (2.14). The following values of the parameters are chosen:

Table 5.2: Notation and parameter values pertaining to the prescribed hip motion.

Symbol	Value
V	1.70 m/s, see below
h_0	0.64 m, see below
B_1	0.021 m [10]
B_2	0.016 m [10]
ω	6.28 s^{-1}, see below

The final time t_f of the motion was chosen towards the high end of the swing phase times found in the available references: [8], [9]. h_0 is chosen so that the foot is at ground level at the start of the swing phase, Section 5.1.1 contains more details. ω is defined as the angular frequency corresponding to the time of one whole period of the walking process, that is, the time of two steps. We denote this time by T, so that $\omega = 2\pi/T$. By symmetry, the time of one step is exactly half of that and the corresponding angular frequency thus exactly the double; 2ω. By [11], one swing phase lasts 42% of one period, so $\omega = 2\pi/(t_f/0.42) \approx 6.28$ s^{-1}.

5.1.1 Boundary conditions

The initial conditions shown in Table 5.3 below are taken from [8]. The terminal conditions are calculated as described below.

Table 5.3: Boundary conditions for the motion

Parameter	Value at $t = 0$	Value at $t = t_f$
$\theta_1(t)$	$\theta_{10} = -10^\circ$	$\theta_{1f} = 46.27^\circ$
$\theta_2(t)$	$\theta_{20} = -40^\circ$	$\theta_{2f} = 5.86^\circ$
$\dot{\theta}_1(t)$	$\dot{\theta}_{10} = 0$	$\dot{\theta}_{1f}$, free
$\dot{\theta}_2(t)$	$\dot{\theta}_{20} = 0$	$\dot{\theta}_{2f}$, free

Both the step length L and the constant component of the velocity of the hip V are chosen based on the end angle configuration $\theta_1(t_f) = 40^\circ$ and $\theta_2(t_f) = 10^\circ$. Figure 5.1 shows two snapshots of the walking process, using these end angles, which are one half period (i.e. the time of one step: $T/2$) apart. Note that while the backward leg in each snapshot touches the ground (by definition of h_0), because the limbs are different in length, the forward leg doesn't.

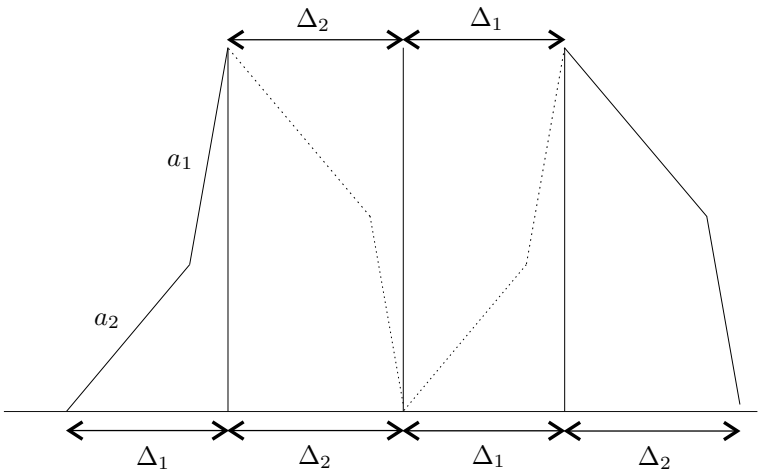

Figure 5.1: The construction used to calculate L and V.

V is chosen as the velocity required to traverse the distance $\Delta_1 + \Delta_2$ in the time of one step: $T/2$, that is:

$$V = (\Delta_1 + \Delta_2)/(T/2) = 2(a_1 \sin(10^\circ) + a_2 \sin(40^\circ) + a_1 \sin(40^\circ) + a_2 \sin(10^\circ))/T.$$

L is chosen as the total distance traversed by the foot in the x-direction using the configuration described above:

$$L = 2(\Delta_1 + \Delta_2) = 2(a_1 \sin(10^\circ) + a_2 \sin(40^\circ) + a_1 \sin(40^\circ) + a_2 \sin(10^\circ)).$$

The terminal limb angles for which the step length is equal to L, for which the foot is at ground level and which satisfy the knee constraint (see below), are subsequently solved for, which is done using `fmincon`. The result is shown in Table 5.3. A visualization of the boundary conditions as well as the motion of the hip during the swing phase can be seen in Figure 5.2.

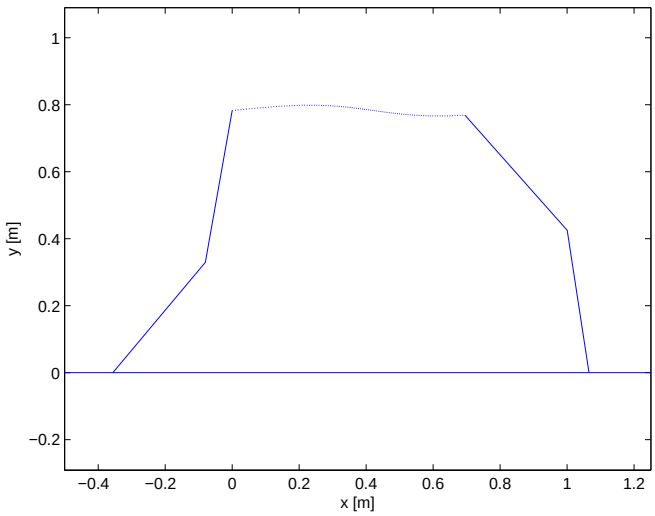

Figure 5.2: The start and end configurations of the system and the motion of the hip node in the swing phase of a step.

From the initial conditions, we can now also see that $h_0 = a_1 \cos\theta_1(0) + a_2 \cos\theta_2(0) = 0.64$ m.

5.1.2 Constraints

Using the above boundary conditions and setting $\delta = 0.1$, the constraints introduced in Section 2.1.4 become:

Table 5.4: Constraints on the motion

Constraint	Description
$\theta_2(t) - \theta_1(t) \leq 0$	The opening angle of the knee must be negative.
$\theta_1(t) \geq -10.1^\circ$	The thigh is not allowed to swing too far back.
$y_A(t) \geq 0$	The foot must be above or at ground level at all times.

The value of δ was an arbitrary choice that was found to work numerically. Although the second constraint is physically valid, it is also necessary in order to avoid solutions corresponding to local (not being global) minima in the optimization problem which involve clockwise rotation of the leg around the hip.

5.2 Convergence study

The behavior of the objective function corresponding to the optimal motion, E, as a function of the number of optimization parameters in the numerical optimization problem, is investigated in this section. The aim is to find values of the parameters controlling the fineness of the discretizations of the free variables (M for the `ode45` method and the finite element method and K for the Fourier method) that result in converged values of the objective function (see section 4.3). These values of the parameters are denoted M^* and N^*, respectively, and are considered to represent a "good enough" solution of the optimal control problem. In fact, there is more to tune than just these parameters, namely the tolerances involved in `fmincon` and, for the associated method, `ode45`. The details on how these were chosen are discussed in Section 4.3.

5.2.1 ode45

Figure 5.3 shows how the value of the objective function E varies with the number of node values in the discretizations of the control moments M and with the number of optimization parameters ($2M$) for some different values of `MaxStep`. Note that the curves for `MaxStep`=$\Delta t/80$

and `MaxStep`=$\Delta t/100$ coincide. The lowest value of M used was 4, a minimum could not be found for lower settings.

Although `MaxStep`=$\Delta t/80$ satisfies the criteria for convergence, `MaxStep` is chosen to be $\Delta t/100$ to make sure the relative error shown in Figure 5.21 is strictly decreasing in the vicinity of the values of the tolerances `TolFun` and `TolCon` that give convergence. The set of parameters that were found to satisfy the convergence criteria mentioned in Section 4.3.1 is presented in Table 5.5.

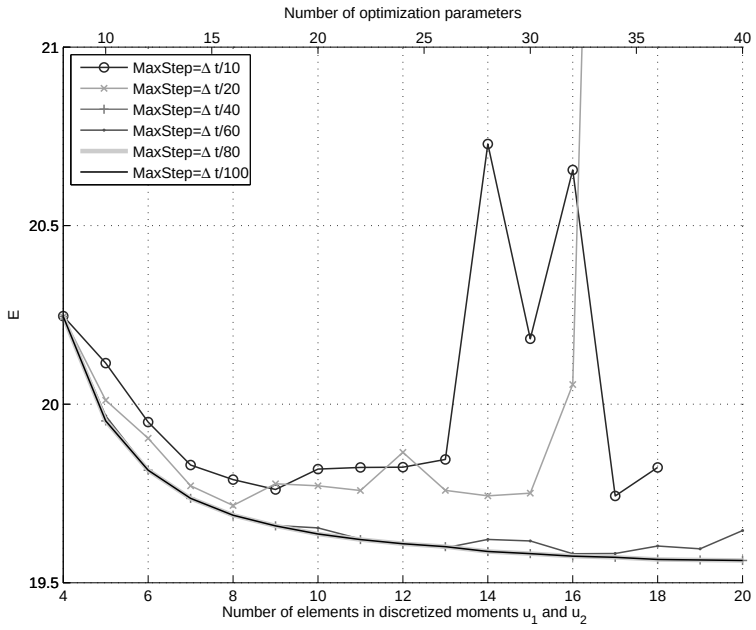

Figure 5.3: The value of the objective function E plotted against M and number of optimization parameters for some different values of `MaxStep`.

Table 5.5: Parameter values satisfying the convergence criteria for the `ode45` method.

Parameter	Value
`TolFun`, `TolCon`	10^{-5}
`MaxStep`	$\frac{\Delta t}{100}$
`RelTol`, `AbsTol`	10^{-6}
M	9

Note that $M = 9$ means $2M = 18$ optimization parameters.

With increasing fineness of the discretization, it is expected that the solution of the optimal control problem will converge against a theoretical exact solution. This means that all characteristics of the solution will converge towards stationary values. So should also, as discussed, the calculated energy E of the optimal solution. In this respect, the data presented in Figure 5.3 shows satisfactory behavior.

5.2.2 Temporal finite element method

To find suitable settings for the optimization tolerances, different values were tested. Figure 5.4 shows the objective function E plotted against the number of node values in the discretizations of the control moments M and the number of optimization parameters ($2M$) for tolerances `TolFun`=10^{-5}, `TolCon`=10^{-5} while Figure 5.5 shows the same but with tolerances `TolFun`=10^{-6}, `TolCon`=10^{-6}. Both plots include curves for a few different values of N. The lowest value of M used was 4, a minimum could not be found for lower settings. The set of parameters for which the the convergence criteria were deemed to be fulfilled is shown in Table 5.6.

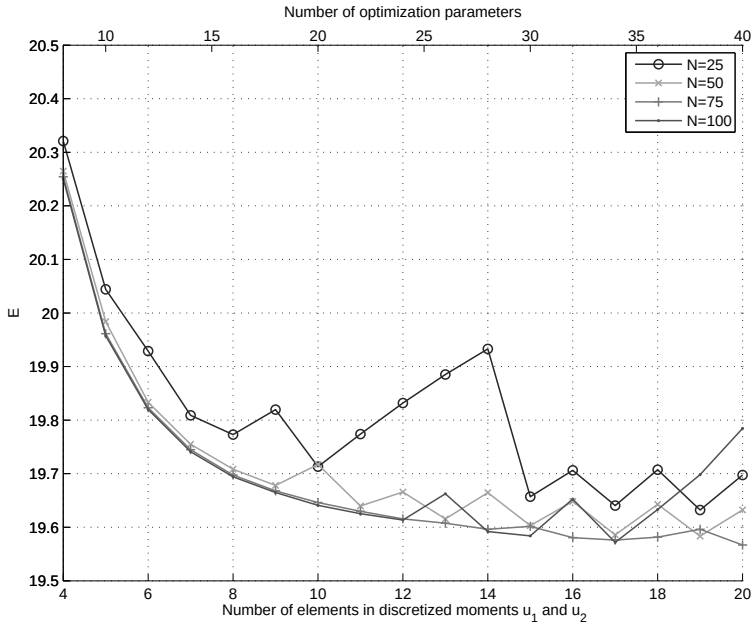

Figure 5.4: The value of the objective function E plotted against M and number of optimization parameters $2M$ for optimization tolerances `TolFun`=10^{-5}, `TolCon`=10^{-5}, for some different N.

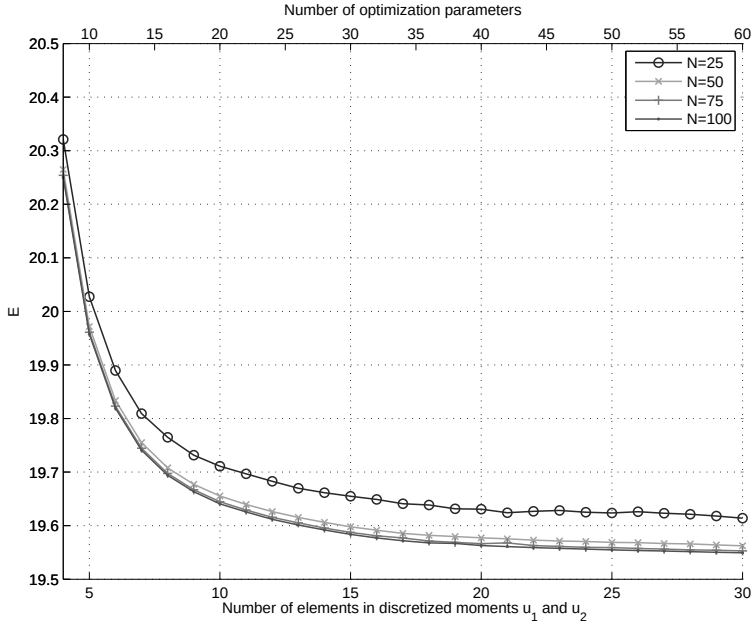

Figure 5.5: The value of the objective function E plotted against M and number of optimization parameters $2M$ for optimization tolerances `TolFun`=10^{-6}, `TolCon`=10^{-6}, for some different N.

Table 5.6: Parameter values satisfying the convergence criteria for the temporal finite element method.

Parameter	Value
`TolFun`, `TolCon`	10^{-6}
N	75
M	9

Note that $M = 9$ means $2M = 18$ optimization parameters.

When the tolerances `TolFun` and `TolCon` had been adjusted to make the associated errors in the objective function E decrease sufficiently, the objective function curves for the temporal finite element method also show the expected convergent behavior discussed above. For the finer tolerances, the curves are smooth for all values of N. Also, it is clear how the curves converge towards a fixed position as N increases.

5.2.3 Fourier method

Figure 5.6 shows the value of the objective function E corresponding to the optimal motion as solved using the Fourier method, plotted against the number of Fourier coefficients K as well as the number of optimization parameters $(4K + 2)$. The lowest value of K used was 3, a minimum could not be found for lower settings. The set of parameters found to satisfy the criteria for convergence is shown in Table 5.7.

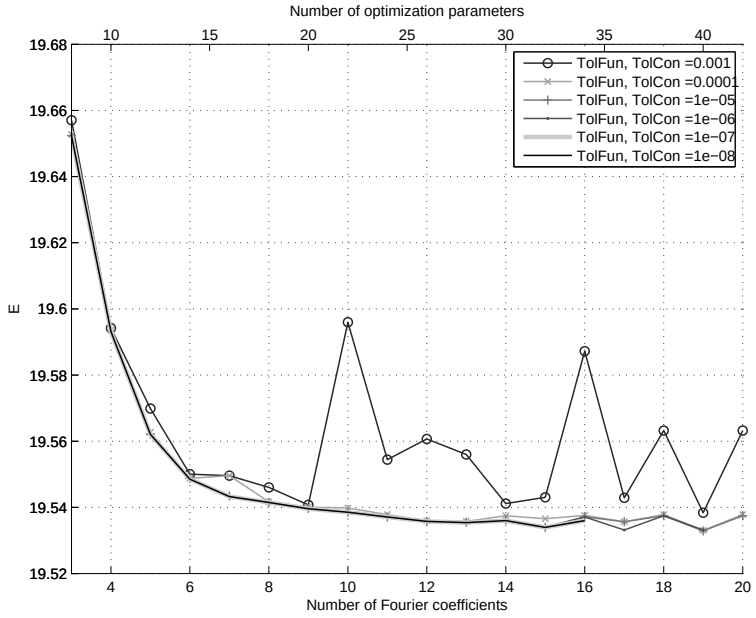

Figure 5.6: Optimized objective function value E plotted against number of Fourier coefficients K and number of optimization parameters $4K + 2$, for some different tolerances in the optimization routine.

Table 5.7: Parameter values satisfying the convergence criteria for the Fourier method.

Parameter	Value
`TolFun`, `TolCon`	10^{-4}
K	3

Note that $K = 3$ means $4K + 2 = 14$ optimization parameters.

The objective function curves for the Fourier method, too, show the expected convergence behavior, although the optimization tolerances had to be properly tuned before a satisfying result could be obtained.

5.3 Characteristics of optimal motion

In this section, the characteristics of the three solutions of the optimal control problem obtained using the three respective methods are explored. The chosen solution in each case is the one corresponding to that setup of the discretization and tolerance parameters of the problem (those controlling the fineness of discretizations as well as solver and optimization tolerances) deemed to be sufficient to generate a solution close enough to the exact one (see Tables 5.5, 5.6 and 5.7). Note that for the optimal solution, no constraint was active, except for the fact that constraint number 3 in Table 5.4 was active at the end points $t = t_0$ and $t = t_f$.

Figure 5.7 is a visualization of the motion of the leg according to the optimal solution obtained from the temporal finite element method. In fact, all three methods produce very similar solutions, as shown in Figure 5.8. Note that the curves for the `ode45` method and the temporal finite element method overlap in this figure. The solutions obtained seem to adequately resemble intuitive notions of human motion, which is pleasing.

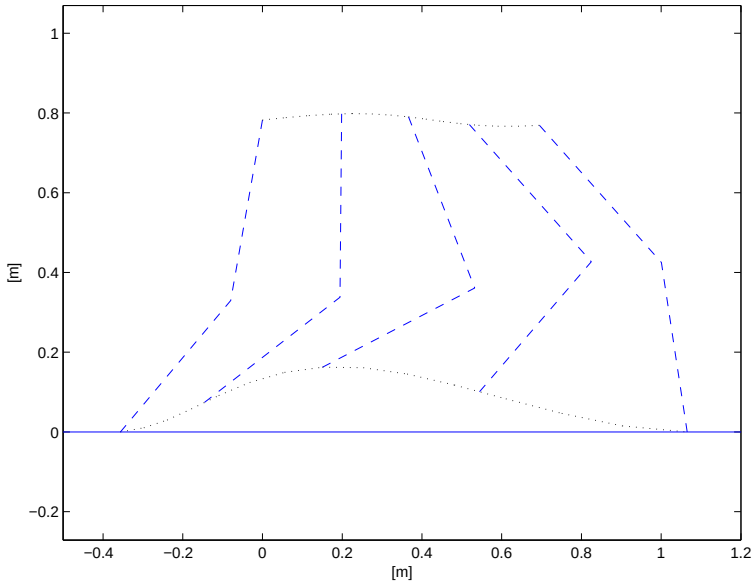

Figure 5.7: Visualization of the motion of the leg during the swing phase as solved by the temporal finite element method: trajectories of the hip and the ankle as well as snapshots of initial, final and some intermediate positions of the leg.

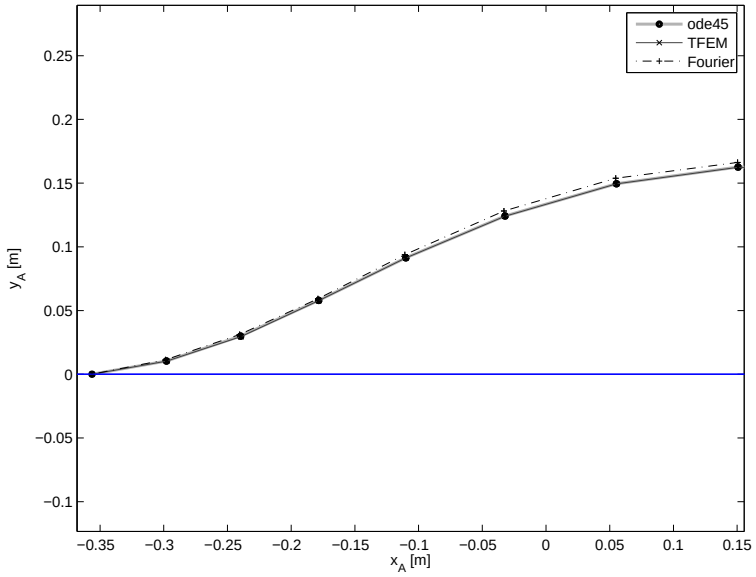

Figure 5.8: The trajectory of the ankle according to the chosen solutions for the three respective methods.

5.3.1 Objective function values

The values of the objective function E corresponding to the chosen solutions for the three respective methods are shown in Table 5.8.

Table 5.8: Values of the objective function E corresponding to the chosen solutions for the three respective methods.

Method	E	E_1	E_2
`ode45`	19.66	8.97	26.72
Temporal FEM	19.67	8.98	26.73
Fourier	19.59	8.97	26.57

The objective function value corresponding to the chosen solution is thus lower for the Fourier method than for the other two methods. As seen before, the number of optimization parameters used for the chosen Fourier method solution (14) is also lower than the number used for the other two methods (18). Thus, it seems that the approach used in the Fourier method to approximate the free variables by functions of the type (3.51) – (3.53) is better suited for the given type of problem than the approach of using piecewise linear functions on an evenly spaced time mesh to approximate the control variables. The inherent smoothness in the Fourier approximation versus the lack of the same in a piecewise linear approximation might be the reason. It is probable, however, that the piecewise linear approximations employed in the `ode45` method and the temporal FE method would work better for optimally chosen spacings in the meshes, perhaps giving better results than the Fourier method for the same number of optimization parameters. This subject has not been explored here, however.

5.3.2 Control torques

The time histories of the control torques $u_1(t)$ and $u_2(t)$ corresponding to the chosen solution are plotted for each method in Figure 5.9.

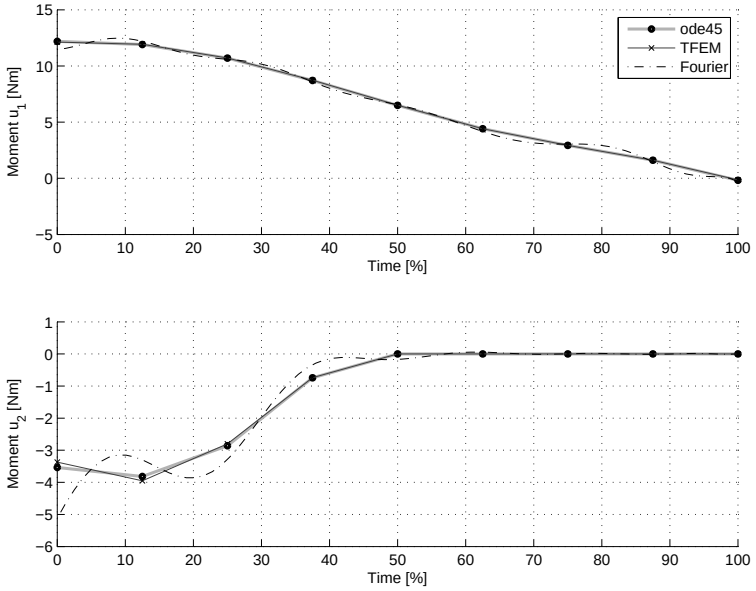

Figure 5.9: Time histories of the control torques $u_1(t)$ and $u_2(t)$ corresponding to the chosen solution for each of the three methods.

While the `ode45` method solution and the temporal FEM solution closely concide, the Fourier method solution behaves differently. As seen in Table 5.8 however, the Fourier solution has slightly greater success with respect to the value of the objective function E. The figure also indicates that, at least for $u_2(t)$, a different spacing of the time mesh used for the control variable discretizations for the former two methods will be more effective. It seems probable that concentrating the time points more to the first half of the time interval in the $u_2(t)$ case would yield a better solution and a lower value of E.

Figure 5.10 shows $u_1(t)$ and $u_2(t)$ corresponding to the chosen solution using the `ode45` method, for different numbers of node points in the discretizations of the moments, M. The darker the line, the higher the value of M. Figure 5.11 shows the same thing but for the temporal finite

element method. Figure 5.12 shows the same thing but for the Fourier method. Also, K is here the varying parameter between curves.

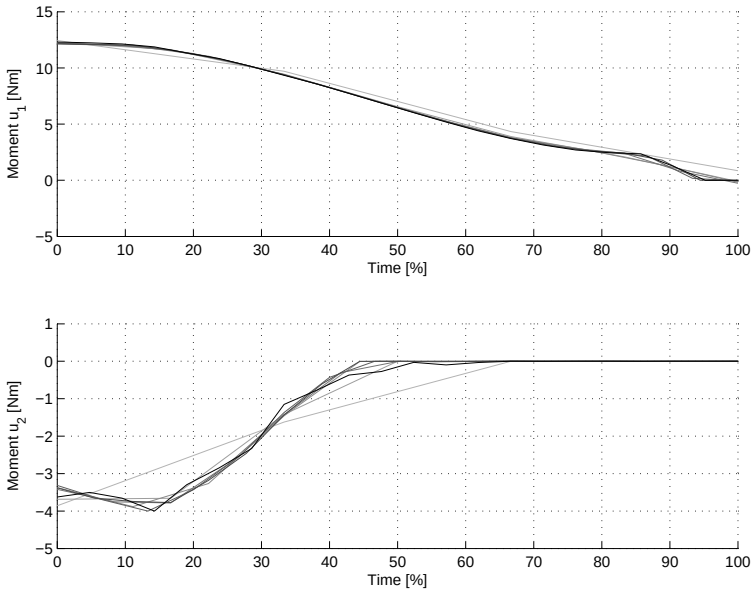

Figure 5.10: Time histories of the control torques $u_1(t)$ and $u_2(t)$ corresponding to the chosen solution using the `ode45` method, for different values of M. Darker line: higher value of M.

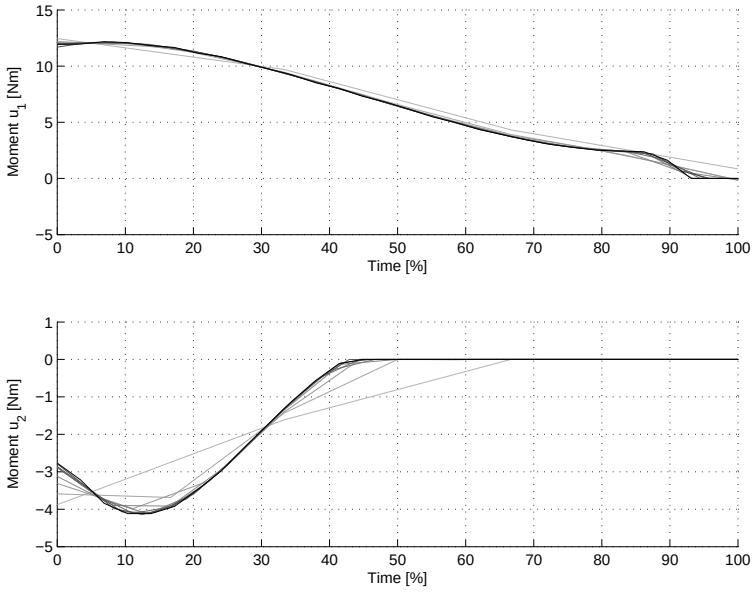

Figure 5.11: Time histories of the control torques $u_1(t)$ and $u_2(t)$ corresponding to the chosen solution using the temporal FE method, for different values of M. Darker line: higher value of M.

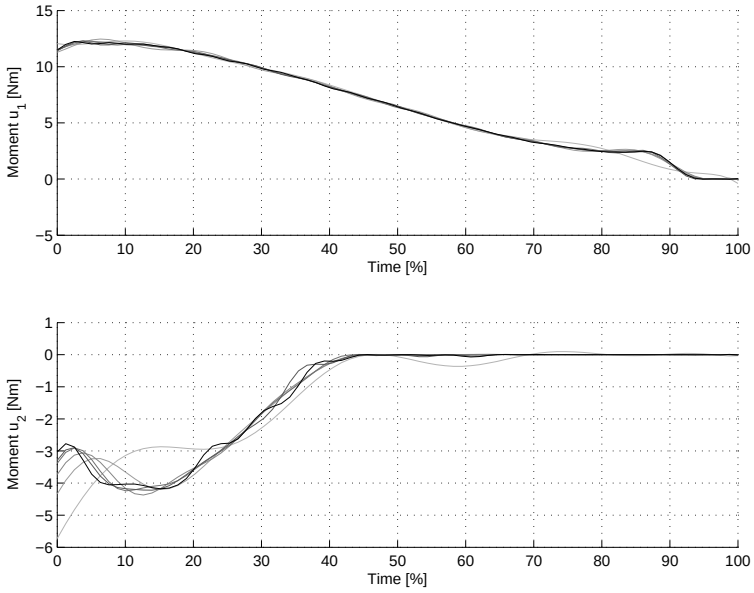

Figure 5.12: Time histories of the control torques $u_1(t)$ and $u_2(t)$ corresponding to the chosen solution using the Fourier method, for different values of K. Darker line: higher value of K.

These figures show from another point of view than the plots in Section 5.2 how the solution converges to a stationary one for increasing fineness of the discretizations.

Figure 5.13 shows $u_1(t)$ and $u_2(t)$ corresponding to the chosen solution for each of the three methods, with the exception that large values of the discretization parameters are used: $M = 22$ for the `ode45` method, $M = 30$ for the temporal FE method and $K = 20$. Notice how the Fourier method curve and the temporal finite element method curve closely resemble each other.

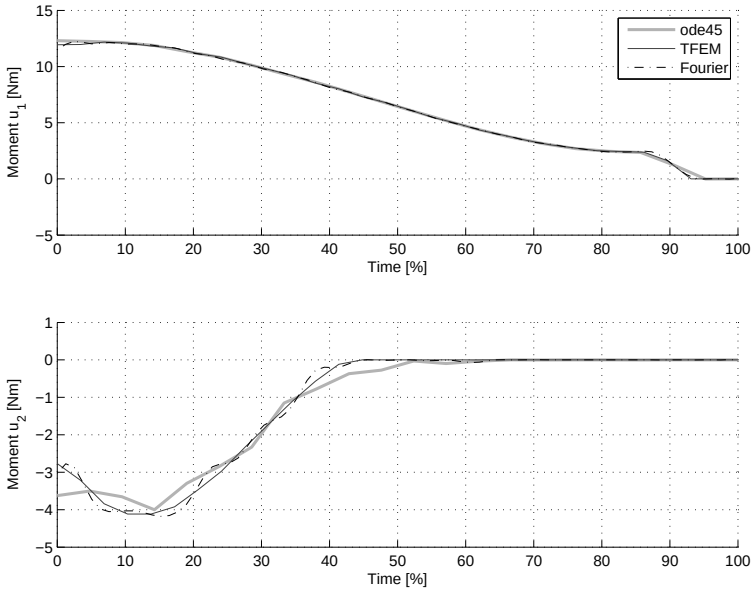

Figure 5.13: Time histories of the control torques $u_1(t)$ and $u_2(t)$ for all three methods using large values of the discretization parameters: $M = 22$ for the `ode45` method, $M = 30$ for the temporal FEM and $K = 20$.

5.3.3 Limb angles and angle velocities

The time histories of the limb angles $\theta_1(t)$ and $\theta_2(t)$ corresponding to the chosen solution are plotted for each method in Figure 5.14. The limb angle velocities $\dot{\theta}_1(t)$ and $\dot{\theta}_2(t)$ are shown in Figure 5.15.

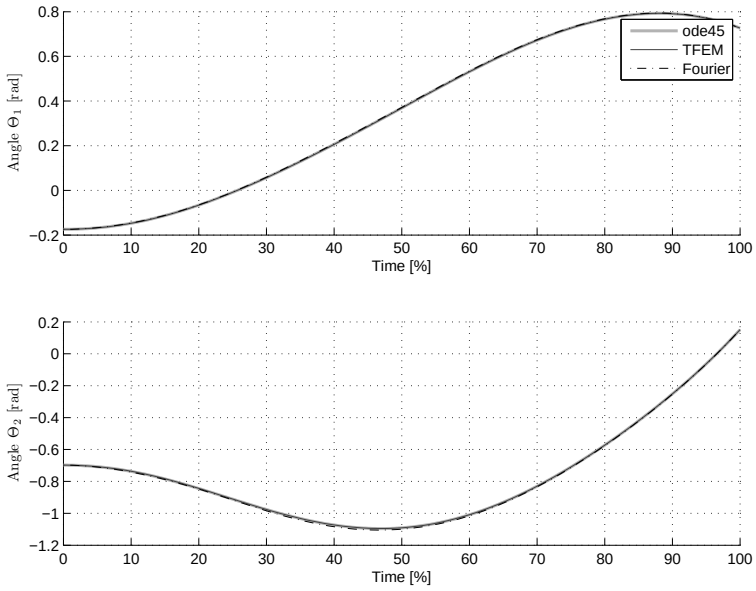

Figure 5.14: Time histories of the limb angles $\theta_1(t)$ and $\theta_2(t)$ corresponding to the chosen solution for each of the three methods.

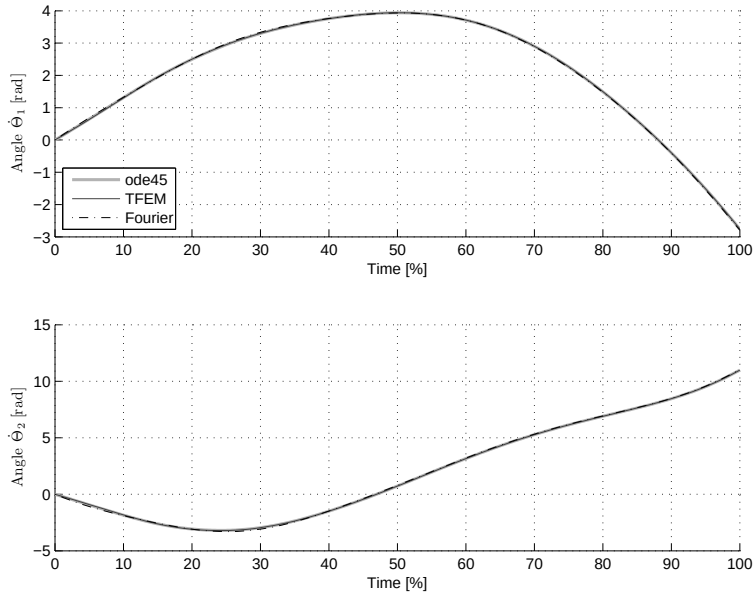

Figure 5.15: Time histories of the limb angle velocities $\dot{\theta}_1(t)$ and $\dot{\theta}_2(t)$ corresponding to the chosen solution for each of the three methods.

It is interesting to note that the time histories of the limb angles and the limb angle velocities clearly coincide to a much higher degree between the three methods than those of the control torques.

5.3.4 Dependency on λ_2

Figure 5.16 shows how the motion of the ankle corresponding to the chosen optimal solution changes with λ_2 for all three methods. The comparison was made in order to obtain an understanding of the impact of different choices of weighting between E_1 and E_2 for the objective function E. As can be seen, no significant changes occur, despite the fact that E_2 typically is close to a factor of 3 larger than E_1 (see Table 5.8).

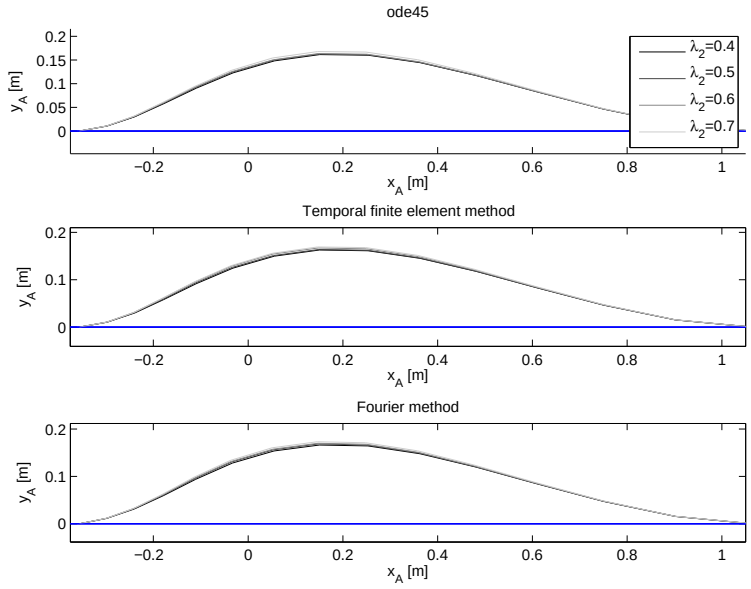

Figure 5.16: The motion of the ankle corresponding to the chosen optimal solution for all three methods for different values of λ_2.

5.4 Evaluation of numerical methods

5.4.1 Error estimation

This section contains plots of the errors in the values of the objective function E and its two components E_1 and E_2, as a function of certain ones of the discretization and tolerance parameters of the problem. The errors investigated are relative errors, where the quantity compared against in each case (denoted E^*) is the one corresponding to a set of discretization and tolerance settings deemed to be the among the best ones realistically possible.

The tolerance of the Newton solver used in each iteration of the solver for the limb angle nodes in the temporal finite element method was not varied in the following error analysis. To justify this, a comparison using the standard value 10^{-6} and the value 10^{-14}, which is almost the machine precision, was made, showing that the results differ by very little: in a test made with the number of optimization parameters varying from 8 to 40, the results differed by at most .0161%. Thus, the discretization error associated with the tolerance 10^{-6} is negligible. Each one of the discretization and tolerance parameters mentioned in Section 3.5, except for the aforementioned Newton method tolerance, are represented in the different plots below.

Figure 5.17 shows the value of the objective function E for varying numbers of optimization parameters for the different methods. Tolerances are chosen according to the chosen solutions presented in Tables 5.5 – 5.7. The figure shows that the objective function value corresponding to the Fourier method is always lower than that of the other two methods for the same number of optimization parameters, illustrating the superiority of the former compared to the others in solving the given type of problem. A possible reason for this, as discussed in Section 5.3.1, is that the types of functions employed as approximations in the Fourier method are better suited for representing the type of motion inherent to the given problem. However, the figure also indicates that the objective function value is still decreasing for the temporal FE method at the highest numbers of optimization parameters considered. Here, the value corresponding to the Fourier method has long since converged.

The reason that the value of the objective function was not plotted for numbers of optimization parameters above 44 for the `ode45` method was that for higher numbers, the objective function value showed very unstable behavior.

Figure 5.18 shows the relative error $(E^* - E_n^*)/E^*$ plotted against minimum number of time steps allowed (which is equal to $\Delta t/$`MaxStep`) for the `ode45` method. E_n^* denotes the value of the objective function corresponding to a certain value of the varying parameter.

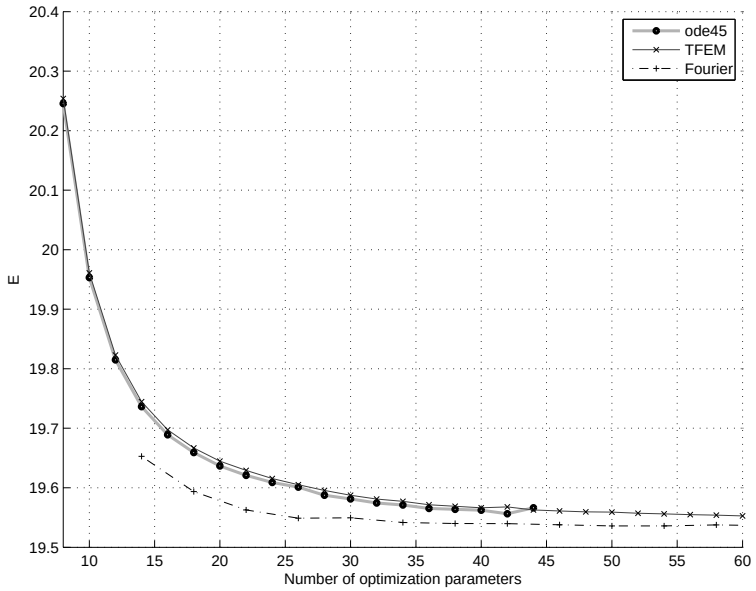

Figure 5.17: Objective function value as a function of the number of optimization parameters for the three methods.

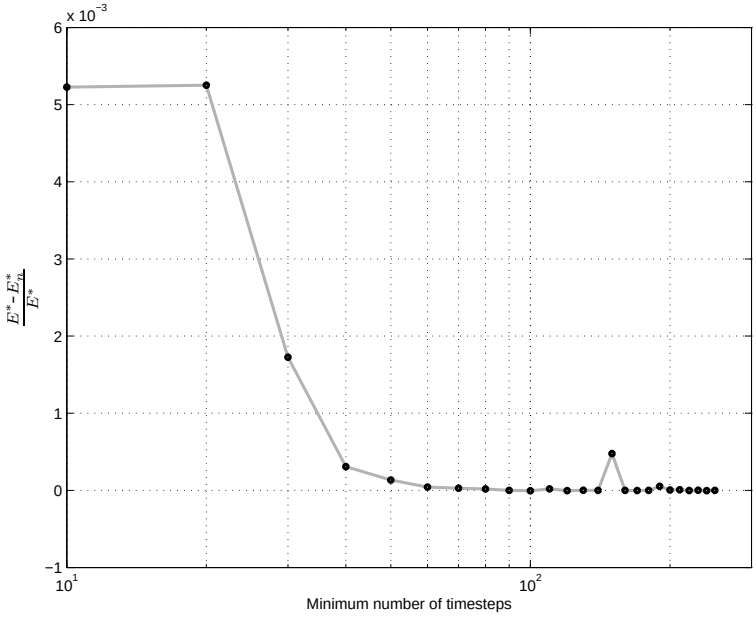

Figure 5.18: Relative error versus minimum number of time steps allowed ($\Delta t/$`MaxStep`) for the `ode45` method.

The error shows expected behavior in that it converges towards zero for increasing values of the minimum number of time steps.

Figure 5.19 shows the relative error $(E^* - E_n^*)/E^*$ plotted against the tolerances `AbsTol` and `RelTol` (set equal) for the `ode45` method.

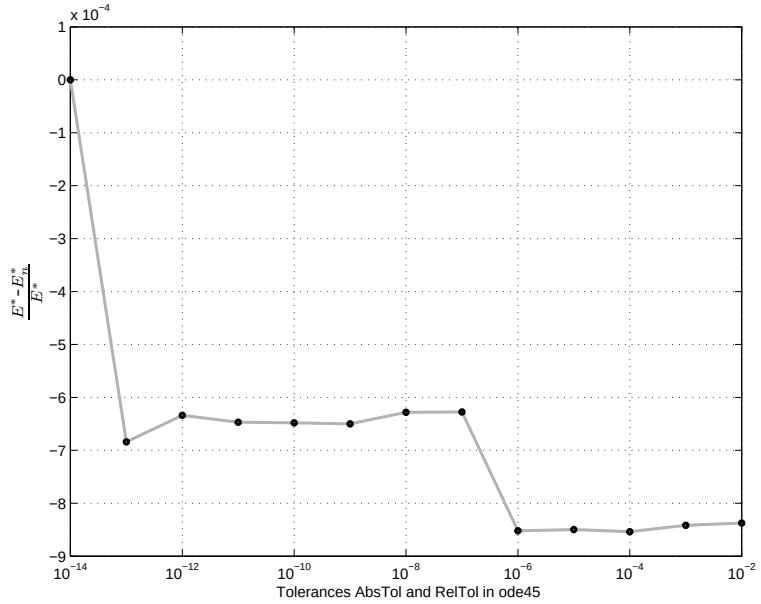

Figure 5.19: Relative error versus the tolerances `AbsTol` and `RelTol` (set equal) for the `ode45` method.

The behavior of the relative error is not quite as expected: for decreasing tolerances, no clear decrease in the error occurs. The reason might be that the error connected to the considered tolerances is negligible compared to other errors for the whole interval considered in the plot.

Figure 5.20 shows the relative error $(E^* - E_n^*)/E^*$ plotted against number of discretization points (minus one), N, used for the discretization of the time domain in the temporal FEM solver.

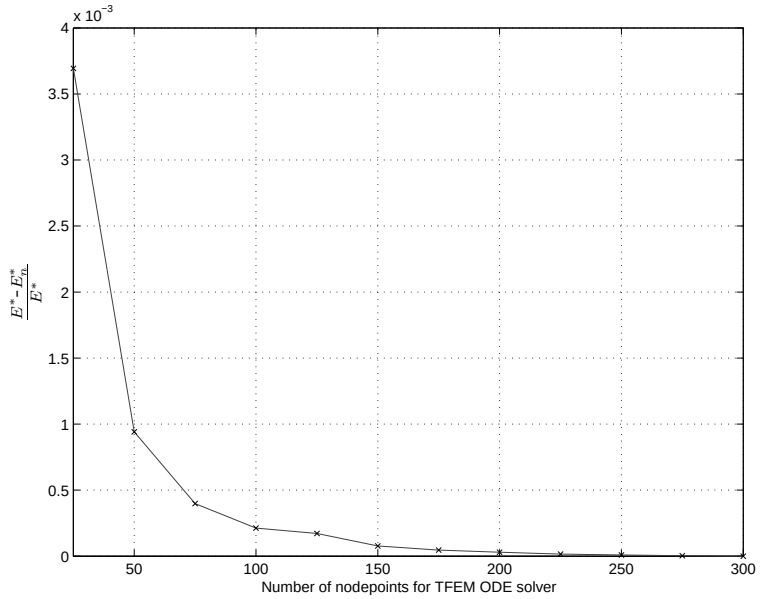

Figure 5.20: Relative error versus number of discretization points (minus one), N, used for the discretization of the time domain in the temporal FEM solver.

The plot shows a clear convergence of the error towards zero for increasing N.

Figure 5.21 shows the relative error $(E^* - E_n^*)/E^*$ plotted against the tolerances `TolFun` and `TolCon` (set equal) for each of the three methods.

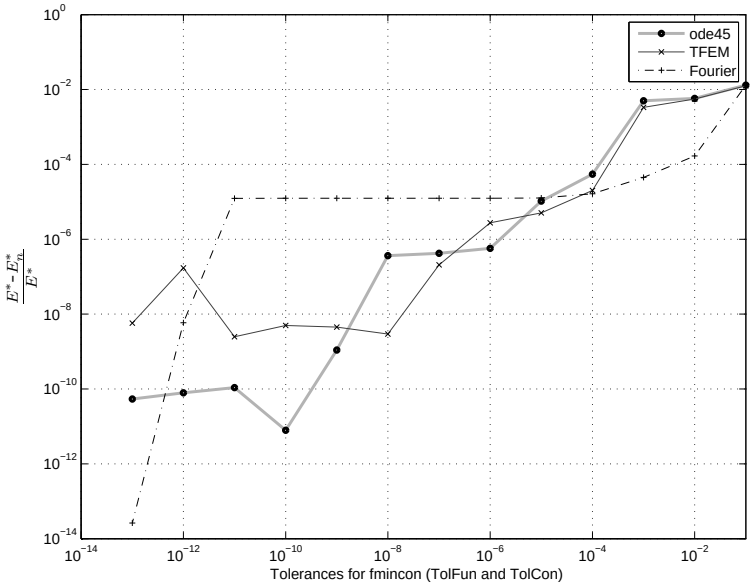

Figure 5.21: Relative error versus the tolerances `TolFun` and `TolCon` (set equal) for each of the three methods.

The error shows, for all three methods, a tendency to decrease for decreasing values of the tolerances, as it should.

5.4.2 CPU time

As discussed, the different methods have different computational benefits. The `ode45` method uses precompiled code, which significantly increases the efficiency of its execution. The temporal finite element method has the possibility to supply analytical expressions for the gradients of the objective and constraint functions to the optimizer. The Fourier method enables an exact solution of the equations of motion and an efficient way of satisfying the boundary conditions. A plot of the CPU time required to solve the optimization problem versus number of optimization parameters, for each of the three methods, is shown in Figure 5.22. Except for the discretization parameters, the settings used for this plot are the ones in Tables 5.5 – 5.7.

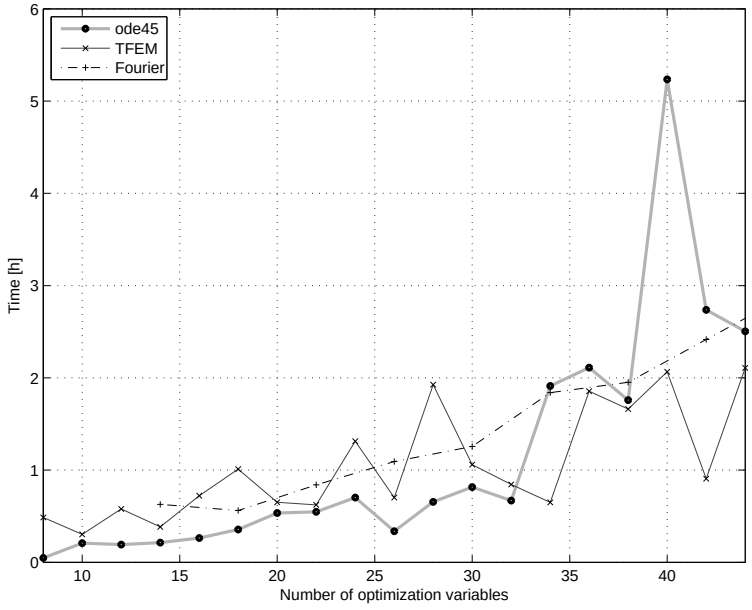

Figure 5.22: CPU time required to solve the optimization problem versus number of optimization parameters, for each of the three methods.

It seems that none of the three methods clearly yield consistently lower computational times than any other. It should be mentioned, however, that there is still much to do to make sure the different methods are efficiently coded. This is especially true for the temporal finite element method and the Fourier method. This task has not been given the highest priority during the project so there is potentially room for improvement in this respect. Also, the quality of the initial guess has an effect on the solution time. The guesses are in closely correspondence, however, since they are calculated from the same original guess, as described in Section 4.2.

Chapter 6

Conclusions

In this contribution, three methods for solving a biomechanics-related optimal control problem were analyzed and compared: a method based on Matlabs `ode45`, one based on a temporal finite element method and one based on Fourier series approximations and inverse dynamics. The considered mechanical system was a model of a human leg during the swing phase of a step and parameters based on data from real human gait were chosen. The numbers of optimization parameters (numerical degrees of freedom in the optimization problem) that were deemed necessary for a good enough solution of the optimal control problem for tolerance settings found suitable were 18, 18 and 14 for the respective methods. The Fourier method was thus the most efficient for solving a problem of the given type. The reason might be that the types of functions employed as approximations in the Fourier method are better suited for representing the type of motion inherent to the given problem. The characteristics of the chosen solutions were very similar between the methods and plots and animations of the motion of the leg, satisfyingly enough, looked quite true to life.

Numerical optimization is a complex problem so satisfactory solutions could sometimes be quite evasive. A good initial guess and properly tuned tolerances were necessary to obtain good results and the work with developing a system for assuring this took up a large portion of the time spent on the project. When such a system is in place however, the optimization routine can be quite powerful and able to efficiently solve even the fairly large-scale problems resulting from the highest settings of the discretization parameters used in this project.

The project ended up having a chiefly numerical focus and the model of the leg was never extended to make it more realistic, therefore comparisons with test results were not made. However, most of the original goals of the project were achieved. Optimal control solvers based on each of the three methods were successfully constructed and they all gave similar results. The feature to supply analytical expressions for the gradients of the objective function and the constraint functions was successfully implemented in the temporal finite element based optimizer.

6.1 Future work

A functional framework in Matlab for solving the optimal control problem using any one of the considered methods has been established and further development of the methods used can quite easily be incorporated in the code. For example, a system for error estimation connected to a local mesh refinement routine could be built into the temporal finite element solver. In fact, this is considered to be the natural next step in the development of the method and one that holds promise in terms of increased computational effectivity of the method. The next step in the development of the Fourier method could be to explore the possibility of supplying analytical expressions for the gradients of the objective function and the constraint functions.

The Matlab code was written with modularity in mind and the mechanical system used in the code is exchangeable to some degree. It could, for instance, be replaced by a more advanced model for human gait, or even by a completely different system.

Also, much is still to be done to optimize the performance of the code, a task that has not been given the highest priority during the work on the thesis. The comparison of the different solution methods regarding CPU-time will not be fair until each of their implementations are as efficiently coded as they can be. Translating the code into another programming language is also a thinkable option to improve its performance.

Bibliography

[1] Goldstein, H., Poole, C., Safko, J. (2002): *Classical Mechanics, Third Edition*. Addison Wesley, San Fransisco.

[2] `http://www.mathworks.com/access/helpdesk/help/toolbox/optim/ug/fmincon.html` (May 2009)

[3] `http://www.mathworks.com/access/helpdesk/help/techdoc/ref/ode45.html` (May 2009)

[4] Eriksson, A. (2008): Optimization in target movement simulations. *Comput. Methods Appl. Mech. Engrg.*, Vol. 197, 4207 – 4215.

[5] Eriksson, A. (2007): Temporal finite elements for target control dynamics of mechanisms. *Computers and Structures*, Vol. 85, 1399 – 1408.

[6] Kaphle, M., Eriksson, A. (2008): Optimality in forward dynamics simulations. *Journal of Biomechanics*, Vol. 41, 1213 – 1221.

[7] Eriksson, K., Estep, D., Hansbo, P., Johnson, C. (1996): *Computational Differential Equations*. Studentlitteratur, Lund.

[8] Nakamura, M., Mori, M., Nishii, J. (2004): *Trajectory planning for a leg swing during human walking*. 2004 IEEE International Conference on Systems, Man and Cybernetics.

[9] Vaughan, C. L., Davis, B. L., O'Connor, J. C. (1999): *Dynamics of Human Gait, Second Edition*. Kiboho Publishers, Cape Town.

[10] Berbyuk, V., Lytwyn, B., Demydyuk, M. (2005): Energy-optimal Control of Underactuated Bipedal Locomotion Systems *MULTIBODY DYNAMICS, ECCOMAS Thematic Conference*. Madrid, Spain.

[11] Berbyuk, V. E. (1989): *Dynamics and optimization of a robotechnical system*. Naukova Dumka, Kiev.

[12] Berbyuk, V. E., Nishchenko, N. I. (2001) Mathematical Design of Energy-Optimal Femoral Prostheses *Journal of Mathematical Sciences*, Vol. 107, 3647 – 3654.

[13] Nagurka, M. L., Yen, V. (1990): Fourier-Based Optimal Control of Nonlinear Dynamic Systems. *Journal of Dynamic Systems, Measurement, and Control*, Vol. 112, 17 – 26.

[14] Berbyuk, V. E. (1982): Program level of a control system for a walking robot, for a case of motion at a specified velocity, *J. Mechanics of Solids*, Vol. 17, 43 – 48.