Numerical Methods for ODE Optimal Control Problems

MINOA Summer School

University of Heidelberg, June 21-25, 2021

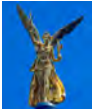
Matthias Gerdts

Institute of Applied Mathematics and Scientific Computing
Department of Aerospace Engineering
Universität der Bundeswehr München (UniBw M)
matthias.gerdts@unibw.de
http://www.unibw.de/ingmathe, http://www.optimal-control.de

Fotos: http://de.wikipedia.org/wiki/München
Magnus Manske (Panorama), Luidger (Theatinerkirche), Kurmis (Chin. Turm), Arad Mojtahedi (Olympiapark), Max-k (Deutsches Museum), Oliver Raupach (Friedensengel), Andreas Praefcke (Nationaltheater)






Universität der Bundeswehr München

Universität der Bundeswehr München
Professur für Ingenieurmathematik

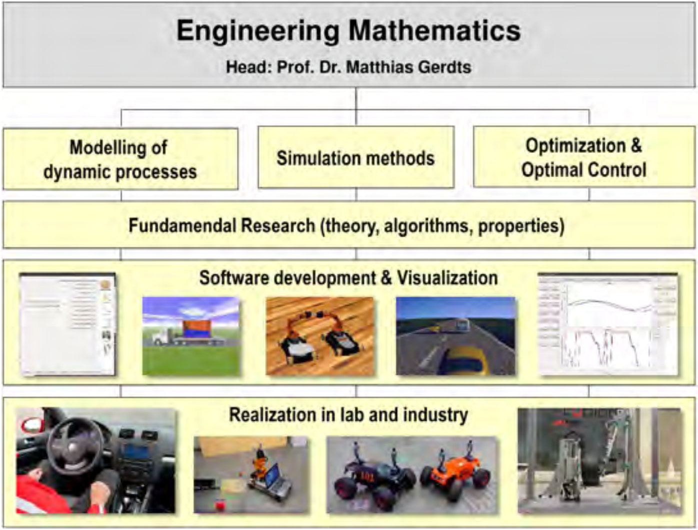
Engineering Mathematics @ Department of Aerospace Engineering

Prof Dr. Matthias Gerdts
Head of Engineering Mathematics Group
matthias.gerdts@unibw.de
http://www.unibw.de/ingmathe
http://www.optimal-control.de

Curriculum Vitae

1992 – 1997	studies of Mathematics with minor Computer Science, TU Clausthal
1997 – 2001	Phd, TU Clausthal/Uni Bayreuth, “Simulation of test-drives at driving limit”
2001 – 2004	assistant lecturer, Uni Bayreuth
2004 – 2007	Junior Professor (W1) for “Optimal Control”, Uni Hamburg
2006	Habilitation, Uni Bayreuth
2007 – 2009	Lecturer for “Mathematical Optimization”, University of Birmingham, U.K.
2009 – 2010	Associate Professor (W2) for “Optimal Control”, Uni Würzburg
since 2010	Full Professor (W3) for “Engineering Mathematics”, UniBw München

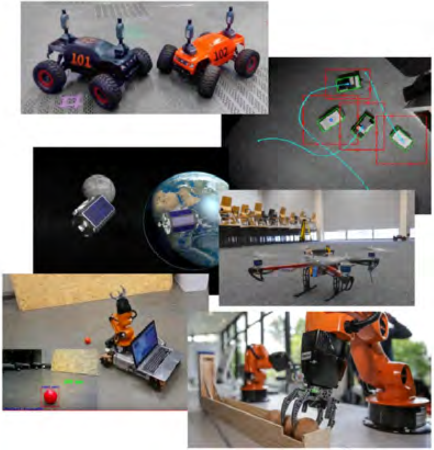
Research @ Engineering Mathematics



Research @ Engineering Mathematics

Research topics

- Optimization and optimal control of dynamic systems
- Sensitivity analysis (influence of parameters)
- Model-predictive control and real-time optimization
- Modelling, simulation, and parameter identification



Applications

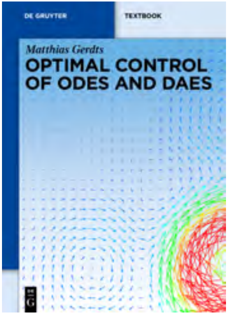
- Automatic driving and virtual testdrives
- Path planning in robotics
- Flight path optimization
- Docking maneuver
- Collision avoidance
- Control of co-operative systems

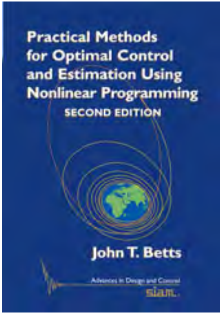
Schedule

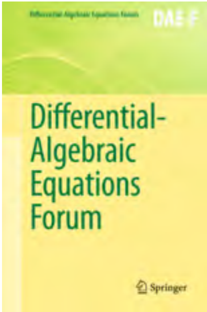
Nonlinear Optimal Control Problems: Methods and Applications

Time (CEST)	Monday June 21	Tuesday June 22	Wednesday June 23	Thursday June 24	Friday June 25
9:00-10:00	MINOA training till 13:00	MINOA training till 13:00			
10:00-11:30			Mathias Gerdts: Numerical Methods OCP-ODE	Christian Kirches: Mixed Integer OCP	Industrial session: Otmar Gehring (Daimler)
11:30-14:00			Break	Break	Break
14:00-15:30		Opening Ekaterina Kostina: Introduction OCP	Mathias Gerdts: Numerical Methods OCP-ODE	Christian Kirches: Mixed Integer OCP	Karl Worthmann: Feedback control and NMPC
15:30-16:00		Interaction	Interaction	Interaction	Interaction
16:00-17:30			Industrial session Armin Nurkanovic (Siemens)	Sven Leyffer: Mixed Integer OCP-PDE	Sven Leyffer: Mixed Integer OCP-PDE

Literature and Resources







Contents

Introduction

Direct Discretization : First Discretize – Then Optimize
- Overview
- Full Discretization (Collocation, Direct Transcription)
- Numerical Solution of Discretized Problem
- Reduced Discretization (Shooting)
- Adjoint Estimation
- Convergence

Applications

Contents

Introduction

Direct Discretization : First Discretize – Then Optimize
- Overview
- Full Discretization (Collocation, Direct Transcription)
- Numerical Solution of Discretized Problem
- Reduced Discretization (Shooting)
- Adjoint Estimation
- Convergence

Applications

Optimal Control

Basic optimal control task:

Control a dynamic system through control inputs subject to constraints such that an objective function is minimized or maximized!

Optimal Control

Basic optimal control task:

Control a dynamic system through control inputs subject to constraints such that an objective function is minimized or maximized!

What is a dynamic system?

Optimal Control

Basic optimal control task:

Control a dynamic system through control inputs subject to constraints such that an objective function is minimized or maximized!

What is a dynamic system?

- A dynamic system is a (technical, biological, economical, . . .) system in motion.

Optimal Control

Basic optimal control task:

Control a dynamic system through control inputs subject to constraints such that an objective function is minimized or maximized!

What is a dynamic system?

- A dynamic system is a (technical, biological, economical, ...) system in motion.
- The state $x(t) \in \mathbb{R}^n$ of the system (e.g. position, velocity, ...) changes with time t.

Optimal Control

Basic optimal control task:

Control a dynamic system through control inputs subject to constraints such that an objective function is minimized or maximized!

What is a dynamic system?

- A dynamic system is a (technical, biological, economical, ...) system in motion.
- The state $x(t) \in \mathbb{R}^n$ of the system (e.g. position, velocity, ...) changes with time t.
- The system at time t can be influenced by a control input $u(t) \in \mathbb{R}^m$.

Optimal Control

Basic optimal control task:

Control a dynamic system through control inputs subject to constraints such that an objective function is minimized or maximized!

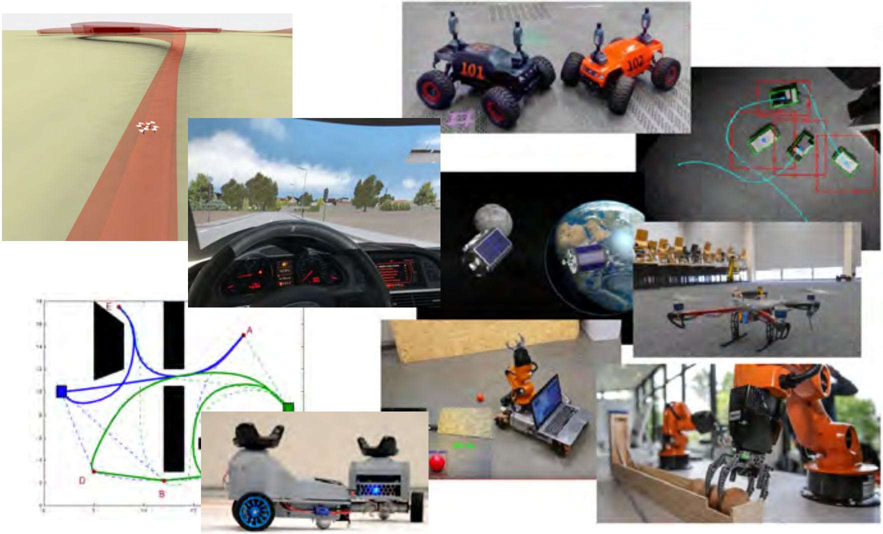
What is a dynamic system?

- A dynamic system is a (technical, biological, economical, ...) system in motion.
- The state $x(t) \in \mathbb{R}^n$ of the system (e.g. position, velocity, ...) changes with time t.
- The system at time t can be influenced by a control input $u(t) \in \mathbb{R}^m$.

Mathematical model:

$$
\begin{aligned}
x(0) &= x_0 && \text{(given initial state)} \\
x'(t) &= f(t, x(t), u(t)) && (t \geq 0)
\end{aligned}
$$

Optimal Control



Open-loop Control vs Closed-loop Control

Open-loop control

Apply a **time-dependent** control $u(t)$ (**open-loop control**) to the dynamic system:

$$x'(t) = f(t, x(t), u(t)), \qquad x(0) = x_0.$$

Open-loop Control vs Closed-loop Control

Open-loop control

Apply a **time-dependent** control $u(t)$ (**open-loop control**) to the dynamic system:

$$x'(t) = f(t, x(t), u(t)), \qquad x(0) = x_0.$$

$\leadsto$ an open-loop control cannot react on perturbations in x!

Open-loop Control vs Closed-loop Control

Open-loop control

Apply a **time-dependent** control $u(t)$ (**open-loop control**) to the dynamic system:

$$x'(t) = f(t, x(t), u(t)), \qquad x(0) = x_0.$$

$\leadsto$ an open-loop control cannot react on perturbations in x!

Closed-loop control / feedback control

Application of a **feedback control law** of type $u(t) = \mu(t, x(t))$ to the dynamic system yields a **closed-loop system**:

$$x'(t) = f(t, x(t), \mu(t, x(t))), \qquad x(0) = x_0.$$

Open-loop Control vs Closed-loop Control

Open-loop control

Apply a **time-dependent** control $u(t)$ (**open-loop control**) to the dynamic system:

$$x'(t) = f(t, x(t), u(t)), \qquad x(0) = x_0.$$

$\rightsquigarrow$ an open-loop control cannot react on perturbations in x!

Closed-loop control / feedback control

Application of a **feedback control law** of type $u(t) = \mu(t, x(t))$ to the dynamic system yields a **closed-loop system**:

$$x'(t) = f(t, x(t), \mu(t, x(t))), \qquad x(0) = x_0.$$

$\rightsquigarrow$ a feedback control is able to react on perturbations in x!

Open-loop Control

In this lecture:

Open-loop control

Apply a **time-dependent** control $u(t)$ (**open-loop control**) to the dynamic system:

$$x'(t) = f(t, x(t), u(t)), \qquad x(0) = x_0.$$

Extension to closed-loop control is possible using model-predictive control!

Optimal Control and Path Planning Tasks

Path Tracking Task

Follow a given (optimal) reference trajectory

$$(x_{ref}(t), u_{ref}(t)) \qquad (t \in [0, t_f]) \ !$$

vs

Path Planning Task

Compute a (locally) optimal trajectory

$$(x_{ref}(t), u_{ref}(t)) \qquad (t \in [0, t_f]) \ !$$

Optimal Control Problem of Tracking Type

OCP of Tracking Type

Minimize weighted tracking error

$$\frac{1}{2}\int_0^{t_f} \|x(t) - x_{ref}(t)\|_V^2 + \|u(t) - u_{ref}(t)\|_W^2 \, dt$$

subject to the constraints

$$\begin{aligned} x'(t) &= f(t, x(t), u(t)) && (t \in [0, t_f]) \\ x(t) &\in X && (t \in [0, t_f]) \\ u(t) &\in U && (t \in [0, t_f]) \\ x(0) &= x_0 \end{aligned}$$

x_0: initial state, t_f: final time, (x_{ref}, u_{ref}): reference trajectory, X: state set, U: control set, $\|z\|_M := \left(z^\top M z\right)^{1/2}$: weighted norm

General Optimal Control Problem

General OCP

Minimize

$$\varphi(x(t_f)) + \int_0^{t_f} f_0(x(t), u(t))\, dt$$

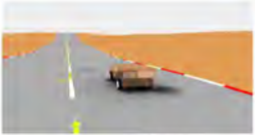
subject to the constraints

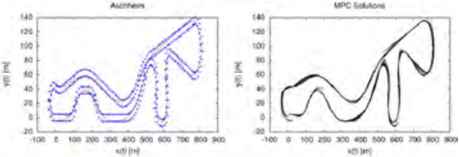
$$\begin{aligned} x'(t) &= f(t, x(t), u(t)) && (t \in [0, t_f]) \\ x(t) &\in X && (t \in [0, t_f]) \\ u(t) &\in U && (t \in [0, t_f]) \\ x(0) &= x_0 \end{aligned}$$

x_0: initial state, t_f: final time, X: state set, U: control set, φ: terminal costs, f_0: running costs

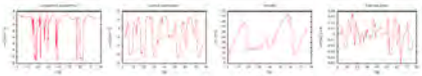
Application: Automatic Driving

- Modelling of an “optimal” driver (time minimal, fuel efficient)
- Consideration of track bounds and obstacles
- Online optimization

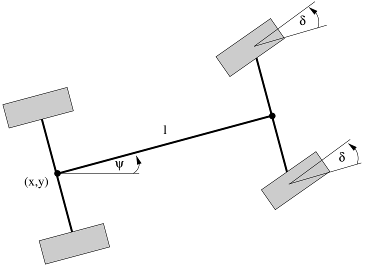




Left to right: longitudinal acceleration, lateral acceleration, velocity, slip angle



Virtual Testdrive: Mathematical Model

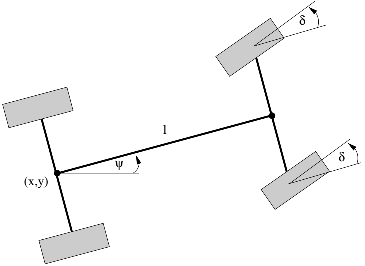


Notation:

δ	steering angle
v	velocity
ψ	yaw angle
ℓ	distance front to rear axle
(x, y)	reference point

Virtual Testdrive: Mathematical Model

$$
\begin{aligned}
x'(t) &= v(t)\cos\psi(t), \\
y'(t) &= v(t)\sin\psi(t), \\
\psi'(t) &= \frac{v(t)}{\ell}\tan\delta(t) \\
v'(t) &= u_2(t) \\
\delta'(t) &= u_1(t)
\end{aligned}
$$



Notation:

δ	steering angle
v	velocity
ψ	yaw angle
ℓ	distance front to rear axle
(x, y)	reference point

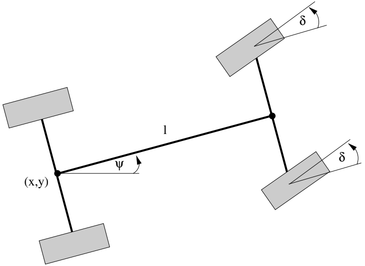
Virtual Testdrive: Mathematical Model

$$
\begin{aligned}
x'(t) &= v(t)\cos\psi(t),\\
y'(t) &= v(t)\sin\psi(t),\\
\psi'(t) &= \frac{v(t)}{\ell}\tan\delta(t)\\
v'(t) &= u_2(t)\\
\delta'(t) &= u_1(t)
\end{aligned}
$$

and

$\delta(t)$	$\in [-30, 30]$	in [deg]
$v(t)$	$\in [0, 6]$	in [m/s]
$u_1(t)$	$\in [-30, 30]$	in [deg/s]
$u_2(t)$	$\in [-1, 1]$	in $[m/s^2]$

+ initial conditions & road boundaries



Notation:

δ	steering angle
v	velocity
ψ	yaw angle
ℓ	distance front to rear axle
(x, y)	reference point

Virtual Testdrive: Mathematical Model

Minimize

$$t_f + \alpha \int_0^{t_f} u_1(t)^2 dt$$

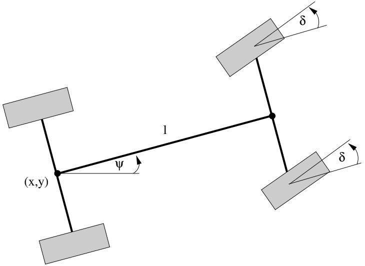
s.t.

$$\begin{aligned} x'(t) &= v(t)\cos\psi(t), \\ y'(t) &= v(t)\sin\psi(t), \\ \psi'(t) &= \frac{v(t)}{\ell}\tan\delta(t) \\ v'(t) &= u_2(t) \\ \delta'(t) &= u_1(t) \end{aligned}$$

and

$\delta(t) \in [-30, 30]$	in [deg]
$v(t) \in [0, 6]$	in [m/s]
$u_1(t) \in [-30, 30]$	in [deg/s]
$u_2(t) \in [-1, 1]$	in $[m/s^2]$

+ initial conditions & road boundaries

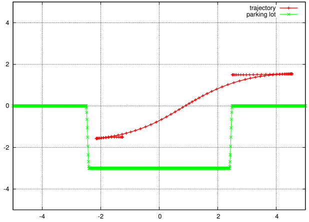


Notation:

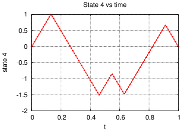
δ	steering angle
v	velocity
ψ	yaw angle
ℓ	distance front to rear axle
(x, y)	reference point

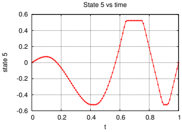
Example: Parking

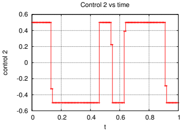
Trajectory (x, y) for optimal parking maneuver:

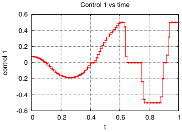


Left to right: velocity v, steering angle δ, acceleration a, steering wheel velocity w:

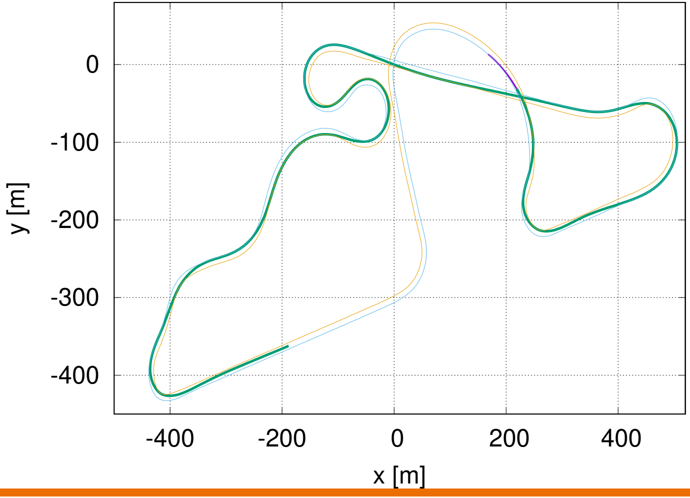








Example: Drive along a Track (Course 3 UniBw M)



Contents

Introduction

Direct Discretization : First Discretize – Then Optimize
- Overview
- Full Discretization (Collocation, Direct Transcription)
- Numerical Solution of Discretized Problem
- Reduced Discretization (Shooting)
- Adjoint Estimation
- Convergence

Applications

Contents

Introduction

Direct Discretization : First Discretize – Then Optimize
- Overview
- Full Discretization (Collocation, Direct Transcription)
- Numerical Solution of Discretized Problem
- Reduced Discretization (Shooting)
- Adjoint Estimation
- Convergence

Applications

Optimal Control Problem

Optimal Control Problem (OCP)

Minimize

$$\varphi(x(t_0), x(t_f))$$

subject to the differential equation

$$x'(t) = f(t, x(t), u(t)) \qquad (t \in [t_0, t_f])$$

the control and state constraints

$$c(t, x(t), u(t)) \leq 0 \qquad (t \in [t_0, t_f])$$

and the boundary conditions

$$\psi(x(t_0), x(t_f)) = 0.$$

Optimal Control Problem

Optimal Control Problem (OCP)

Minimize

$$\varphi(x(t_0), x(t_f))$$

subject to the differential equation

$$x'(t) = f(t, x(t), u(t)) \qquad (t \in [t_0, t_f])$$

the control and state constraints

$$c(t, x(t), u(t)) \leq 0 \qquad (t \in [t_0, t_f])$$

and the boundary conditions

$$\psi(x(t_0), x(t_f)) = 0.$$

Remark:

- w.l.o.g. t_0 and t_f are fixed, $-\infty < t_0 < t_f < \infty$

Various Ways to Approach the Problem

infinite problem “first optimize – then discretize”	vs	discretized problem “first discretize – then optimize”

Various Ways to Approach the Problem

infinite problem “first optimize – then discretize”	vs	discretized problem “first discretize – then optimize”
direct solution (descent methods, SQP, IP, ...)	vs	indirect solution (necessary conditions, semismooth Newton, BVP, ...)

Various Ways to Approach the Problem

infinite problem “first optimize – then discretize”	vs	discretized problem “first discretize – then optimize”
direct solution (descent methods, SQP, IP, ...)	vs	indirect solution (necessary conditions, semismooth Newton, BVP, ...)
full problem large scale & sparse	vs	reduced problem small & dense

Various Ways to Approach the Problem

infinite problem “first optimize – then discretize”	vs	discretized problem “first discretize – then optimize”
direct solution (descent methods, SQP, IP, ...)	vs	indirect solution (necessary conditions, semismooth Newton, BVP, ...)
full problem large scale & sparse	vs	reduced problem small & dense

... in addition: Dynamic Programming/Hamilton-Jacobi theory

Direct Discretization ("First Discretize – Then Optimize") – A Framework

OCP (infinite dimensional)

$$
\begin{array}{ll}
\text{Min} & J(x, u) \\
\text{s.t.} & G(x, u) \leq_K 0 \\
 & H(x, u) = 0
\end{array}
$$

Direct Discretization (“First Discretize – Then Optimize”) – A Framework

OCP (infinite dimensional)

$$\begin{aligned} \text{Min} \quad & J(x, u) \\ \text{s.t.} \quad & G(x, u) \leq_K 0 \\ & H(x, u) = 0 \end{aligned}$$

$\Downarrow$

$\boxed{\text{discretization}}$

$\Downarrow$

Direct Discretization (“First Discretize – Then Optimize”) – A Framework

OCP (infinite dimensional)

$$
\begin{array}{ll}
\text{Min} & J(x,u) \\
\text{s.t.} & G(x,u) \leq_K 0 \\
 & H(x,u) = 0
\end{array}
$$

$\Downarrow$

$\boxed{\text{discretization}}$

$\Downarrow$

state discretization scheme for ODE

$$x'(t) = f(t, x(t), u(t))$$

Direct Discretization ("First Discretize – Then Optimize") – A Framework

OCP (infinite dimensional)

$$\begin{array}{ll} \text{Min} & J(x,u) \\ \text{s.t.} & G(x,u) \leq_K 0 \\ & H(x,u) = 0 \end{array}$$

$\Downarrow$

$\boxed{\text{discretization}}$

$\Downarrow$

state discretization scheme for ODE

$$x'(t) = f(t, x(t), u(t))$$

control parameterization scheme

$$u(t) \approx u_h(t; w)$$

Direct Discretization (“First Discretize – Then Optimize”) – A Framework

OCP (infinite dimensional)

$$\begin{array}{ll} \text{Min} & J(x,u) \\ \text{s.t.} & G(x,u) \preceq_K 0 \\ & H(x,u) = 0 \end{array}$$

$\Downarrow$

$\boxed{\text{discretization}}$

$\Downarrow$

NLP (finite dimensional)

$$\begin{array}{ll} \text{Min} & J_h(x_h,u_h) \\ \text{s.t.} & G_h(x_h,u_h) \leq 0 \\ & H_h(x_h,u_h) = 0 \end{array}$$

state discretization scheme for ODE

$$x'(t) = f(t, x(t), u(t))$$

control parameterization scheme

$$u(t) \approx u_h(t; w)$$

Direct Discretization (“First Discretize – Then Optimize”) – A Framework

OCP (infinite dimensional)

$$\begin{aligned} \text{Min} \quad & J(x, u) \\ \text{s.t.} \quad & G(x, u) \leq_K 0 \\ & H(x, u) = 0 \end{aligned}$$

$\Downarrow$

$\boxed{\text{discretization}}$

$\Downarrow$

NLP (finite dimensional)

$$\begin{aligned} \text{Min} \quad & J_h(x_h, u_h) \\ \text{s.t.} \quad & G_h(x_h, u_h) \leq 0 \\ & H_h(x_h, u_h) = 0 \end{aligned}$$

state discretization scheme for ODE

$$x'(t) = f(t, x(t), u(t))$$

control parameterization scheme

$$u(t) \approx u_h(t; w)$$

NLP solver

Direct Discretization (“First Discretize – Then Optimize”) – A Framework

OCP (infinite dimensional)

$$\begin{aligned} \text{Min} \quad & J(x,u) \\ \text{s.t.} \quad & G(x,u) \leq_K 0 \\ & H(x,u) = 0 \end{aligned}$$

$\Downarrow$

discretization

$\Downarrow$

NLP (finite dimensional)

$$\begin{aligned} \text{Min} \quad & J_h(x_h,u_h) \\ \text{s.t.} \quad & G_h(x_h,u_h) \leq 0 \\ & H_h(x_h,u_h) = 0 \end{aligned}$$

state discretization scheme for ODE

$$x'(t) \quad = \quad f(t, x(t), u(t))$$

control parameterization scheme

$$u(t) \approx u_h(t; w)$$

NLP solver

Many options: collocation, pseudospectral methods, direct shooting methods, semi-smooth Newton, ...

Contents

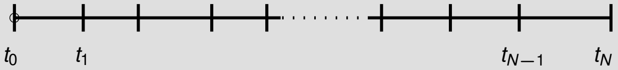
Introduction

Direct Discretization : First Discretize – Then Optimize
- Overview
- Full Discretization (Collocation, Direct Transcription)
- Numerical Solution of Discretized Problem
- Reduced Discretization (Shooting)
- Adjoint Estimation
- Convergence

Applications

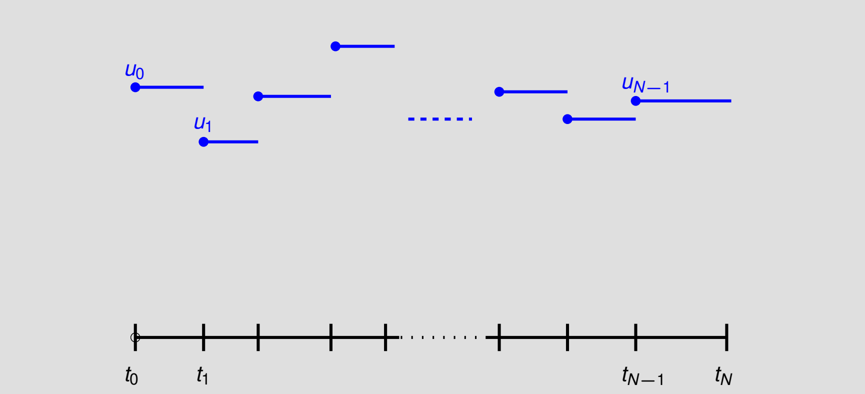
Full Discretization using Euler's Method

- Grid $\mathbb{G}_h = \{t_i \mid t_i = t_0 + ih, i = 0, \ldots, N\}$, $N \in \mathbb{N}$, $h = (t_f - t_0)/N$



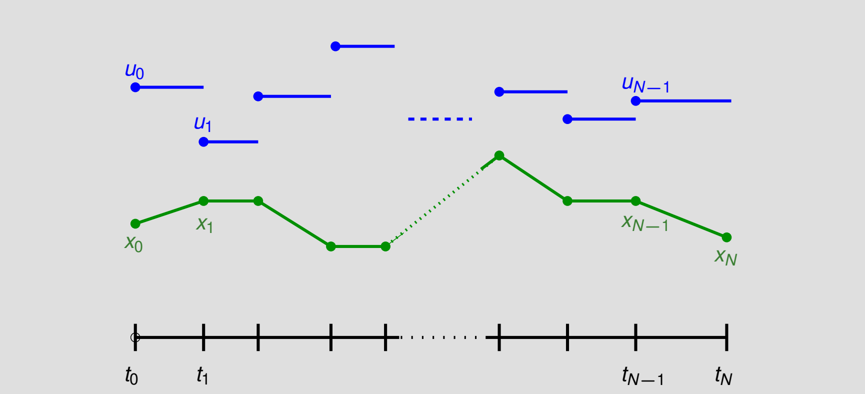
Full Discretization using Euler's Method

- Grid $\mathbb{G}_h = \{t_i \mid t_i = t_0 + ih, i = 0, \ldots, N\}$, $N \in \mathbb{N}$, $h = (t_f - t_0)/N$
- $u_h = (u_0, \ldots, u_{N-1})^\top$: control parameterization on $\mathbb{G}_h$; piecewise constant



Full Discretization using Euler's Method

- Grid $\mathbb{G}_h = \{t_i \mid t_i = t_0 + ih, i = 0, \ldots, N\}$, $N \in \mathbb{N}$, $h = (t_f - t_0)/N$
- $u_h = (u_0, \ldots, u_{N-1})^\top$: control parameterization on $\mathbb{G}_h$; piecewise constant
- $x_h = (x_0, \ldots, x_N)^\top$: state approximation on $\mathbb{G}_h$; explicit Euler method



Full Discretization using Euler's Method

Full Discretization of OCP

Minimize

$$\varphi(x_0, x_N)$$

with respect to (x_h, u_h) subject to

$$\begin{aligned}
\frac{x_{i+1} - x_i}{h} - f(t_i, x_i, u_i) &= 0, \quad i = 0, 1, \dots, N-1, \\
c(t_i, x_i, u_i) &\leq 0, \quad i = 0, 1, \dots, N, \\
\psi(x_0, x_N) &= 0.
\end{aligned}$$

Full Discretization using Euler's Method

Full Discretization of OCP

Minimize

$$\varphi(x_0, x_N)$$

with respect to (x_h, u_h) subject to

$$\begin{aligned} \frac{x_{i+1} - x_i}{h} - f(t_i, x_i, u_i) &= 0, \quad i = 0, 1, \ldots, N-1, \\ c(t_i, x_i, u_i) &\leq 0, \quad i = 0, 1, \ldots, N, \\ \psi(x_0, x_N) &= 0. \end{aligned}$$

Remarks:

- $u_N := u_{N-1}$
- constrained optimization problem, finite dimensional, large-scale, sparse
- u_h can be interpreted as a piecewise constant function in $L_\infty^{n_u}([t_0, t_f])$
- x_h can be interpreted as a continuous, piecewise linear function in $W_{1,\infty}^{n_x}([t_0, t_f])$

Example

Example

Minimize $\alpha x(1) + y(1)$ subject to

$$x'(t) = u(t), \qquad x(0) = 0,$$
$$y'(t) = \frac{1}{2}u(t)^2, \qquad y(0) = 0.$$

Example

Example

Minimize $\alpha x(1) + y(1)$ subject to

$$x'(t) = u(t), \qquad x(0) = 0,$$
$$y'(t) = \frac{1}{2}u(t)^2, \qquad y(0) = 0.$$

Fully discretized problem:

Minimize $\alpha x_N + y_N$ subject to

$$\frac{x_{i+1} - x_i}{h} - u_i = 0, \qquad x_0 = 0 \qquad (i = 0, \ldots, N-1)$$
$$\frac{y_{i+1} - y_i}{h} - \frac{1}{2}u_i^2 = 0, \qquad y_0 = 0 \qquad (i = 0, \ldots, N-1)$$

More General State Discretization

Grid: (equidistant for simplicity)

$$\mathbb{G}_h := \{t_i := t_0 + ih \mid i = 0, 1, \ldots, N\}, \quad h = \frac{t_f - t_0}{N}, \ N \in \mathbb{N}$$

Example (Heun's Method)

$$\left.\begin{aligned} x_{i+1} &= x_i + \frac{h}{2}(k_1 + k_2) \\ k_1 &= f(t_i, x_i, ?) \\ k_2 &= f(t_{i+1}, x_i + hk_1, ?) \end{aligned}\right\} \text{"stage derivatives"}$$

More General State Discretization

Grid: (equidistant for simplicity)

$$\mathbb{G}_h := \{t_i := t_0 + ih \mid i = 0, 1, \ldots, N\}, \quad h = \frac{t_f - t_0}{N}, \ N \in \mathbb{N}$$

Example (Heun's Method)

$$\begin{aligned} x_{i+1} &= x_i + \frac{h}{2}(k_1 + k_2) \\ k_1 &= f(t_i, x_i, u_i) \\ k_2 &= f(t_{i+1}, x_i + hk_1, u_i) \end{aligned} \quad \left.\begin{matrix} \\ \\ \end{matrix}\right\} \text{“stage derivatives”}$$

More General State Discretization

Grid: (equidistant for simplicity)

$$\mathbb{G}_h := \{t_i := t_0 + ih \mid i = 0, 1, \ldots, N\}, \quad h = \frac{t_f - t_0}{N}, \; N \in \mathbb{N}$$

Example (Heun's Method)

$$\begin{aligned} x_{i+1} &= x_i + \frac{h}{2}(k_1 + k_2) \\ k_1 &= f(t_i, x_i, u_i) \\ k_2 &= f(t_{i+1}, x_i + hk_1, u_i) \end{aligned}$$

$$\left.\begin{aligned} k_1 &= f(t_i, x_i, u_i) \\ k_2 &= f(t_{i+1}, x_i + hk_1, u_i) \end{aligned}\right\} \text{“stage derivatives”}$$

Then:

$$\begin{aligned} x_{i+1} &= x_i + h\Phi(t_i, x_i, u_i, h) \\ \Phi(t, x, u, h) &= \frac{1}{2}\left(f(t, x, u) + f(t + h, x + hf(t, x, u), u)\right) \end{aligned}$$

Φ is called increment function

More General State Discretization

Grid: (equidistant for simplicity)

$$\mathbb{G}_h := \{t_i := t_0 + ih \mid i = 0, 1, \ldots, N\}, \quad h = \frac{t_f - t_0}{N}, \; N \in \mathbb{N}$$

Example (Heun's Method)

$$\begin{aligned} x_{i+1} &= x_i + \frac{h}{2}(k_1 + k_2) \\ k_1 &= f(t_i, x_i, u_i) \\ k_2 &= f(t_{i+1}, x_i + hk_1, u_{i+1}) \end{aligned}$$

$\left.\begin{matrix} k_1 \\ k_2 \end{matrix}\right\}$ "stage derivatives"

More General State Discretization

Grid: (equidistant for simplicity)

$$\mathbb{G}_h := \{t_i := t_0 + ih \mid i = 0, 1, \ldots, N\}, \quad h = \frac{t_f - t_0}{N}, \ N \in \mathbb{N}$$

Example (Heun's Method)

$$\begin{aligned} x_{i+1} &= x_i + \frac{h}{2}(k_1 + k_2) \\ k_1 &= f(t_i, x_i, u_i) \\ k_2 &= f(t_{i+1}, x_i + hk_1, u_{i+1}) \end{aligned} \quad \left.\begin{matrix} \\ \\ \end{matrix}\right\} \text{“stage derivatives”}$$

Then:

$$\begin{aligned} x_{i+1} &= x_i + h\Phi(t_i, x_i, u_i, u_{i+1}, h) \\ \Phi(t, x, u_i, u_{i+1}, h) &= \frac{1}{2}\left(f(t, x, u_i) + f(t + h, x + hf(t, x, u_i), u_{i+1})\right) \end{aligned}$$

Φ is called increment function

State Discretization

Example (Modified Euler Method)

$$\begin{aligned} x_{i+1} &= x_i + hk_2 \\ k_1 &= f(t_i, x_i, \ ?) \\ k_2 &= f(t_i + \frac{h}{2}, x_i + \frac{h}{2}k_1, \ ?) \end{aligned}$$

State Discretization

Example (Modified Euler Method)

$$
\begin{aligned}
x_{i+1} &= x_i + hk_2 \\
k_1 &= f(t_i, x_i, u_i) \\
k_2 &= f\left(t_i + \frac{h}{2}, x_i + \frac{h}{2}k_1, u_i\right)
\end{aligned}
$$

State Discretization

Example (Modified Euler Method)

$$
\begin{aligned}
x_{i+1} &= x_i + hk_2 \\
k_1 &= f(t_i, x_i, u_i) \\
k_2 &= f(t_i + \frac{h}{2}, x_i + \frac{h}{2}k_1, u_i)
\end{aligned}
$$

Then:

$$
\begin{aligned}
x_{i+1} &= x_i + h\Phi(t_i, x_i, u_i, h) \\
\Phi(t, x, u, h) &= f(t + \frac{h}{2}, x + \frac{h}{2}f(t, x, u), u)
\end{aligned}
$$

State Discretization

Example (Modified Euler Method)

$$
\begin{aligned}
x_{i+1} &= x_i + hk_2 \\
k_1 &= f(t_i, x_i, u_i) \\
k_2 &= f\left(t_i + \frac{h}{2}, x_i + \frac{h}{2}k_1, \frac{u_i + u_{i+1}}{2}\right)
\end{aligned}
$$

State Discretization

Example (Modified Euler Method)

$$
\begin{aligned}
x_{i+1} &= x_i + hk_2 \\
k_1 &= f(t_i, x_i, u_i) \\
k_2 &= f(t_i + \frac{h}{2}, x_i + \frac{h}{2}k_1, \frac{u_i + u_{i+1}}{2})
\end{aligned}
$$

Then:

$$
\begin{aligned}
x_{i+1} &= x_i + h\Phi(t_i, x_i, u_i, u_{i+1}, h) \\
\Phi(t, x, u_i, u_{i+1}, h) &= f(t + \frac{h}{2}, x + \frac{h}{2}f(t, x, u_i), \frac{u_i + u_{i+1}}{2})
\end{aligned}
$$

State Discretization

Example (Modified Euler Method)

$$
\begin{aligned}
x_{i+1} &= x_i + hk_2 \\
k_1 &= f(t_i, x_i, u_i) \\
k_2 &= f\left(t_i + \frac{h}{2}, x_i + \frac{h}{2}k_1, u_{i+1/2}\right)
\end{aligned}
$$

State Discretization

Example (Modified Euler Method)

$$
\begin{aligned}
x_{i+1} &= x_i + hk_2 \\
k_1 &= f(t_i, x_i, u_i) \\
k_2 &= f(t_i + \frac{h}{2}, x_i + \frac{h}{2} k_1, u_{i+1/2})
\end{aligned}
$$

Then:

$$
\begin{aligned}
x_{i+1} &= x_i + h\Phi(t_i, x_i, u_i, u_{i+1/2}, h) \\
\Phi(t, x, u_i, u_{i+1/2}, h) &= f(t + \frac{h}{2}, x + \frac{h}{2} f(t, x, u_i), u_{i+1/2})
\end{aligned}
$$

State Discretization by Collocation

Collocation Idea

Approximate the solution x of the initial value problem

$$x'(t) = f(t, x(t), u(t)), \qquad x(t_i) = x_i,$$

in $[t_i, t_{i+1}]$ by a polynomial $p : [t_i, t_{i+1}] \to \mathbb{R}^{n_x}$ of degree s.

Construction:

State Discretization by Collocation

Collocation Idea

Approximate the solution x of the initial value problem

$$x'(t) = f(t, x(t), u(t)), \qquad x(t_i) = x_i,$$

in $[t_i, t_{i+1}]$ by a polynomial $p : [t_i, t_{i+1}] \to \mathbb{R}^{n_x}$ of degree s.

Construction:

- collocation points $t_i \leq \tau_1 < \tau_2 < \cdots \tau_s \leq t_{i+1}$

State Discretization by Collocation

Collocation Idea

Approximate the solution x of the initial value problem

$$x'(t) = f(t, x(t), u(t)), \qquad x(t_i) = x_i,$$

in $[t_i, t_{i+1}]$ by a polynomial $p : [t_i, t_{i+1}] \to \mathbb{R}^{n_x}$ of degree s.

Construction:

- collocation points $t_i \le \tau_1 < \tau_2 < \cdots \tau_s \le t_{i+1}$
- collocation conditions:

$$p(t_i) = x_i, \quad p'(\tau_k) = f(\tau_k, p(\tau_k), u(\tau_k)), \quad k = 1, \ldots, s$$

State Discretization by Collocation

Collocation Idea

Approximate the solution x of the initial value problem

$$x'(t) = f(t, x(t), u(t)), \qquad x(t_i) = x_i,$$

in $[t_i, t_{i+1}]$ by a polynomial $p : [t_i, t_{i+1}] \to \mathbb{R}^{n_x}$ of degree s.

Construction:

- collocation points $t_i \leq \tau_1 < \tau_2 < \cdots \tau_s \leq t_{i+1}$
- collocation conditions:

$$p(t_i) = x_i, \quad p'(\tau_k) = f(\tau_k, p(\tau_k), u(\tau_k)), \quad k = 1, \ldots, s$$

- define $x_{i+1} := p(t_{i+1})$

State Discretization by Collocation

Collocation Idea

Approximate the solution x of the initial value problem

$$x'(t) = f(t, x(t), u(t)), \qquad x(t_i) = x_i,$$

in $[t_i, t_{i+1}]$ by a polynomial $p : [t_i, t_{i+1}] \to \mathbb{R}^{n_x}$ of degree s.

Construction:

- collocation points $t_i \leq \tau_1 < \tau_2 < \cdots \tau_s \leq t_{i+1}$
- collocation conditions:

$$p(t_i) = x_i, \quad p'(\tau_k) = f(\tau_k, p(\tau_k), u(\tau_k)), \quad k = 1, \ldots, s$$

- define $x_{i+1} := p(t_{i+1})$

In general: Every collocation method corresponds to an implicit Runge-Kutta method.

State Discretization by Collocation

Example (Implicit Trapezoidal Rule)

For $s = 2$, $\tau_1 = t_i$, $\tau_2 = t_{i+1}$, the collocation idea yields

$$x_{i+1} = x_i + \frac{h}{2}\left(f(t_i, x_i, u_i) + f(t_{i+1}, x_{i+1}, u_{i+1})\right).$$

This is the implicit trapezoidal rule!

Full Hermite-Simpson Discretization (Collocation)

Example (Hermite-Simpson, Collocation)

The Hermite-Simpson rule reads

$$x_{i+1} = x_i + \frac{h}{6}\left(f_i + 4f_{i+\frac{1}{2}} + f_{i+1}\right), \quad i = 0, 1, \ldots, N-1,$$

where

$$\begin{aligned}
f_i &:= f(t_i, x_i, u_i), \quad f_{i+1} := f(t_{i+1}, x_{i+1}, u_{i+1}), \\
f_{i+\frac{1}{2}} &:= f(t_i + \frac{h}{2}, x_{i+\frac{1}{2}}, u_{i+\frac{1}{2}}), \\
x_{i+\frac{1}{2}} &:= \frac{1}{2}(x_i + x_{i+1}) + \frac{h}{8}(f_i - f_{i+1}).
\end{aligned}$$

Full Hermite-Simpson Discretization (Collocation)

Example (Hermite-Simpson, Collocation)

The Hermite-Simpson rule reads

$$x_{i+1} = x_i + \frac{h}{6}\left(f_i + 4f_{i+\frac{1}{2}} + f_{i+1}\right), \quad i = 0, 1, \ldots, N-1,$$

where

$$\begin{aligned} f_i &:= f(t_i, x_i, u_i), \quad f_{i+1} := f(t_{i+1}, x_{i+1}, u_{i+1}), \\ f_{i+\frac{1}{2}} &:= f(t_i + \frac{h}{2}, x_{i+\frac{1}{2}}, u_{i+\frac{1}{2}}), \\ x_{i+\frac{1}{2}} &:= \frac{1}{2}(x_i + x_{i+1}) + \frac{h}{8}(f_i - f_{i+1}). \end{aligned}$$

Herein, we need to specify what $u_{i+\frac{1}{2}}$ is supposed to be. Several choices are possible, e.g.

Full Hermite-Simpson Discretization (Collocation)

Example (Hermite-Simpson, Collocation)

The Hermite-Simpson rule reads

$$x_{i+1} = x_i + \frac{h}{6}\left(f_i + 4f_{i+\frac{1}{2}} + f_{i+1}\right), \quad i = 0, 1, \ldots, N-1,$$

where

$$\begin{aligned} f_i &:= f(t_i, x_i, u_i), \quad f_{i+1} := f(t_{i+1}, x_{i+1}, u_{i+1}), \\ f_{i+\frac{1}{2}} &:= f(t_i + \frac{h}{2}, x_{i+\frac{1}{2}}, u_{i+\frac{1}{2}}), \\ x_{i+\frac{1}{2}} &:= \frac{1}{2}(x_i + x_{i+1}) + \frac{h}{8}(f_i - f_{i+1}). \end{aligned}$$

Herein, we need to specify what $u_{i+\frac{1}{2}}$ is supposed to be. Several choices are possible, e.g.

- if a continuous and piecewise linear control approximation is chosen, then $u_{i+\frac{1}{2}} = \frac{1}{2}(u_i + u_{i+1})$;

Full Hermite-Simpson Discretization (Collocation)

Example (Hermite-Simpson, Collocation)

The Hermite-Simpson rule reads

$$x_{i+1} = x_i + \frac{h}{6}\left(f_i + 4f_{i+\frac{1}{2}} + f_{i+1}\right), \quad i = 0, 1, \ldots, N-1,$$

where

$$\begin{aligned} f_i &:= f(t_i, x_i, u_i), \quad f_{i+1} := f(t_{i+1}, x_{i+1}, u_{i+1}), \\ f_{i+\frac{1}{2}} &:= f(t_i + \frac{h}{2}, x_{i+\frac{1}{2}}, u_{i+\frac{1}{2}}), \\ x_{i+\frac{1}{2}} &:= \frac{1}{2}(x_i + x_{i+1}) + \frac{h}{8}(f_i - f_{i+1}). \end{aligned}$$

Herein, we need to specify what $u_{i+\frac{1}{2}}$ is supposed to be. Several choices are possible, e.g.

- if a continuous and piecewise linear control approximation is chosen, then $u_{i+\frac{1}{2}} = \frac{1}{2}(u_i + u_{i+1})$;
- $u_{i+\frac{1}{2}}$ can be introduced as an additional optimization variable without specifying any relations to u_i and u_{i+1}.

Full Discretization of OCP

Implementation of Constraints

Full Discretization of OCP

Implementation of Constraints

- Version A: (condensed)
  Stage equations $k_j = \ldots$ are not explicitly added as equality constraints in D-OCP.

  Example: Heun

$$x_i + \frac{h}{2}\left(f(t_i, x_i, u_i) + f(t_{i+1}, x_i + hf(t_i, x_i, u_i), u_i)\right) - x_{i+1} = 0$$

Full Discretization of OCP

Implementation of Constraints

- Version A: (condensed)
  Stage equations $k_j = \ldots$ are not explicitly added as equality constraints in D-OCP.

  Example: Heun

$$x_i + \frac{h}{2}\left(f(t_i, x_i, u_i) + f(t_{i+1}, x_i + hf(t_i, x_i, u_i), u_i)\right) - x_{i+1} = 0$$

- Version B: (stage formulation)
  Stage equations $k_j = \ldots$ are explicitly added as equality constraints in D-OCP.

  Example: Heun

$$\begin{aligned} k_1 - f(t_i, x_i, u_i) &= 0 \\ k_2 - f(t_{i+1}, x_i + hk_1, u_i) &= 0 \\ x_i + \frac{h}{2}(k_1 + k_2) - x_{i+1} &= 0 \end{aligned}$$

  This version is typically implemented for implicit Runge-Kutta methods.

A Unified Full Discretization Approach

Summary:

A Unified Full Discretization Approach

Summary:

- control parameterization has an influence on the increment function Φ

A Unified Full Discretization Approach

Summary:

- control parameterization has an influence on the increment function Φ
- state discretization scheme can be formulated in different ways (condensed or with stage equations)

A Unified Full Discretization Approach

Summary:

- control parameterization has an influence on the increment function Φ
- state discretization scheme can be formulated in different ways (condensed or with stage equations)
- different nonlinear optimization problems arise (degree of sparsity differs)

A Unified Full Discretization Approach

Summary:

- control parameterization has an influence on the increment function Φ
- state discretization scheme can be formulated in different ways (condensed or with stage equations)
- different nonlinear optimization problems arise (degree of sparsity differs)

Unification:

A Unified Full Discretization Approach

Summary:

- control parameterization has an influence on the increment function Φ
- state discretization scheme can be formulated in different ways (condensed or with stage equations)
- different nonlinear optimization problems arise (degree of sparsity differs)

Unification:

- control parameterization using B-splines

A Unified Full Discretization Approach

Summary:

- control parameterization has an influence on the increment function Φ
- state discretization scheme can be formulated in different ways (condensed or with stage equations)
- different nonlinear optimization problems arise (degree of sparsity differs)

Unification:

- control parameterization using B-splines
- one-step method with increment function Φ

Control Parameterization

Grid: (equidistant for simplicity)

$$\mathbb{G}_h := \{t_0 + ih \mid i = 0, 1, \ldots, N\}, \quad h = \frac{t_f - t_0}{N}, \; N \in \mathbb{N}$$

Control Parameterization

Grid: (equidistant for simplicity)

$$\mathbb{G}_h := \{t_0 + ih \mid i = 0, 1, \ldots, N\}, \quad h = \frac{t_f - t_0}{N}, \; N \in \mathbb{N}$$

Control parameterization via B-Splines

B-spline approximations of order $k \in \mathbb{N}$:

$$u_h(t; w) := \sum_{i=1}^{N+k-1} w_i \, B_i^k(t)$$

Control Parameterization

Grid: (equidistant for simplicity)

$$\mathbb{G}_h := \{t_0 + ih \mid i = 0, 1, \ldots, N\}, \quad h = \frac{t_f - t_0}{N}, \; N \in \mathbb{N}$$

Control parameterization via B-Splines

B-spline approximations of order $k \in \mathbb{N}$:

$$u_h(t; w) := \sum_{i=1}^{N+k-1} w_i \, B_i^k(t)$$

Notation:

- control parameterization: $w = (w_1, \ldots, w_M)^\top \in \mathbb{R}^{Mn_u}$, $M := N + k - 1$
- basis functions B_i^k (elementary B-splines of order k)

Control Parameterization

Example (Elementary B-splines B_i^k)

Elementary B-splines of order $k = 1$ are piecewise constant basis functions on $\mathbb{G}_h$.

Control Parameterization

Example (Elementary B-splines B_i^k)

Elementary B-splines of order $k = 1$ are piecewise constant basis functions on $\mathbb{G}_h$.

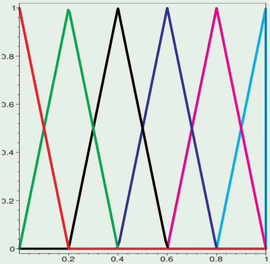
Elementary B-splines of orders $k = 2, 3, 4$: ($[t_0, t_f] = [0, 1]$, $N = 5$, equidistant grid)

Control Parameterization

Example (Elementary B-splines B_i^k)

Elementary B-splines of order $k = 1$ are piecewise constant basis functions on $\mathbb{G}_h$.

Elementary B-splines of orders $k = 2, 3, 4$: ($[t_0, t_f] = [0, 1]$, $N = 5$, equidistant grid)

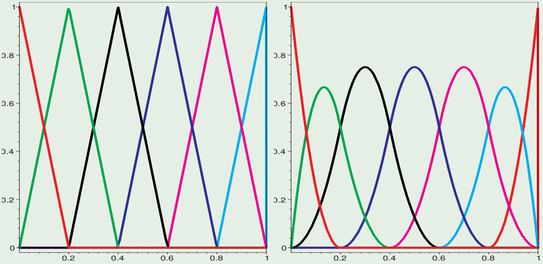


Control Parameterization

Example (Elementary B-splines B_i^k)

Elementary B-splines of order $k = 1$ are piecewise constant basis functions on $\mathbb{G}_h$.

Elementary B-splines of orders $k = 2, 3, 4$: ($[t_0, t_f] = [0, 1]$, $N = 5$, equidistant grid)

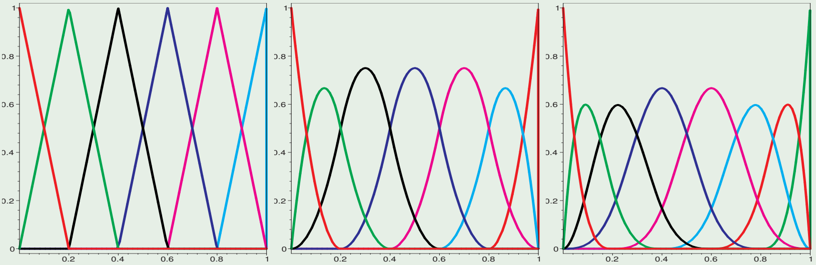


Control Parameterization

Example (Elementary B-splines B_i^k)

Elementary B-splines of order $k = 1$ are piecewise constant basis functions on $\mathbb{G}_h$.

Elementary B-splines of orders $k = 2, 3, 4$: ($[t_0, t_f] = [0, 1]$, $N = 5$, equidistant grid)

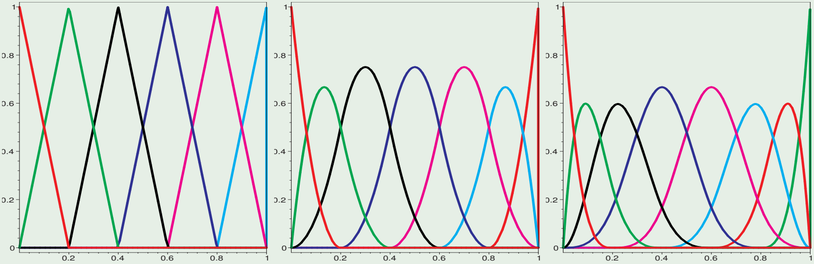


Control Parameterization

Example (Elementary B-splines B_i^k)

Elementary B-splines of order $k = 1$ are piecewise constant basis functions on $\mathbb{G}_h$.

Elementary B-splines of orders $k = 2, 3, 4$: ($[t_0, t_f] = [0, 1]$, $N = 5$, equidistant grid)



Properties:

- B_i^k has local support (depends on order k)
- smoothness of B_i^k can be adjusted with order k

Alternative Parameterizations

Special cases:

- piecewise constant control approximation ($k = 1$):

$$w = (u_0, u_1, \dots, u_{N-1})^\top, \qquad u_i = u_h(t_i; w) \quad (i = 0, \dots, N-1)$$

Alternative Parameterizations

Special cases:

- piecewise constant control approximation ($k = 1$):

$$w = (u_0, u_1, \dots, u_{N-1})^\top, \qquad u_i = u_h(t_i; w) \quad (i = 0, \dots, N-1)$$

- continuous and piecewise linear control approximation ($k = 2$):

$$w = (u_0, u_1, \dots, u_N)^\top, \qquad u_i = u_h(t_i; w) \quad (i = 0, \dots, N)$$

Alternative Parameterizations

Special cases:

- piecewise constant control approximation ($k = 1$):

$$w = (u_0, u_1, \ldots, u_{N-1})^\top, \qquad u_i = u_h(t_i; w) \quad (i = 0, \ldots, N-1)$$

- continuous and piecewise linear control approximation ($k = 2$):

$$w = (u_0, u_1, \ldots, u_N)^\top, \qquad u_i = u_h(t_i; w) \quad (i = 0, \ldots, N)$$

Alternatives to B-splines:

- interpolating cubic spline (non-local support, twice continuously differentiable)
- polynomials (non-local support, smooth) $\rightsquigarrow$ pseudospectral methods

State Discretization by One-Step Method

Given: Control parameterization

$$u_h(\cdot; w) \qquad (w = \text{control parameters; B-spline representation})$$

State Discretization by One-Step Method

Given: Control parameterization

$$u_h(\cdot; w) \qquad (w = \text{control parameters; B-spline representation})$$

State discretization

For a given control parameterization $u_h(\cdot; w)$ approximations $x_i \approx x(t_i)$, $t_i \in \mathbb{G}_h$ are obtained by a one-step method

$$x_{i+1} = x_i + h\Phi(t_i, x_i, w, h), \quad i = 0, 1, \ldots, N-1.$$

This yields a state approximation

$$x_h = (x_0, x_1, \ldots, x_N)^\top \in \mathbb{R}^{(N+1)n_x}$$

Full Discretization of OCP

Discretization of the optimal control problem with Mayer-type objective function and B-spline control parameterization of order k yields:

Fully discretized optimal control problem (D-OCP)

Find $x_h = (x_0, \ldots, x_N)^\top \in \mathbb{R}^{(N+1)n_x}$ and $w = (w_1, \ldots, w_M)^\top \in \mathbb{R}^{Mn_u}$ such that

$$\varphi(x_0, x_N)$$

becomes minimal subject to the discretized dynamic constraints

$$x_i + h\Phi(t_i, x_i, w, h) - x_{i+1} = 0, \quad i = 0, 1, \ldots, N-1,$$

the discretized control and state constraints

$$c(t_i, x_i, u_h(t_i; w)) \leq 0, \quad i = 0, 1, \ldots, N,$$

and the boundary conditions

$$\psi(x_0, x_N) = 0.$$

Full Discretization of OCP

The fully discretized optimal control problem is a standard nonlinear optimization problem, which is large-scale but exhibits a sparse structure:

Nonlinear Optimization Problem (NLP)

Minimize

$$J_h(z) := \varphi(x_0, x_N)$$

w.r.t. $z = (x_h, w)^\top$ subject to the constraints

$$H_h(z) = 0, \qquad G_h(z) \leq 0,$$

where

$$H_h(z) := \begin{pmatrix} x_0 + h\Phi(t_0, x_0, w, h) - x_1 \\ \vdots \\ x_{N-1} + h\Phi(t_{N-1}, x_{N-1}, w, h) - x_N \\ \psi(x_0, x_N) \end{pmatrix}, \quad G_h(z) := \begin{pmatrix} c(t_0, x_0, u_h(t_0; w)) \\ \vdots \\ c(t_N, x_N, u_h(t_N; w)) \end{pmatrix}.$$

Sparsity Structures in D-OCP

$$J_h'(z) \quad = \quad \left(\begin{array}{cc|c} \varphi'_{x_0} & \varphi'_{x_N} & 0 \end{array} \right),$$

Sparsity Structures in D-OCP

$$J_h'(z) = \left(\begin{array}{cc|c} \varphi'_{x_0} & \varphi'_{x_N} & 0 \end{array} \right),$$

$$H_h'(z) = \left(\begin{array}{ccccc|c} M_0 & -Id & & & & h\Phi'_w[t_0] \\ & \ddots & \ddots & & & \vdots \\ & & & M_{N-1} & -Id & h\Phi'_w[t_{N-1}] \\ \hline \psi'_{x_0} & & & & \psi'_{x_N} & 0 \end{array} \right), \quad M_j := Id + h\Phi'_x[t_j]$$

Sparsity Structures in D-OCP

$$J_h'(z) = \left(\begin{array}{cc|c} \varphi'_{x_0} & \varphi'_{x_N} & 0 \end{array} \right),$$

$$H_h'(z) = \left(\begin{array}{cccc|c} M_0 & -Id & & & h\Phi'_w[t_0] \\ & \ddots & \ddots & & \vdots \\ & & M_{N-1} & -Id & h\Phi'_w[t_{N-1}] \\ \hline \psi'_{x_0} & & & \psi'_{x_N} & 0 \end{array} \right), \quad M_j := Id + h\Phi'_x[t_j]$$

$$G_h'(z) = \left(\begin{array}{ccc|c} c'_x[t_0] & & & c'_u[t_0]u'_{h,w}(t_0;w) \\ & \ddots & & \vdots \\ & & c'_x[t_N] & c'_u[t_N]u'_{h,w}(t_N;w) \end{array} \right),$$

Sparsity Structures in D-OCP

$$J_h'(z) = \left(\begin{array}{cc|c} \varphi'_{x_0} & \varphi'_{x_N} & 0 \end{array} \right),$$

$$H_h'(z) = \left(\begin{array}{cccc|c} M_0 & -Id & & & h\Phi'_w[t_0] \\ & \ddots & \ddots & & \vdots \\ & & M_{N-1} & -Id & h\Phi'_w[t_{N-1}] \\ \hline \psi'_{x_0} & & & \psi'_{x_N} & 0 \end{array} \right), \quad M_j := Id + h\Phi'_x[t_j]$$

$$G_h'(z) = \left(\begin{array}{ccc|c} c'_x[t_0] & & & c'_u[t_0]u'_{h,w}(t_0; w) \\ & \ddots & & \vdots \\ & & c'_x[t_N] & c'_u[t_N]u'_{h,w}(t_N; w) \end{array} \right),$$

$$u'_{h,w}(t_j; w) = \left(\begin{array}{c|c|c|c} B_1^k(t_j) \cdot Id & B_2^k(t_j) \cdot Id & \cdots & B_M^k(t_j) \cdot Id \end{array} \right)$$

Sparsity Structures in D-OCP

$$J_h'(z) = \left(\begin{array}{cc|c} \varphi_{x_0}' & \varphi_{x_N}' & 0 \end{array} \right),$$

$$H_h'(z) = \left(\begin{array}{cccc|c} M_0 & -Id & & & h\Phi_w'[t_0] \\ & \ddots & \ddots & & \vdots \\ & & M_{N-1} & -Id & h\Phi_w'[t_{N-1}] \\ \hline \psi_{x_0}' & & & \psi_{x_N}' & 0 \end{array} \right), \quad M_j := Id + h\Phi_x'[t_j]$$

$$G_h'(z) = \left(\begin{array}{ccc|c} c_x'[t_0] & & & c_u'[t_0]u_{h,w}'(t_0; w) \\ & \ddots & & \vdots \\ & & c_x'[t_N] & c_u'[t_N]u_{h,w}'(t_N; w) \end{array} \right),$$

$$u_{h,w}'(t_j; w) = \left(\begin{array}{c|c|c|c} B_1^k(t_j) \cdot Id & B_2^k(t_j) \cdot Id & \cdots & B_M^k(t_j) \cdot Id \end{array} \right)$$

Observation:

- B_i^k have local support $\Longrightarrow$ most entries in $u_{h,w}'(t_j; w)$ and in $\Phi_w'[t_j]$ vanish
- $H_h'(z)$ and $G_h'(z)$ have a large-scale and sparse structure.

Sparsity Structures in the Full Discretization Approach

Lagrange function:

$$
\begin{aligned}
L_h(z,\boldsymbol{\lambda},\boldsymbol{\mu},\boldsymbol{\sigma}) \quad = \quad & \varphi(x_0,x_N) + \boldsymbol{\sigma}^\top \psi(x_0,x_N) \\
& + \sum_{i=0}^{N-1} \boldsymbol{\lambda}_{i+1}^\top (x_i + h\Phi(t_i,x_i,w,h) - x_{i+1}) + \sum_{i=0}^{N} \boldsymbol{\mu}_i^\top c(t_i,x_i,u_h(t_i;w))
\end{aligned}
$$

Sparsity Structures in the Full Discretization Approach

Lagrange function:

$$
\begin{aligned}
L_h(z,\boldsymbol{\lambda},\boldsymbol{\mu},\boldsymbol{\sigma}) &= \varphi(x_0,x_N) + \boldsymbol{\sigma}^\top \psi(x_0,x_N) \\
&\quad + \sum_{i=0}^{N-1} \boldsymbol{\lambda}_{i+1}^\top (x_i + h\Phi(t_i,x_i,w,h) - x_{i+1}) + \sum_{i=0}^{N} \boldsymbol{\mu}_i^\top c(t_i,x_i,u_h(t_i;w))
\end{aligned}
$$

Hessian matrix:

$$
\nabla_{zz}^2 L_h(z,\boldsymbol{\lambda},\boldsymbol{\mu},\boldsymbol{\sigma}) = \left(\begin{array}{ccc|c}
L''_{x_0,x_0} & & L''_{x_0,x_N} & L''_{x_0,w} \\
 & \ddots & & \vdots \\
L''_{x_N,x_0} & & L''_{x_N,x_N} & L''_{x_N,w} \\
\hline
L''_{w,x_0} & \cdots & L''_{w,x_N} & L''_{w,w}
\end{array}\right)
$$

Sparsity Structures in the Full Discretization Approach

Lagrange function:

$$
\begin{aligned}
L_h(z,\boldsymbol{\lambda},\boldsymbol{\mu},\boldsymbol{\sigma}) &= \varphi(x_0,x_N) + \boldsymbol{\sigma}^\top \psi(x_0,x_N) \\
&\quad + \sum_{i=0}^{N-1} \boldsymbol{\lambda}_{i+1}^\top (x_i + h\Phi(t_i,x_i,w,h) - x_{i+1}) + \sum_{i=0}^{N} \boldsymbol{\mu}_i^\top c(t_i,x_i,u_h(t_i;w))
\end{aligned}
$$

Hessian matrix:

$$
\nabla_{zz}^2 L_h(z,\boldsymbol{\lambda},\boldsymbol{\mu},\boldsymbol{\sigma}) = \left(\begin{array}{ccc|c}
L''_{x_0,x_0} & & L''_{x_0,x_N} & L''_{x_0,w} \\
 & \ddots & & \vdots \\
L''_{x_N,x_0} & & L''_{x_N,x_N} & L''_{x_N,w} \\
\hline
L''_{w,x_0} & \cdots & L''_{w,x_N} & L''_{w,w}
\end{array}\right)
$$

Note: The blocks $L''_{x_j,w} = (L''_{w,x_j})^\top$ and $L''_{w,w}$ are sparse matrices if B-splines are used.

Some Caution is Necessary

Caution!

See what happens if you apply the modified Euler method with additional optimization variables $u_{k+\frac{1}{2}}$ at the midpoints $t_k + \frac{h}{2}$ to the following problem:

Minimize

$$\frac{1}{2}\int_0^1 u(t)^2 + 2x(t)^2 dt$$

subject to the constraints

$$x'(t) = \frac{1}{2}x(t) + u(t), \quad x(0) = 1.$$

The optimal solution is

$$\hat{x}(t) = \frac{2\exp(3t) + \exp(3)}{\exp(3t/2)(2 + \exp(3))}, \quad \hat{u}(t) = \frac{2(\exp(3t) - \exp(3))}{\exp(3t/2)(2 + \exp(3))}.$$

Grid Refinement

Approaches:

- Refinement based on the local discretization error of the state dynamics and local refinement at junction points of active/inactive state constraints, see [1, 2]
- Refinement taking into account the discretization error in the optimality system including adjoint equations, see [3]

[1] Betts, J. T. and Huffman, W. P.
Mesh Refinement in Direct Transcription Methods for Optimal Control.
Optimal Control Applications and Methods, 19; 1–21, 1998.

[2] C. Büskens.
Optimierungsmethoden und Sensitivitätsanalyse für optimale Steuerprozesse mit Steuer- und Zustandsbeschränkungen.
PhD thesis, Fachbereich Mathematik, Westfälische Wilhems-Universität Münster, Münster, Germany, 1998.

[3] J. Laurent-Varin, F. Bonnans, N. Berend, C. Talbot, and M. Haddou.
On the refinement of discretization for optimal control problems.
IFAC Symposium on Automatic Control in Aerospace, St. Petersburg, 2004.

Pseudospectral Methods

Approach:

- global approximation of control and state by Legendre or Chebyshev polynomials
- direct discretization
- sparse nonlinear programming solver

Advantages:

- exponential (or spectral) rate of convergence (faster than polynomial)
- good accuracy already with coarse grids

Disadvantages:

- oscillations for non-differentiable trajectories

[1] G. Elnagar, M. A. Kazemi, and M. Razzaghi.
The Pseudospectral Legendre Method for Discretizing Optimal Control Problems.
IEEE Transactions on Automatic Control, 40:1793–1796, 1995.

[2] J. Vlassenbroeck and R. Van Doreen.
A Chebyshev Technique for Solving Nonlinear Optimal Control Problems.
IEEE Transactions on Automatic Control, 33:333–340, 1988.

[3] F. Fahroo and I.M. Ross.
Direct Trajectory Optimization by a Chebyshev Pseudospectral Method.
Journal of Guidance Control and Dynamics, 25, 2002.

Contents

Introduction

Direct Discretization : First Discretize – Then Optimize
- Overview
- Full Discretization (Collocation, Direct Transcription)
- Numerical Solution of Discretized Problem
- Reduced Discretization (Shooting)
- Adjoint Estimation
- Convergence

Applications

Numerical Solution of D-OCP

It remains to solve D-OCP.

NLP

$$
\begin{aligned}
\text{Minimize} \quad & J_h(x_h, w) \\
\text{w.r.t.} \quad & z = (x_h, w) \\
\text{subject to} \quad & G_h(x_h, w) \leq 0 \\
& H_h(x_h, w) = 0
\end{aligned}
$$

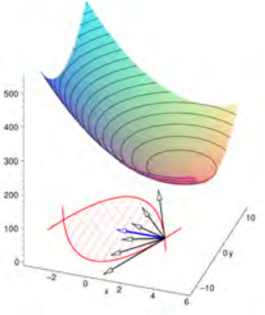
Properties: (depend on discretization)

- high degree of nonlinearity
- large-scale and sparse Jacobians and Hessian
- particular block-sparse structure

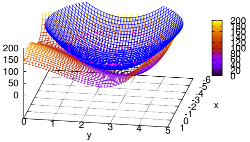
Solving Nonlinear Optimization Problems

Optimization frameworks: (many options!)

- nonlinear problems: sequential-quadratic programming (SQP), interior-point methods (IP), penalty or multiplier-penalty methods, semi-smooth Newton methods, ...
- linear quadratic problems: active-set methods, IP, semi-smooth Newton, ADMM, OSQP, ...



Himmelblau function and quadratic approximation at (-2,2)



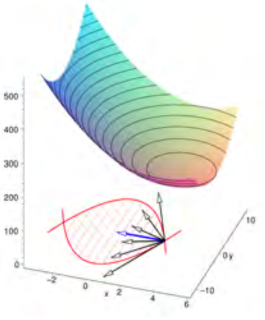
Solving Nonlinear Optimization Problems

Optimization frameworks: (many options!)

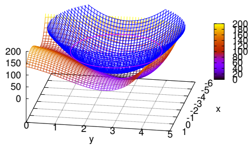
- nonlinear problems: sequential-quadratic programming (SQP), interior-point methods (IP), penalty or multiplier-penalty methods, semi-smooth Newton methods, ...
- linear quadratic problems: active-set methods, IP, semi-smooth Newton, ADMM, OSQP, ...

Globalization: (convergence from arbitrary points)

- line-search / trust-region methods / filter methods



Himmelblau function and quadratic approximation at (-2,2)



Solving Nonlinear Optimization Problems

Optimization frameworks: (many options!)

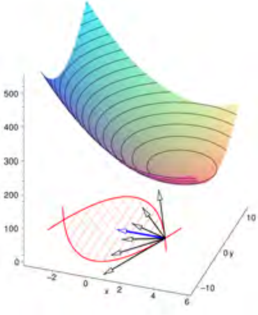
- nonlinear problems: sequential-quadratic programming (SQP), interior-point methods (IP), penalty or multiplier-penalty methods, semi-smooth Newton methods, ...
- linear quadratic problems: active-set methods, IP, semi-smooth Newton, ADMM, OSQP, ...

Globalization: (convergence from arbitrary points)

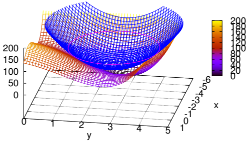
- line-search / trust-region methods / filter methods

Peripheral problems:

- sparsity / large scales / rank deficiencies / regularization



Himmelblau function and quadratic approximation at (-2,2)



Solving Nonlinear Optimization Problems

Optimization frameworks: (many options!)

- nonlinear problems: sequential-quadratic programming (SQP), interior-point methods (IP), penalty or multiplier-penalty methods, semi-smooth Newton methods, ...
- linear quadratic problems: active-set methods, IP, semi-smooth Newton, ADMM, OSQP, ...

Globalization: (convergence from arbitrary points)

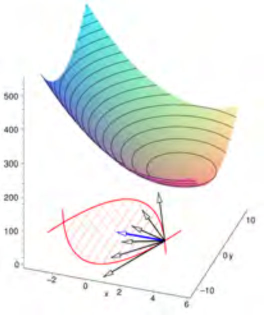
- line-search / trust-region methods / filter methods

Peripheral problems:

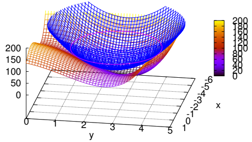
- sparsity / large scales / rank deficiencies / regularization

Comprehensive textbook:

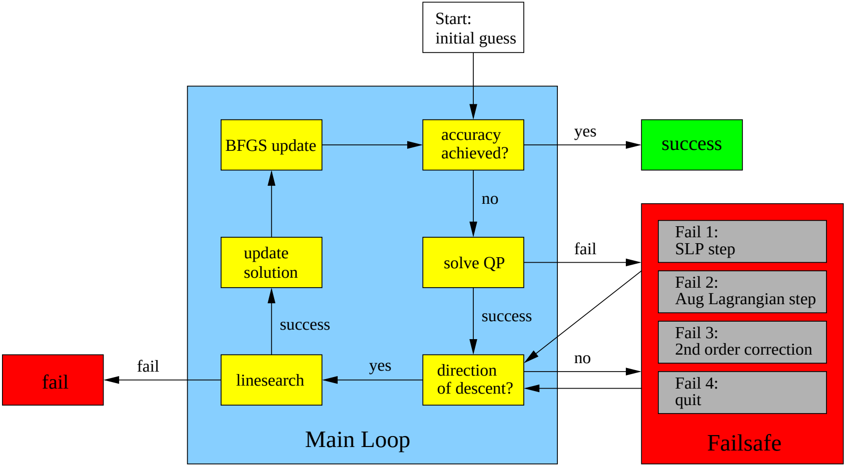
[1] J. Nocedal and S. J. Wright.
Numerical optimization.
2nd ed. New York, NY: Springer, 2006.



Himmelblau function and quadratic approximation at (-2,2)



Scheme for Linesearch SQP



Linear Algebra

Interior-point methods and active set methods require to solve symmetric linear equations with saddlepoint structure:

$$\begin{pmatrix} Q & A^\top & B^\top \\ A & 0 & 0 \\ B & 0 & -\Lambda^{-1}S \end{pmatrix}$$

Linear Algebra

Interior-point methods and active set methods require to solve symmetric linear equations with saddlepoint structure:

$$\begin{pmatrix} Q & A^\top & B^\top \\ A & 0 & 0 \\ B & 0 & -\Lambda^{-1} S \end{pmatrix}$$

Semismooth Newton methods yield unsymmetric systems:

$$\begin{pmatrix} Q & A^\top & B^\top \\ A & 0 & 0 \\ \Lambda B & 0 & -S \end{pmatrix}$$

Linear Algebra

Interior-point methods and active set methods require to solve symmetric linear equations with saddlepoint structure:

$$\begin{pmatrix} Q & A^\top & B^\top \\ A & 0 & 0 \\ B & 0 & -\Lambda^{-1}S \end{pmatrix}$$

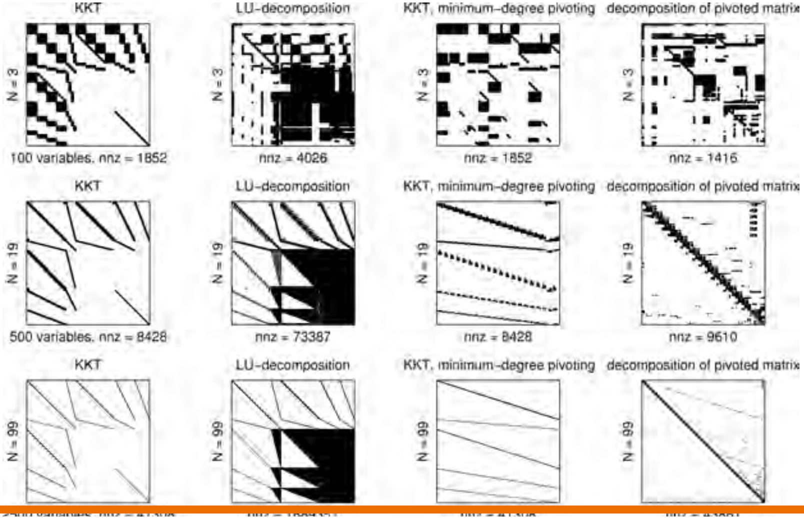
Semismooth Newton methods yield unsymmetric systems:

$$\begin{pmatrix} Q & A^\top & B^\top \\ A & 0 & 0 \\ \Lambda B & 0 & -S \end{pmatrix}$$

- $Q = \nabla^2_{zz} L_h(z, \boldsymbol{\lambda}, \boldsymbol{\mu}, \boldsymbol{\sigma})$: Hessian of Lagrangian (or approximation)
- $A = H_h'(z)$: Jacobian of equality constraints $H_h(z) = 0$
- $B = G_h'(z)$: Jacobian of inequality constraints $G_h(z) \leq 0$
- S, Λ: diagonal matrices, positiv (semi-)definite

Direct Factorization of Sparse Matrices

sparse matrix | dense LU | re-ordering | sparse LU



External Sources

PARDISO (http://www.pardiso-project.org)

free for academic purposes; commercial licenses available; registration required

MA57, MA48 (http://www.hsl.rl.ac.uk)

free for academic purposes; commercial licenses available; registration required

SuperLU (http://crd.lbl.gov/∼xiaoye/SuperLU/)

copyright by Lawrence Berkeley National Laboratory; redistribution and use in source and binary forms, with or without modification, are permitted

Tailored Structure Exploitation for D-OCP

The following problem frequently occurs in linear model-predictive control:

Example (Linear-Quadratic D-OCP)

Minimize

$$\frac{1}{2}\sum_{k=0}^{N-1}\left(x_k^\top V_k x_k + u_k^\top W_k u_k\right)$$

subject to the constraints

$$\begin{aligned} x_{k+1} &= A_k x_k + B_k u_k && (k = 0, \ldots, N-1) \\ x_0 &= \bar{x}_0 && (\bar{x}_0 \text{ given}) \end{aligned}$$

Assumptions:

(A1) W_k symmetric and positive definite for all k

(A2) V_k symmetric and positive semi-definite for all k

Tailored Structure Exploitation for D-OCP

Evaluation of the necessary Karush-Kuhn-Tucker (KKT) conditions yields a large-scale and sparse linear equation:

$$
\left(\begin{array}{ccccc:cccc}
Q_0 & & & & & E_0^\top & M_0^\top & & \\
& Q_1 & & & & & E_1^\top & & \\
& & \ddots & & & & & \ddots & \\
& & & Q_{N-1} & & & & \ddots & M_{N-1}^\top \\
& & & & & & & & E_N^\top \\
\hdashline
E_0 & & & & & & & & \\
M_0 & E_1 & & & & & & & \\
& \ddots & \ddots & & & & & & \\
& & & M_{N-1} & E_N & & & &
\end{array}\right)
\left(\begin{array}{c}
z_0 \\ z_1 \\ \vdots \\ z_{N-1} \\ z_N \\ \hdashline -\boldsymbol{\sigma} \\ \boldsymbol{\lambda}_1 \\ \vdots \\ \boldsymbol{\lambda}_N
\end{array}\right)
=
\left(\begin{array}{c}
0 \\ 0 \\ \vdots \\ 0 \\ 0 \\ \hdashline -\bar{x}_0 \\ 0 \\ \vdots \\ 0
\end{array}\right)
$$

where $z_k = (x_k, u_k)$, $k = 0, \ldots, N-1$, $z_N = x_N$, $S_N = -I$,

$$
Q_k = \begin{pmatrix} V_k & \\ & W_k \end{pmatrix}, \quad M_k = \begin{pmatrix} A_k & B_k \end{pmatrix}, \quad E_k = \begin{pmatrix} -I & 0 \end{pmatrix} \qquad (k = 0, \ldots, N-1)
$$

Tailored Structure Exploitation for D-OCP

Re-arranging the matrix by column and row permutations yields:

$$
\left(\begin{array}{c:cc:cc:cc:cc:c}
 & E_0 & & & & & & & & \\
\hdashline
E_0^\top & Q_0 & M_0^\top & & & & & & & \\
 & M_0 & & E_1 & & & & & & \\
\hdashline
 & & E_1^\top & Q_1 & M_1^\top & & & & & \\
 & & & M_1 & & E_2 & & & & \\
\hdashline
 & & & & \ddots & \ddots & \ddots & & & \\
 & & & & & \ddots & \ddots & E_{N-1} & & \\
\hdashline
 & & & & & & E_{N-1}^\top & Q_{N-1} & M_{N-1}^\top & \\
 & & & & & & & M_{N-1} & & E_N \\
\hdashline
 & & & & & & & & E_N^\top &
\end{array}\right)
\left(\begin{array}{c}
-\boldsymbol{\sigma} \\
\hdashline
z_0 \\
\boldsymbol{\lambda}_1 \\
\hdashline
z_1 \\
\boldsymbol{\lambda}_2 \\
\hdashline
\vdots \\
\hdashline
z_{N-1} \\
\boldsymbol{\lambda}_N \\
\hdashline
z_N
\end{array}\right)
=
\left(\begin{array}{c}
-\bar{x}_0 \\
\hdashline
0 \\
0 \\
\hdashline
0 \\
0 \\
\hdashline
\vdots \\
\hdashline
0 \\
0 \\
\hdashline
0
\end{array}\right)
$$

Tailored Structure Exploitation for D-OCP

Re-arranging the matrix by column and row permutations yields:

$$
\left(\begin{array}{c|cc|cc|ccc|cc|c}
 & E_0 & & & & & & & & & \\ \hline
E_0^\top & Q_0 & M_0^\top & & & & & & & & \\
 & M_0 & & E_1 & & & & & & & \\ \hline
 & & E_1^\top & Q_1 & M_1^\top & & & & & & \\
 & & & M_1 & & E_2 & & & & & \\ \hline
 & & & & \ddots & \ddots & \ddots & & & & \\
 & & & & & \ddots & \ddots & E_{N-1} & & & \\ \hline
 & & & & & & E_{N-1}^\top & Q_{N-1} & M_{N-1}^\top & & \\
 & & & & & & & M_{N-1} & & E_N & \\ \hline
 & & & & & & & & E_N^\top & &
\end{array}\right)
\left(\begin{array}{c}
-\sigma \\ \hline z_0 \\ \lambda_1 \\ \hline z_1 \\ \lambda_2 \\ \hline \vdots \\ \hline z_{N-1} \\ \lambda_N \\ \hline z_N
\end{array}\right)
=
\left(\begin{array}{c}
-\bar{x}_0 \\ \hline 0 \\ 0 \\ \hline 0 \\ 0 \\ \hline \vdots \\ \hline 0 \\ 0 \\ \hline 0
\end{array}\right)
$$

$\rightsquigarrow$ banded symmetric matrix, bandwidth depends only on number of states and controls

$\rightsquigarrow$ computational effort for LU factorization depends linearly on the preview horizon N!

$\rightsquigarrow$ LU factorization by, e.g., LAPACK or INTEL MKS routines DGBTRF, DGBTRS

Tailored Structure Exploitation for D-OCP

- similar structures arise in Interior-Point methods, SQP methods, or semismooth Newton methods when applied to D-OCP

Tailored Structure Exploitation for D-OCP

- similar structures arise in Interior-Point methods, SQP methods, or semismooth Newton methods when applied to D-OCP
- If coupled boundary conditions, coupled Mayer terms or parameters are included, linear systems of the following type arise:

$$\left[\begin{array}{c:c} \Gamma & V^\top \\ \hdashline V & \Lambda \end{array}\right] \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \alpha \\ \beta \end{bmatrix}$$

(Γ large scale, banded, Λ low dimensional)

Tailored Structure Exploitation for D-OCP

- similar structures arise in Interior-Point methods, SQP methods, or semismooth Newton methods when applied to D-OCP
- If coupled boundary conditions, coupled Mayer terms or parameters are included, linear systems of the following type arise:

$$\left[\begin{array}{c:c} \Gamma & V^\top \\ \hdashline V & \Lambda \end{array}\right] \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \boldsymbol{\alpha} \\ \boldsymbol{\beta} \end{bmatrix}$$

(Γ large scale, banded, Λ low dimensional)

Solution procedure:

(1) Compute LU decompostion of block diagonal matrix Γ by LAPACK with OpenBLAS (or INTEL MKL).

(2) Solve low dimensional (dense) system

$$\left(\Lambda - V\Gamma^{-1}V^\top\right) y = \boldsymbol{\beta} - V\Gamma^{-1}\boldsymbol{\alpha}$$

(3) Solve large dimensional (banded) system $\Gamma x = \boldsymbol{\alpha} - V^\top y$.

Tailored Structure Exploitation for D-OCP

- similar structures arise in Interior-Point methods, SQP methods, or semismooth Newton methods when applied to D-OCP
- If coupled boundary conditions, coupled Mayer terms or parameters are included, linear systems of the following type arise:

$$\left[\begin{array}{c:c} \Gamma & V^\top \\ \hdashline V & \Lambda \end{array}\right] \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \alpha \\ \beta \end{bmatrix}$$

(Γ large scale, banded, Λ low dimensional)

Solution procedure:

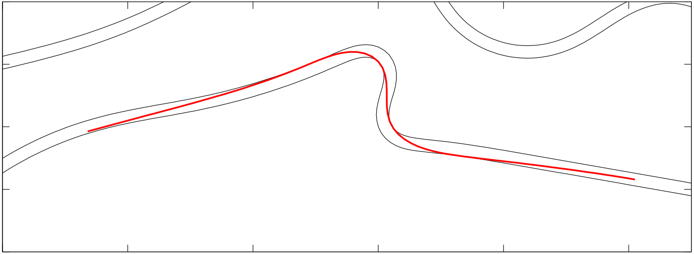
(1) Compute LU decompostion of block diagonal matrix Γ by LAPACK with OPENBLAS (or INTEL MKL).
(2) Solve low dimensional (dense) system

$$\left(\Lambda - V\Gamma^{-1}V^\top\right) y = \beta - V\Gamma^{-1}\alpha$$

(3) Solve large dimensional (banded) system $\Gamma x = \alpha - V^\top y$.

[A. Huber, M. Gerdts, E. Bertolazzi: Structure Exploitation in an Interior-Point Method for Fully Discretized, State Constrained Optimal Control Problems, Vietnam Journal of Mathematics, Vol. 46(4), pp. 1089–1113, 2018.]

Path Generation, Test 1



Solution with Interior Point method on a horizon of 500 [m] and 80 gridpoints needs 0.022 [s] to optimize (solver: `https://www.ocpbasic.com/`)

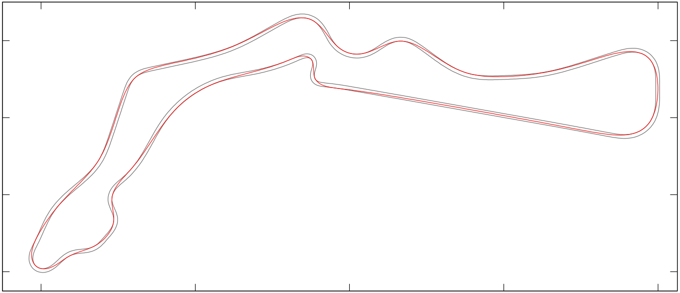
Path Generation, Test 1, parallel $\Gamma^{-1} V^{\top}$

Grid points	LAPACK KKT		HSL/MA57	
	T_{ges}	T_{lin}	T_{ges}	T_{lin}
10	0.002600 s	0.001125 s	0.003664 s	0.002169 s
100	0.022445 s	0.008535 s	0.035134 s	0.020237 s
1000	0.152356 s	0.080296 s	0.555060 s	0.479449 s
10000	1.592431 s	0.754386 s	10.160251 s	9.498326 s
100000	10.875577 s	6.980800 s	427.733114 s	423.264320 s
150000	17.158301 s	10.833331 s	931.503742 s	924.729321 s

Tabelle: Test of linear solvers in an Interior Point method (solver: `https://www.ocpbasic.com/`)

```
export OMP_PROC_BIND=TRUE
export OMP_WAIT_POLICY=PASSIVE
```

Path Generation Full Lap



Optimal control problem with free final time and boundary conditions.

Solution for a lap with 600 gridpoints needs 0.26 s to optimize.

Contents

Introduction

Direct Discretization : First Discretize – Then Optimize
- Overview
- Full Discretization (Collocation, Direct Transcription)
- Numerical Solution of Discretized Problem
- Reduced Discretization (Shooting)
- Adjoint Estimation
- Convergence

Applications

Example

Example

Minimize $\alpha x(1) + y(1)$ subject to

$$x'(t) = u(t), \qquad x(0) = 0,$$

$$y'(t) = \frac{1}{2}u(t)^2, \qquad y(0) = 0.$$

Example

Example

Minimize $\alpha x(1) + y(1)$ subject to

$$x'(t) = u(t), \qquad x(0) = 0,$$
$$y'(t) = \frac{1}{2}u(t)^2, \qquad y(0) = 0.$$

Fully discretized problem:

Minimize $\alpha x_N + y_N$ w.r.t. $(x_0, y_0, u_0, \ldots, x_{N-1}, y_{N-1}, u_{N-1}, x_N, y_N)^\top$ subject to

$$\frac{x_{i+1} - x_i}{h} - u_i = 0, \qquad x_0 = 0 \qquad (i = 0, \ldots, N-1)$$
$$\frac{y_{i+1} - y_i}{h} - \frac{1}{2}u_i^2 = 0, \qquad y_0 = 0 \qquad (i = 0, \ldots, N-1)$$

Example

Example

Minimize $\alpha x_N + y_N$ subject to

$$\frac{x_{i+1} - x_i}{h} - u_i = 0, \qquad x_0 = 0 \qquad (i = 0, \ldots, N-1)$$

$$\frac{y_{i+1} - y_i}{h} - \frac{1}{2}u_i^2 = 0, \qquad y_0 = 0 \qquad (i = 0, \ldots, N-1)$$

Example

Example

Minimize $\alpha x_N + y_N$ subject to

$$\frac{x_{i+1} - x_i}{h} - u_i = 0, \qquad x_0 = 0 \qquad (i = 0, \ldots, N-1)$$
$$\frac{y_{i+1} - y_i}{h} - \frac{1}{2} u_i^2 = 0, \qquad y_0 = 0 \qquad (i = 0, \ldots, N-1)$$

Solving constraints yields:

$$x_i = h \sum_{j=0}^{i-1} u_j, \qquad y_i = \frac{h}{2} \sum_{j=0}^{i-1} u_j^2 \qquad (i = 1, \ldots, N)$$

Example

Example

Minimize $\alpha x_N + y_N$ subject to

$$\frac{x_{i+1} - x_i}{h} - u_i = 0, \qquad x_0 = 0 \qquad (i = 0, \dots, N-1)$$

$$\frac{y_{i+1} - y_i}{h} - \frac{1}{2} u_i^2 = 0, \qquad y_0 = 0 \qquad (i = 0, \dots, N-1)$$

Solving constraints yields:

$$x_i = h \sum_{j=0}^{i-1} u_j, \qquad y_i = \frac{h}{2} \sum_{j=0}^{i-1} u_j^2 \qquad (i = 1, \dots, N)$$

Reduced discretization:

$$\text{Minimize} \quad h \sum_{j=0}^{N-1} \left(\alpha u_j + \frac{1}{2} u_j^2 \right) \quad \text{w.r.t.} \quad u_h = (u_0, \dots, u_{N-1})^\top$$

Reduced Discretization (Direct Single Shooting)

Fully discretized optimal control problem

Minimize

$$\varphi(x_0, x_N)$$

s.t.

$$\begin{aligned} x_i + h\Phi(t_i, x_i, w, h) - x_{i+1} &= 0, \quad i = 0, 1, \ldots, N-1, \\ c(t_i, x_i, u_h(t_i; w)) &\leq 0, \quad i = 0, 1, \ldots, N, \\ \psi(x_0, x_N) &= 0. \end{aligned}$$

Notation:

- $x_h = (x_0, \ldots, x_N)^\top$: state discretization
- $w = (w_1, \ldots, w_M)^\top$: control parameterization

Reduced Discretization (Direct Single Shooting)

Reduction of size by solving the discretized differential equations:

Reduced Discretization (Direct Single Shooting)

Reduction of size by solving the discretized differential equations:

$$x_1 = x_0 + h\Phi(t_0, x_0, w, h)$$

Reduced Discretization (Direct Single Shooting)

Reduction of size by solving the discretized differential equations:

$$x_1 = x_0 + h\Phi(t_0, x_0, w, h) \qquad =: X_1(x_0, w),$$

Reduced Discretization (Direct Single Shooting)

Reduction of size by solving the discretized differential equations:

$$
\begin{aligned}
x_1 &= x_0 + h\Phi(t_0, x_0, w, h) && =: X_1(x_0, w),\\
x_2 &= x_1 + h\Phi(t_1, x_1, w, h)
\end{aligned}
$$

Reduced Discretization (Direct Single Shooting)

Reduction of size by solving the discretized differential equations:

$$
\begin{aligned}
x_1 &= x_0 + h\Phi(t_0, x_0, w, h) && =: X_1(x_0, w), \\
x_2 &= x_1 + h\Phi(t_1, x_1, w, h) \\
&= X_1(x_0, w) + h\Phi(t_1, X_1(x_0, w), w, h)
\end{aligned}
$$

Reduced Discretization (Direct Single Shooting)

Reduction of size by solving the discretized differential equations:

$$\begin{aligned} x_1 &= x_0 + h\Phi(t_0, x_0, w, h) && =: X_1(x_0, w), \\ x_2 &= x_1 + h\Phi(t_1, x_1, w, h) \\ &= X_1(x_0, w) + h\Phi(t_1, X_1(x_0, w), w, h) && =: X_2(x_0, w), \end{aligned}$$

Reduced Discretization (Direct Single Shooting)

Reduction of size by solving the discretized differential equations:

$$
\begin{aligned}
x_1 &= x_0 + h\Phi(t_0, x_0, w, h) && =: X_1(x_0, w), \\
x_2 &= x_1 + h\Phi(t_1, x_1, w, h) \\
&= X_1(x_0, w) + h\Phi(t_1, X_1(x_0, w), w, h) && =: X_2(x_0, w), \\
&\vdots
\end{aligned}
$$

Reduced Discretization (Direct Single Shooting)

Reduction of size by solving the discretized differential equations:

$$
\begin{aligned}
x_1 &= x_0 + h\Phi(t_0, x_0, w, h) && =: X_1(x_0, w),\\
x_2 &= x_1 + h\Phi(t_1, x_1, w, h) &&\\
&= X_1(x_0, w) + h\Phi(t_1, X_1(x_0, w), w, h) && =: X_2(x_0, w),\\
&\vdots &&\\
x_N &= x_{N-1} + h\Phi(t_{N-1}, x_{N-1}, w, h) &&
\end{aligned}
$$

Reduced Discretization (Direct Single Shooting)

Reduction of size by solving the discretized differential equations:

$$
\begin{aligned}
x_1 &= x_0 + h\Phi(t_0, x_0, w, h) && =: X_1(x_0, w),\\
x_2 &= x_1 + h\Phi(t_1, x_1, w, h) &&\\
&= X_1(x_0, w) + h\Phi(t_1, X_1(x_0, w), w, h) && =: X_2(x_0, w),\\
&\vdots &&\\
x_N &= x_{N-1} + h\Phi(t_{N-1}, x_{N-1}, w, h) &&\\
&= X_{N-1}(x_0, w) + h\Phi(t_{N-1}, X_{N-1}(x_0, w), w, h_{N-1}) &&
\end{aligned}
$$

Reduced Discretization (Direct Single Shooting)

Reduction of size by solving the discretized differential equations:

$$
\begin{aligned}
x_1 &= x_0 + h\Phi(t_0, x_0, w, h) && =: X_1(x_0, w),\\
x_2 &= x_1 + h\Phi(t_1, x_1, w, h) \\
&= X_1(x_0, w) + h\Phi(t_1, X_1(x_0, w), w, h) && =: X_2(x_0, w),\\
&\vdots \\
x_N &= x_{N-1} + h\Phi(t_{N-1}, x_{N-1}, w, h) \\
&= X_{N-1}(x_0, w) + h\Phi(t_{N-1}, X_{N-1}(x_0, w), w, h_{N-1}) && =: X_N(x_0, w).
\end{aligned}
$$

Reduced Discretization (Direct Single Shooting)

Reduction of size by solving the discretized differential equations:

$$
\begin{aligned}
x_1 &= x_0 + h\Phi(t_0, x_0, w, h) && =: X_1(x_0, w), \\
x_2 &= x_1 + h\Phi(t_1, x_1, w, h) \\
&= X_1(x_0, w) + h\Phi(t_1, X_1(x_0, w), w, h) && =: X_2(x_0, w), \\
&\vdots \\
x_N &= x_{N-1} + h\Phi(t_{N-1}, x_{N-1}, w, h) \\
&= X_{N-1}(x_0, w) + h\Phi(t_{N-1}, X_{N-1}(x_0, w), w, h_{N-1}) && =: X_N(x_0, w).
\end{aligned}
$$

Remarks:

Reduced Discretization (Direct Single Shooting)

Reduction of size by solving the discretized differential equations:

$$
\begin{aligned}
x_1 &= x_0 + h\Phi(t_0, x_0, w, h) && =: X_1(x_0, w),\\
x_2 &= x_1 + h\Phi(t_1, x_1, w, h) &&\\
&= X_1(x_0, w) + h\Phi(t_1, X_1(x_0, w), w, h) && =: X_2(x_0, w),\\
&\vdots &&\\
x_N &= x_{N-1} + h\Phi(t_{N-1}, x_{N-1}, w, h) &&\\
&= X_{N-1}(x_0, w) + h\Phi(t_{N-1}, X_{N-1}(x_0, w), w, h_{N-1}) && =: X_N(x_0, w).
\end{aligned}
$$

Remarks:

- The state trajectory is fully determined by the initial value x_0 and the control parameterization w (direct shooting idea).

Reduced Discretization (Direct Single Shooting)

Reduction of size by solving the discretized differential equations:

$$
\begin{aligned}
x_1 &= x_0 + h\Phi(t_0, x_0, w, h) && =: X_1(x_0, w),\\
x_2 &= x_1 + h\Phi(t_1, x_1, w, h) &&\\
&= X_1(x_0, w) + h\Phi(t_1, X_1(x_0, w), w, h) && =: X_2(x_0, w),\\
&\vdots &&\\
x_N &= x_{N-1} + h\Phi(t_{N-1}, x_{N-1}, w, h) &&\\
&= X_{N-1}(x_0, w) + h\Phi(t_{N-1}, X_{N-1}(x_0, w), w, h_{N-1}) && =: X_N(x_0, w).
\end{aligned}
$$

Remarks:

- The state trajectory is fully determined by the initial value x_0 and the control parameterization w (direct shooting idea).
- The above procedure is nothing else than solving the initial value problem

$$x'(t) = f(t, x(t), u_h(t; w)), \quad x(t_0) = x_0$$

with the one-step method with increment function Φ.

Reduced Discretization (Direct Single Shooting)

Reduced Discretization (RD-OCP)

Minimize

$$\varphi(x_0, X_N(x_0, w))$$

with respect to $x_0 \in \mathbb{R}^{n_x}$ and $w \in \mathbb{R}^{Mn_u}$ subject to the constraints

$$\begin{aligned} \psi(x_0, X_N(x_0, w)) &= 0, \\ c(t_j, X_j(x_0, w), u_h(t_j; w)) &\leq 0, \qquad\qquad j = 0, 1, \ldots, N. \end{aligned}$$

Remarks:

- much smaller than D-OCP (fewer optimization variables and fewer constraints)
- but: more nonlinear than D-OCP

Reduced Discretization (Direct Single Shooting)

The reduced discretization is again a finite dimensional nonlinear optimization problem, but of reduced size:

Reduced Nonlinear Optimization Problem (R-NLP)

Minimize

$$J_h(z) := \varphi(x_0, X_N(x_0, w))$$

w.r.t. $z = (x_0, w)^\top \in \mathbb{R}^{n_x + Mn_u}$ subject to the constraints

$$H_h(z) = 0, \qquad G_h(z) \leq 0,$$

where

$$H_h(z) := \psi(x_0, X_N(x_0, w)), \quad G_h(z) := \begin{pmatrix} c(t_0, x_0, u_h(t_0; w)) \\ c(t_1, X_1(x_0, w), u_h(t_1; w)) \\ \vdots \\ c(t_N, X_N(x_0, w), u_h(t_N; w)) \end{pmatrix}.$$

Reduced Discretization (Direct Single Shooting)

Derivatives: (required in NLP solver)

$$J_h'(z) = \left(\varphi_{x_0}' + \varphi_{x_f}' \cdot X_{N,x_0}' \;\middle|\; \varphi_{x_f}' \cdot X_{N,w}' \right)$$

Reduced Discretization (Direct Single Shooting)

Derivatives: (required in NLP solver)

$$
\begin{aligned}
J_h'(z) &= \left(\varphi_{x_0}' + \varphi_{x_f}' \cdot X_{N,x_0}' \;\middle|\; \varphi_{x_f}' \cdot X_{N,w}' \right) \\
G_h'(z) &= \left(\begin{array}{c|c}
c_x'[t_0] & c_x'[t_0] + c_u'[t_0] \cdot u_{h,w}'(t_0; w) \\
c_x'[t_1] \cdot X_{1,x_0}' & c_x'[t_1] \cdot X_{1,w}' + c_u'[t_1] \cdot u_{h,w}'(t_1; w) \\
\vdots & \vdots \\
c_x'[t_N] \cdot X_{N,x_0}' & c_x'[t_N] \cdot X_{N,w}' + c_u'[t_N] \cdot u_{h,w}'(t_N; w)
\end{array} \right)
\end{aligned}
$$

Reduced Discretization (Direct Single Shooting)

Derivatives: (required in NLP solver)

$$
\begin{aligned}
J_h'(z) &= \left(\varphi_{x_0}' + \varphi_{x_f}' \cdot X_{N,x_0}' \;\middle|\; \varphi_{x_f}' \cdot X_{N,w}' \right) \\
G_h'(z) &= \left(\begin{array}{c|c}
c_x'[t_0] & c_x'[t_0] + c_u'[t_0] \cdot u_{h,w}'(t_0; w) \\
c_x'[t_1] \cdot X_{1,x_0}' & c_x'[t_1] \cdot X_{1,w}' + c_u'[t_1] \cdot u_{h,w}'(t_1; w) \\
\vdots & \vdots \\
c_x'[t_N] \cdot X_{N,x_0}' & c_x'[t_N] \cdot X_{N,w}' + c_u'[t_N] \cdot u_{h,w}'(t_N; w)
\end{array} \right) \\
H_h'(z) &= \left(\psi_{x_0}' + \psi_{x_f}' \cdot X_{N,x_0}' \;\middle|\; \psi_{x_f}' \cdot X_{N,w}' \right)
\end{aligned}
$$

Computation of Derivatives

How to compute derivatives J'_h, G'_h, H'_h and/or sensitivity matrices

$$X'_{i,x_0}(x_0, w), \quad X'_{i,w}(x_0, w), \quad i = 1, \ldots, N \,?$$

Computation of Derivatives

How to compute derivatives J_h', G_h', H_h' and/or sensitivity matrices

$$X'_{i,x_0}(x_0, w), \quad X'_{i,w}(x_0, w), \quad i = 1, \ldots, N\ ?$$

Different approaches exist:

(a) The sensitivity differential equation approach or Internal Numerical Differentiation (IND) approach [Bock] is advantageous if the number of constraints is (much) larger than the number of variables in the discretized problem.

Computation of Derivatives

How to compute derivatives J_h', G_h', H_h' and/or sensitivity matrices

$$X'_{i,x_0}(x_0, w), \quad X'_{i,w}(x_0, w), \quad i = 1, \ldots, N \,?$$

Different approaches exist:

(a) The sensitivity differential equation approach or Internal Numerical Differentiation (IND) approach [Bock] is advantageous if the number of constraints is (much) larger than the number of variables in the discretized problem.

(b) The adjoint equation approach is preferable if the number of constraints is less than the number of variables in the discretized problem.

Computation of Derivatives

How to compute derivatives J_h', G_h', H_h' and/or sensitivity matrices

$$X'_{i,x_0}(x_0, w), \quad X'_{i,w}(x_0, w), \quad i = 1, \ldots, N\,?$$

Different approaches exist:

(a) The sensitivity differential equation approach or Internal Numerical Differentiation (IND) approach [Bock] is advantageous if the number of constraints is (much) larger than the number of variables in the discretized problem.

(b) The adjoint equation approach is preferable if the number of constraints is less than the number of variables in the discretized problem.

(c) A powerful tool for the evaluation of derivatives is algorithmic differentiation, see `www.autodiff.org`.

Computation of Derivatives

How to compute derivatives J_h', G_h', H_h' and/or sensitivity matrices

$$X'_{i,x_0}(x_0, w), \quad X'_{i,w}(x_0, w), \quad i = 1, \ldots, N\ ?$$

Different approaches exist:

(a) The sensitivity differential equation approach or Internal Numerical Differentiation (IND) approach [Bock] is advantageous if the number of constraints is (much) larger than the number of variables in the discretized problem.

(b) The adjoint equation approach is preferable if the number of constraints is less than the number of variables in the discretized problem.

(c) A powerful tool for the evaluation of derivatives is algorithmic differentiation, see `www.autodiff.org`.

(d) The approximation by finite differences is straightforward, but has the drawback of being computationally expensive and often suffers from low accuracy.

Gradient Computation by Adjoint Equation

Given:

- One-step discretization scheme ($z = (x_0, w)^\top$):

$$\begin{aligned} X_0(z) &= x_0, \\ X_{i+1}(z) &= X_i(z) + h\Phi(t_i, X_i(z), w, h), \qquad i = 0, 1, \ldots, N-1, \end{aligned}$$

Gradient Computation by Adjoint Equation

Given:

- One-step discretization scheme ($z = (x_0, w)^\top$):

$$\begin{aligned} X_0(z) &= x_0, \\ X_{i+1}(z) &= X_i(z) + h\Phi(t_i, X_i(z), w, h), \qquad i = 0, 1, \ldots, N-1, \end{aligned}$$

- A function Γ of type

$$\Gamma(z) := \gamma(x_0, X_N(z), w).$$

$\rightsquigarrow$ objective function, boundary condition, state constraint (with N replaced by i)

Gradient Computation by Adjoint Equation

Given:

- One-step discretization scheme ($z = (x_0, w)^\top$):

$$\begin{aligned} X_0(z) &= x_0, \\ X_{i+1}(z) &= X_i(z) + h\Phi(t_i, X_i(z), w, h), \qquad i = 0, 1, \ldots, N-1, \end{aligned}$$

- A function Γ of type

$$\Gamma(z) := \gamma(x_0, X_N(z), w).$$

$\rightsquigarrow$ objective function, boundary condition, state constraint (with N replaced by i)

Goals:

- Compute gradient of Γ with respect to z.

Gradient Computation by Adjoint Equation

Given:

- One-step discretization scheme ($z = (x_0, w)^\top$):

$$\begin{aligned} X_0(z) &= x_0, \\ X_{i+1}(z) &= X_i(z) + h\Phi(t_i, X_i(z), w, h), \qquad i = 0, 1, \ldots, N-1, \end{aligned}$$

- A function Γ of type

$$\Gamma(z) := \gamma(x_0, X_N(z), w).$$

$\rightsquigarrow$ objective function, boundary condition, state constraint (with N replaced by i)

Goals:

- Compute gradient of Γ with respect to z.
- Avoid the costly computation of the sensitivity matrices S_i, $i = 0, \ldots, N$, with IND.

Gradient Computation by Adjoint Equation

Define the auxiliary functional Γ_a using multipliers $\boldsymbol{\lambda}_i$, $i = 1, \ldots, N$:

$$\Gamma_a(z) := \Gamma(z) + \sum_{i=0}^{N-1} \boldsymbol{\lambda}_{i+1}^\top \left(X_{i+1}(z) - X_i(z) - h\Phi(t_i, X_i(z), w, h)\right)$$

Gradient Computation by Adjoint Equation

Define the auxiliary functional Γ_a using multipliers $\boldsymbol{\lambda}_i$, $i = 1, \ldots, N$:

$$\Gamma_a(z) := \Gamma(z) + \sum_{i=0}^{N-1} \boldsymbol{\lambda}_{i+1}^\top \left(X_{i+1}(z) - X_i(z) - h\Phi(t_i, X_i(z), w, h) \right)$$

Differentiating Γ_a with respect to z leads to the expression

$$\begin{aligned}
\Gamma_a'(z) \quad &= \quad \left(\gamma'_{x_0} - \boldsymbol{\lambda}_1^\top - h\boldsymbol{\lambda}_1^\top \Phi'_x[t_0]\right) \cdot S_0 + \left(\gamma'_{x_N} + \boldsymbol{\lambda}_N^\top\right) \cdot S_N + \gamma'_w \\
&\quad + \sum_{i=1}^{N-1} \left(\boldsymbol{\lambda}_i^\top - \boldsymbol{\lambda}_{i+1}^\top - h\boldsymbol{\lambda}_{i+1}^\top \Phi'_x[t_i]\right) \cdot S_i - \sum_{i=0}^{N-1} h\boldsymbol{\lambda}_{i+1}^\top \Phi'_w[t_i] \cdot \frac{\partial w}{\partial z}.
\end{aligned}$$

Gradient Computation by Adjoint Equation

Define the auxiliary functional Γ_a using multipliers $\boldsymbol{\lambda}_i$, $i = 1, \ldots, N$:

$$\Gamma_a(z) := \Gamma(z) + \sum_{i=0}^{N-1} \boldsymbol{\lambda}_{i+1}^\top \left(X_{i+1}(z) - X_i(z) - h\Phi(t_i, X_i(z), w, h)\right)$$

Differentiating Γ_a with respect to z leads to the expression

$$\begin{aligned}\Gamma_a'(z) \quad = \quad & \left(\gamma_{x_0}' - \boldsymbol{\lambda}_1^\top - h\boldsymbol{\lambda}_1^\top \Phi_x'[t_0]\right) \cdot S_0 + \left(\gamma_{x_N}' + \boldsymbol{\lambda}_N^\top\right) \cdot S_N + \gamma_w' \\ & + \sum_{i=1}^{N-1} \left(\boldsymbol{\lambda}_i^\top - \boldsymbol{\lambda}_{i+1}^\top - h\boldsymbol{\lambda}_{i+1}^\top \Phi_x'[t_i]\right) \cdot S_i - \sum_{i=0}^{N-1} h\boldsymbol{\lambda}_{i+1}^\top \Phi_w'[t_i] \cdot \frac{\partial w}{\partial z}.\end{aligned}$$

$S_i = X_i'(z)$: sensitivities (costly to compute; avoid!)

Gradient Computation by Adjoint Equation

Define the auxiliary functional Γ_a using multipliers $\boldsymbol{\lambda}_i$, $i = 1, \ldots, N$:

$$\Gamma_a(z) := \Gamma(z) + \sum_{i=0}^{N-1} \boldsymbol{\lambda}_{i+1}^\top \left(X_{i+1}(z) - X_i(z) - h\Phi(t_i, X_i(z), w, h) \right)$$

Differentiating Γ_a with respect to z leads to the expression

$$\begin{aligned} \Gamma_a'(z) \quad = \quad & \left(\gamma_{x_0}' - \boldsymbol{\lambda}_1^\top - h\boldsymbol{\lambda}_1^\top \Phi_x'[t_0] \right) \cdot S_0 + \left(\gamma_{x_N}' + \boldsymbol{\lambda}_N^\top \right) \cdot S_N + \gamma_w' \\ & + \sum_{i=1}^{N-1} \left(\boldsymbol{\lambda}_i^\top - \boldsymbol{\lambda}_{i+1}^\top - h\boldsymbol{\lambda}_{i+1}^\top \Phi_x'[t_i] \right) \cdot S_i - \sum_{i=0}^{N-1} h\boldsymbol{\lambda}_{i+1}^\top \Phi_w'[t_i] \cdot \frac{\partial w}{\partial z}. \end{aligned}$$

$S_i = X_i'(z)$: sensitivities (costly to compute; avoid!)

Idea: Choose $\boldsymbol{\lambda}_i$ such that terms involving S_i vanish.

Gradient Computation by Adjoint Equation

Discrete adjoint equation: (to be solved backwards in time)

$$\boldsymbol{\lambda}_i^\top - \boldsymbol{\lambda}_{i+1}^\top - h\boldsymbol{\lambda}_{i+1}^\top \Phi'_x[t_i] = 0, \quad i = 0, \dots, N-1$$

Transversality condition: (terminal condition at $t = t_N$)

$$\boldsymbol{\lambda}_N^\top = -\gamma'_{x_N}(x_0, X_N(z), w)$$

Gradient Computation by Adjoint Equation

Discrete adjoint equation: (to be solved backwards in time)

$$\boldsymbol{\lambda}_i^\top - \boldsymbol{\lambda}_{i+1}^\top - h\boldsymbol{\lambda}_{i+1}^\top \Phi'_x[t_i] = 0, \quad i = 0, \ldots, N-1$$

Transversality condition: (terminal condition at $t = t_N$)

$$\boldsymbol{\lambda}_N^\top = -\gamma'_{x_N}(x_0, X_N(z), w)$$

Then:

$$\Gamma'_a(z) = \left(\gamma'_{x_0} - \boldsymbol{\lambda}_0^\top\right) \cdot S_0 + \gamma'_w - \sum_{i=0}^{N-1} h\boldsymbol{\lambda}_{i+1}^\top \Phi'_w[t_i] \cdot \frac{\partial w}{\partial z},$$

where $S_0 = \left(\begin{array}{c|c} I & 0 \end{array} \right)$.

Gradient Computation by Adjoint Equation

Discrete adjoint equation: (to be solved backwards in time)

$$\boldsymbol{\lambda}_i^\top - \boldsymbol{\lambda}_{i+1}^\top - h\boldsymbol{\lambda}_{i+1}^\top \Phi'_x[t_i] = 0, \quad i = 0, \dots, N-1$$

Transversality condition: (terminal condition at $t = t_N$)

$$\boldsymbol{\lambda}_N^\top = -\gamma'_{x_N}(x_0, X_N(z), w)$$

Then:

$$\Gamma'_a(z) = \left(\gamma'_{x_0} - \boldsymbol{\lambda}_0^\top\right) \cdot S_0 + \gamma'_w - \sum_{i=0}^{N-1} h\boldsymbol{\lambda}_{i+1}^\top \Phi'_w[t_i] \cdot \frac{\partial w}{\partial z},$$

where $S_0 = \left(\begin{array}{c|c} I & 0 \end{array} \right)$.

What is the relation between Γ'_a and Γ' ?

Gradient Computation by Adjoint Equation

Theorem

It holds

$$\Gamma'(z) = \Gamma'_a(z) = \left(\gamma'_{x_0} - \boldsymbol{\lambda}_0^\top\right) \cdot S_0 + \gamma'_w - \sum_{i=0}^{N-1} h \boldsymbol{\lambda}_{i+1}^\top \Phi'_w[t_i] \cdot \frac{\partial w}{\partial z}.$$

Notes:

Gradient Computation by Adjoint Equation

Theorem

It holds

$$\Gamma'(z) = \Gamma'_a(z) = \left(\gamma'_{x_0} - \boldsymbol{\lambda}_0^\top\right) \cdot S_0 + \gamma'_w - \sum_{i=0}^{N-1} h \boldsymbol{\lambda}_{i+1}^\top \Phi'_w[t_i] \cdot \frac{\partial w}{\partial z}.$$

Notes:

- With $\Gamma'(z) = \Gamma'_a(z)$ we found a formula for the gradient of Γ.

Gradient Computation by Adjoint Equation

Theorem

It holds

$$\Gamma'(z) = \Gamma'_a(z) = \left(\gamma'_{x_0} - \boldsymbol{\lambda}_0^\top\right) \cdot S_0 + \gamma'_w - \sum_{i=0}^{N-1} h \boldsymbol{\lambda}_{i+1}^\top \Phi'_w[t_i] \cdot \frac{\partial w}{\partial z}.$$

Notes:

- With $\Gamma'(z) = \Gamma'_a(z)$ we found a formula for the gradient of Γ.
- size of the adjoint equation independent of dimension of w

Gradient Computation by Adjoint Equation

Theorem

It holds

$$\Gamma'(z) = \Gamma_a'(z) = \left(\gamma_{x_0}' - \boldsymbol{\lambda}_0^\top\right) \cdot S_0 + \gamma_w' - \sum_{i=0}^{N-1} h \boldsymbol{\lambda}_{i+1}^\top \Phi_w'[t_i] \cdot \frac{\partial w}{\partial z}.$$

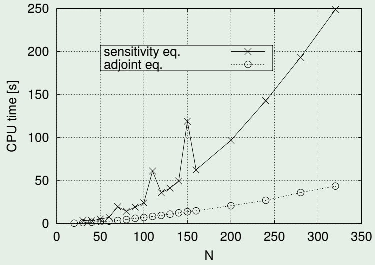
Notes:

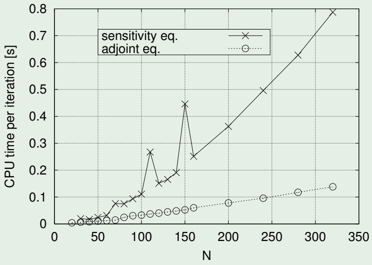
- With $\Gamma'(z) = \Gamma_a'(z)$ we found a formula for the gradient of Γ.
- size of the adjoint equation independent of dimension of w
- number of required adjoint equations depends on number of constraints

Gradient Computation by Adjoint Equation

Example

We compare the CPU times for the emergency landing maneuver without dynamic pressure constraint for the sensitivity equation approach and the adjoint equation approach for different values of N:





Contents

Introduction

Direct Discretization : First Discretize – Then Optimize
- Overview
- Full Discretization (Collocation, Direct Transcription)
- Numerical Solution of Discretized Problem
- Reduced Discretization (Shooting)
- Adjoint Estimation
- Convergence

Applications

Optimal Control Problem and its Euler Discretization

Goal:

- Derive adjoint estimates from the optimal solution of D-OCP

Optimal Control Problem and its Euler Discretization

Goal:

- Derive adjoint estimates from the optimal solution of D-OCP

Approach:

- Compare necessary conditions for OCP and its discretization D-OCP

Optimal Control Problem and its Euler Discretization

Goal:

- Derive adjoint estimates from the optimal solution of D-OCP

Approach:

- Compare necessary conditions for OCP and its discretization D-OCP
- Here we restrict the discussion to problems with mixed control-state constraints only.

Optimal Control Problem and its Euler Discretization

Goal:

- Derive adjoint estimates from the optimal solution of D-OCP

Approach:

- Compare necessary conditions for OCP and its discretization D-OCP
- Here we restrict the discussion to problems with mixed control-state constraints only.
- Pure state constraints are discussed, e.g., in
  [M. Gerdts: Optimal Control of ODEs and DAEs, De Gruyter, 2011]

Optimal Control Problem and its Discretization

OCP

Find an absolutely continuous state vector x and an essentially bounded control vector u such that

$$\varphi(x(t_0), x(t_f))$$

becomes minimal subject to the differential equation

$$x'(t) = f(t, x(t), u(t)) \quad \text{a.e. in } [t_0, t_f],$$

the control-state constraints

$$c(t, x(t), u(t)) \leq 0 \quad \text{a.e. in } [t_0, t_f],$$

and the boundary conditions

$$\psi(x(t_0), x(t_f)) = 0.$$

Optimal Control Problem and its Discretization

OCP

Find an absolutely continuous state vector x and an essentially bounded control vector u such that

$$\varphi(x(t_0), x(t_f))$$

becomes minimal subject to the differential equation

$$x'(t) = f(t, x(t), u(t)) \quad \text{a.e. in } [t_0, t_f],$$

the control-state constraints

$$c(t, x(t), u(t)) \leq 0 \quad \text{a.e. in } [t_0, t_f],$$

and the boundary conditions

$$\psi(x(t_0), x(t_f)) = 0.$$

D-OCP

Find $x_h = (x_0, \dots, x_N)^\top$ and $u_h = (u_0, \dots, u_{N-1})^\top$ such that

$$\varphi(x_0, x_N)$$

becomes minimal subject to the discretized dynamic constraints

$$x_i + h_i f(t_i, x_i, u_i) - x_{i+1} = 0, \quad i = 0, 1, \dots, N-1,$$

the discretized control-state constraints

$$c(t_i, x_i, u_i) \leq 0, \quad i = 0, 1, \dots, N-1,$$

and the boundary conditions

$$\psi(x_0, x_N) = 0.$$

Necessary Conditions of Optimal Control Problem

Augmented Hamilton function: $\hat{\mathcal{H}}(t,x,u,\boldsymbol{\lambda},\boldsymbol{\eta}) := \boldsymbol{\lambda}^\top f(t,x,u) + \boldsymbol{\eta}^\top c(t,x,u)$

Local Minimum Principle

$(\hat{x},\hat{u})$ local minimum of OCP. Then:

- $\ell_0 \geq 0$, $(\ell_0,\boldsymbol{\sigma},\boldsymbol{\lambda},\boldsymbol{\eta}) \neq 0$
- Adjoint differential equation:

$$\boldsymbol{\lambda}'(t) = -\nabla_x \hat{\mathcal{H}}(t,\hat{x}(t),\hat{u}(t),\boldsymbol{\lambda}(t),\boldsymbol{\eta}(t))$$

- Transversality conditions:

$$\boldsymbol{\lambda}(t_0) = -\nabla_{x_0}\left(\ell_0\boldsymbol{\varphi} + \boldsymbol{\sigma}^\top\boldsymbol{\psi}\right), \qquad \boldsymbol{\lambda}(t_f) = \nabla_{x_f}\left(\ell_0\boldsymbol{\varphi} + \boldsymbol{\sigma}^\top\boldsymbol{\psi}\right)$$

- Stationarity of augmented Hamilton function: Almost everywhere we have

$$\nabla_u \hat{\mathcal{H}}(t,\hat{x}(t),\hat{u}(t),\boldsymbol{\lambda}(t),\boldsymbol{\eta}(t)) = 0$$

- Complementarity conditions:

$$0 \leq \boldsymbol{\eta}(t), \qquad \boldsymbol{\eta}(t)^\top c(t,\hat{x}(t),\hat{u}(t)) = 0$$

Necessary Conditions of Discretized Optimal Control Problem

Theorem (Discrete Minimum Principle)

$(\hat{x}_h, \hat{u}_h)$ *local minimum of D-OCP. Then:*

- $\ell_0 \geq 0$, $(\ell_0, \boldsymbol{\kappa}, \boldsymbol{\lambda}, \zeta) \neq 0$
- *Discrete adjoint equations: For* $i = 0, \ldots, N-1$ *we have*

$$\frac{\boldsymbol{\lambda}_{i+1} - \boldsymbol{\lambda}_i}{h_i} = -\nabla_x \hat{\mathcal{H}}\left(t_i, \hat{x}_i, \hat{u}_i, \boldsymbol{\lambda}_{i+1}, \frac{\zeta_i}{h_i}\right)$$

- *Discrete transversality conditions:*

$$\boldsymbol{\lambda}_0 = -\nabla_{x_0}\left(\ell_0 \boldsymbol{\varphi} + \boldsymbol{\kappa}^\top \boldsymbol{\psi}\right), \qquad \boldsymbol{\lambda}_N = \nabla_{x_f}\left(\ell_0 \boldsymbol{\varphi} + \boldsymbol{\kappa}^\top \boldsymbol{\psi}\right)$$

- *Discrete stationarity conditions: For* $i = 0, \ldots, N-1$ *we have*

$$\nabla_u \hat{\mathcal{H}}\left(t_i, \hat{x}_i, \hat{u}_i, \boldsymbol{\lambda}_{i+1}, \frac{\zeta_i}{h_i}\right) = 0$$

- *Discrete complementarity conditions:*

$$0 \leq \zeta_i, \quad \zeta_i^\top c(t_i, \hat{x}_i, \hat{u}_i) = 0, \qquad i = 0, \ldots, N-1$$

Proof: evaluation of Fritz John conditions from nonlinear programming

Approximation of Adjoints – Comparison with Minimum Principle

Adjoints:

- Discrete:

$$
\begin{aligned}
\frac{\boldsymbol{\lambda}_{i+1} - \boldsymbol{\lambda}_i}{h_i} &= -\nabla_x \hat{\mathcal{H}}\left(t_i, \hat{x}_i, \hat{u}_i, \boldsymbol{\lambda}_{i+1}, \frac{\zeta_i}{h_i}\right), \\
\boldsymbol{\lambda}_N &= \nabla_{x_f}\left(\ell_0 \boldsymbol{\varphi} + \boldsymbol{\kappa}^\top \boldsymbol{\psi}\right)
\end{aligned}
$$

Approximation of Adjoints – Comparison with Minimum Principle

Adjoints:

- Discrete:

$$
\begin{aligned}
\frac{\boldsymbol{\lambda}_{i+1} - \boldsymbol{\lambda}_i}{h_i} &= -\nabla_x \hat{\mathcal{H}}\left(t_i, \hat{x}_i, \hat{u}_i, \boldsymbol{\lambda}_{i+1}, \frac{\zeta_i}{h_i}\right), \\
\boldsymbol{\lambda}_N &= \nabla_{x_f}\left(\ell_0 \boldsymbol{\varphi} + \boldsymbol{\kappa}^\top \boldsymbol{\psi}\right)
\end{aligned}
$$

- Continuous:

$$
\begin{aligned}
\boldsymbol{\lambda}'(t) &= -\nabla_x \hat{\mathcal{H}}(t, \hat{x}(t), \hat{u}(t), \boldsymbol{\lambda}(t), \boldsymbol{\eta}(t)) \\
\boldsymbol{\lambda}(t_f) &= \nabla_{x_f}\left(\ell_0 \boldsymbol{\varphi} + \boldsymbol{\sigma}^\top \boldsymbol{\psi}\right)
\end{aligned}
$$

Approximation of Adjoints – Comparison with Minimum Principle

Adjoints:

- Discrete:

$$\begin{aligned} \frac{\boldsymbol{\lambda}_{i+1} - \boldsymbol{\lambda}_i}{h_i} &= -\nabla_x \hat{\mathcal{H}}\left(t_i, \hat{x}_i, \hat{u}_i, \boldsymbol{\lambda}_{i+1}, \frac{\zeta_i}{h_i}\right), \\ \boldsymbol{\lambda}_N &= \nabla_{x_f}\left(\ell_0 \varphi + \boldsymbol{\kappa}^\top \psi\right) \end{aligned}$$

- Continuous:

$$\begin{aligned} \boldsymbol{\lambda}'(t) &= -\nabla_x \hat{\mathcal{H}}(t, \hat{x}(t), \hat{u}(t), \boldsymbol{\lambda}(t), \boldsymbol{\eta}(t)) \\ \boldsymbol{\lambda}(t_f) &= \nabla_{x_f}\left(\ell_0 \varphi + \boldsymbol{\sigma}^\top \psi\right) \end{aligned}$$

Interpretation: ($\boldsymbol{\kappa}$, $\boldsymbol{\lambda}_i$, ζ_i: multipliers of D-OCP)

$$\boldsymbol{\kappa} \approx \boldsymbol{\sigma}, \qquad \boldsymbol{\lambda}_i \approx \boldsymbol{\lambda}(t_i), \qquad \frac{\zeta_i}{h_i} \approx \boldsymbol{\eta}(t_i)$$

Contents

Introduction

Direct Discretization : First Discretize – Then Optimize
- Overview
- Full Discretization (Collocation, Direct Transcription)
- Numerical Solution of Discretized Problem
- Reduced Discretization (Shooting)
- Adjoint Estimation
- Convergence

Applications

Optimal Control Problem with Mixed Control-State Constraints

OCP

Minimize

$$\varphi(x(0), x(1))$$

with respect to

$$(x, u) \in W_{1,\infty}^{n_x}([0, 1]) \times L_{\infty}^{n_u}([0, 1])$$

subject to

ODE

$$x'(t) = f(x(t), u(t))$$

control-state constraints

$$c(x(t), u(t)) \leq 0$$

boundary conditions

$$\psi(x(0), x(1)) = 0$$

Discretized Optimal Control Problem

Grid sequence:

- $\{\mathbb{G}_N\}_{N\in\mathbb{N}}$, $\mathbb{G}_N := \{0 = t_0 < t_1 < \ldots < t_N = 1\}$, $h := \max_i h_i$, $h_i = t_i - t_{i-1}$

DOCP on $\mathbb{G}_N$

Minimize

$$\varphi(x_h(t_0), x_h(t_N))$$

with respect to

$$(x_h, u_h) \in W^{n_x}_{1,\infty,h}([0,1]) \times L^{n_u}_{\infty,h}([0,1])$$

subject to

$$\begin{aligned} f(x_h(t_i), u_h(t_i)) - x_h{}'(t_i) &= 0 && (i = 1, \ldots, N) \\ c(x_h(t_i), u_h(t_i)) &\leq 0 && (i = 1, \ldots, N) \\ \psi(x_h(t_0), x(t_N)) &= 0 \end{aligned}$$

$$x_h{}'(t_i) := \frac{x_h(t_i) - x_h(t_{i-1})}{h_i} \qquad \text{(backward difference, implicit Euler)}$$

Convergence Results – Overview (not complete)

ODEs, Index-1 DAEs, continuous solutions:

[A. L. Dontchev, W. W. Hager, and K. Malanowski, Error bounds for Euler approximation of a state and control constrained optimal control problem, Numerical Functional Analysis and Optimization, 21 (2000), 653–682]

[K. Malanowski, Ch. Büskens, and H. Maurer, Convergence of approximations to nonlinear optimal control problems, in "Mathematical Programming with Data Perturbations" (ed. A. Fiacco), volume 195, "Lecture Notes in Pure and Appl. Math.," Dekker, New York, (1997), 253–284.]

[B. Martens, M. Gerdts: Convergence Analysis of the Implicit Euler-discretization and Sufficient Conditions for Optimal Control Problems Subject to Index-one Differential-algebraic Equations, Set-Valued and Variational Analysis, 2018, DOI 10.1007/s11228-018-0471-x]

ODEs, higher order convergence:

[A. L. Dontchev, W. W. Hager, and V. M. Veliov, Second-Order Runge-Kutta Approximations in Control Constrained Optimal Control, SIAM Journal on Numerical Analysis, 38 (2000), pp. 202–226.]

[W. W. Hager, Runge-Kutta methods in optimal control and the transformed adjoint system, Numerische Mathematik, 87 (2000), pp. 247–282.]

ODE case, discontinuous controls

[W. Alt, R. Baier, M. Gerdts, and F. Lempio, Approximation of linear control problems with bang-bang solutions, Optimization: A Journal of Mathematical Programming and Operations Research, (2011), DOI 10.1080/02331934.2011.568619.]

[W. Alt, R. Baier, M. Gerdts, and F. Lempio, Error bounds for Euler approximation of linear-quadratic control problems with bang-bang solutions, Numerical Algebra, Control and Optimization, 2 (2012), 547–570.]

[M. Gerdts, M. Kunkel: Convergence Analysis of Euler Discretization of Control-State Constrained Optimal Control Problems with Controls of Bounded Variation, Journal of Industrial and Management Optimization, Vol. 10(1), pp. 311-336, 2014.]

Hamiltonian systems:

S. Ober-Blöbaum, O. Junge, J. E. Marsden: *Discrete mechanics and optimal control: an analysis*, ESAIM: Control, Optimisation and Calculus of Variations, ESAIM: COCV 17 (2011) 322–352 DOI: 10.1051/cocv/2010012

Discretization and Convergence – A Framework

OCP (infinite dimensional)

optimal solution: $(\hat{x}, \hat{u})$

Discretization and Convergence – A Framework

OCP (infinite dimensional)

optimal solution: $(\hat{x}, \hat{u})$

$\Downarrow$

$\boxed{\text{discretization}}$

$\Downarrow$

NLP (finite dimensional)

$$\begin{array}{ll} \text{Min} & J_h(x_h, u_h) \\ \text{s.t.} & G_h(x_h, u_h) \leq 0 \\ & H_h(x_h, u_h) = 0 \end{array}$$

optimal solution: $(\hat{x}_h, \hat{u}_h)$

Discretization and Convergence – A Framework

OCP (infinite dimensional)

optimal solution: $(\hat{x}, \hat{u})$

necessary $\Longrightarrow$

generalized equation

$$0 \in F(\hat{z}) + N_K(\hat{z})$$

$$\hat{z} = (\hat{x}, \hat{u}, \hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\eta}}, \hat{\boldsymbol{\sigma}})$$

$\Downarrow$

$\boxed{\text{discretization}}$

$\Downarrow$

NLP (finite dimensional)

$$\begin{array}{ll} \text{Min} & J_h(x_h, u_h) \\ \text{s.t.} & G_h(x_h, u_h) \leq 0 \\ & H_h(x_h, u_h) = 0 \end{array}$$

optimal solution: $(\hat{x}_h, \hat{u}_h)$

Discretization and Convergence – A Framework

OCP (infinite dimensional)

optimal solution: $(\hat{x}, \hat{u})$

necessary $\Longrightarrow$

generalized equation

$$0 \in F(\hat{z}) + N_K(\hat{z})$$
$$\hat{z} = (\hat{x}, \hat{u}, \hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\eta}}, \hat{\boldsymbol{\sigma}})$$

$\Downarrow$

discretization

$\Downarrow$

NLP (finite dimensional)

$$\begin{array}{ll} \text{Min} & J_h(x_h, u_h) \\ \text{s.t.} & G_h(x_h, u_h) \leq 0 \\ & H_h(x_h, u_h) = 0 \end{array}$$

optimal solution: $(\hat{x}_h, \hat{u}_h)$

necessary $\Longrightarrow$

generalized equation

$$0 \in F_h(\hat{z}_h) + N_{K_h}(\hat{z}_h)$$
$$\hat{z}_h = (\hat{x}_h, \hat{u}_h, \hat{\boldsymbol{\lambda}}_h, \hat{\boldsymbol{\eta}}_h, \hat{\boldsymbol{\sigma}}_h)$$

Discretization and Convergence – A Framework

OCP (infinite dimensional)

optimal solution: $(\hat{x}, \hat{u})$

necessary $\Longrightarrow$

generalized equation

$$0 \in F(\hat{z}) + N_K(\hat{z})$$
$$\hat{z} = (\hat{x}, \hat{u}, \hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\eta}}, \hat{\boldsymbol{\sigma}})$$

$\Downarrow$

discretization

$\Downarrow$

$\Uparrow$

Convergence ?

$$\lim_{h \downarrow 0} \|\hat{z} - \hat{z}_h\|_Z = 0$$

$\Uparrow$

NLP (finite dimensional)

$$\begin{array}{ll} \text{Min} & J_h(x_h, u_h) \\ \text{s.t.} & G_h(x_h, u_h) \leq 0 \\ & H_h(x_h, u_h) = 0 \end{array}$$

optimal solution: $(\hat{x}_h, \hat{u}_h)$

necessary $\Longrightarrow$

generalized equation

$$0 \in F_h(\hat{z}_h) + N_{K_h}(\hat{z}_h)$$
$$\hat{z}_h = (\hat{x}_h, \hat{u}_h, \hat{\boldsymbol{\lambda}}_h, \hat{\boldsymbol{\eta}}_h, \hat{\boldsymbol{\sigma}}_h)$$

Convergence

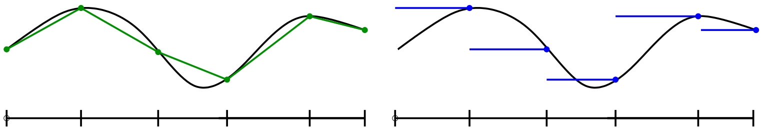
General concept

consistency + stability $\Longrightarrow$ convergence

Spaces and Restriction Operator

Restriction operator: $\Delta_h : Z \longrightarrow Z_h$ with, e.g.,

$$
\begin{aligned}
(\Delta_h x)(t) &= x(t_{i-1}) + x_h'(t_i)(t - t_{i-1}) && \text{for } t \in (t_{i-1}, t_i] \\
(\Delta_h u)(t) &= u(t_i) && \text{for } t \in (t_{i-1}, t_i]
\end{aligned}
$$



Recall: $Z_h \subset Z$, same norms

Convergence

(I) Smoothness: There exists $C_F > 0$ independent of h such that

$$\left\| F_h'\left(z_h^1\right) - F_h'\left(z_h^2\right) \right\|_{\mathfrak{L}(Z_h,\Omega_h)} \leq C_F \left\| z_h^1 - z_h^2 \right\|_Z \qquad \forall z_h^1, z_h^2 \in Z_h$$

Convergence

(I) Smoothness: There exists $C_F > 0$ independent of h such that

$$\left\| F_h'\left(z_h^1\right) - F_h'\left(z_h^2\right)\right\|_{\mathfrak{L}(Z_h,\Omega_h)} \leq C_F \left\| z_h^1 - z_h^2 \right\|_Z \qquad \forall z_h^1, z_h^2 \in Z_h$$

(II) Consistency: For all $h > 0$ there exist $\hat{\omega}_h \in \Omega_h$ with

$$0 \in F_h\left(\Delta_h \hat{z}\right) + \hat{\omega}_h + N_{K_h}\left(\Delta_h \hat{z}\right)$$

and for $h \to 0$ we have

$$\|\Delta_h \hat{z} - \hat{z}\|_Z \to 0, \quad \|\hat{\omega}_h\|_\Omega \to 0$$

Convergence

(III) Uniform strong regularity (Stability): For $h > 0$ and δ sufficiently small there exists a unique solution $z_h(\delta)$ of

$$\delta \in F_h(\Delta_h \hat{z}) + \hat{\omega}_h + F_h'(\Delta_h \hat{z})(z_h - \Delta_h \hat{z}) + N_{K_h}(z_h),$$

and for all perturbations δ_1, δ_2 sufficiently small the inequality

$$\|z_h(\delta_1) - z_h(\delta_2)\|_Z \leq \ell \|\delta_1 - \delta_2\|_\Omega$$

is satisfied for some $\ell > 0$ independent of h.

Convergence

Then:

Convergence Theorem

With **(I)-(III)** there exists a solution $\hat{z}_h$ of $0 \in F_h(z_h) + N_{K_h}(z_h)$ and

$$\|\hat{z}_h - \hat{z}\|_Z \leq C \left(\underbrace{\|\hat{\omega}_h\|_{Z^*}}_{\text{consistency error}} + \underbrace{\|\Delta_h \hat{z} - \hat{z}\|_Z}_{\text{interpolation error}} \right)$$

[K. Malanowski, Ch. Büskens, and H. Maurer, Convergence of approximations to nonlinear optimal control problems, in “Mathematical Programming with Data Perturbations” (ed. A. Fiacco), volume 195, “Lecture Notes in Pure and Appl. Math.,” Dekker, New York, (1997), 253–284.]

[S. M. Robinson: *Strongly regular generalized equations*, Math. Oper. Research, 5, pp. 43-62, 1980]

Standing Assumptions

Throughout we assume:

Assumption **(A1)**

$(\hat{x}, \hat{u}) \in W_{2,\infty}^{n_x}([0,1]) \times W_{1,\infty}^{n_u}([0,1])$ is a local (weak) minimizer.

Note the smoothness: $\hat{u}$ is assumed to be Lipschitz.

Standing Assumptions

Throughout we assume:

Assumption **(A1)**

$(\hat{x}, \hat{u}) \in W^{n_x}_{2,\infty}([0,1]) \times W^{n_u}_{1,\infty}([0,1])$ is a local (weak) minimizer.

Note the smoothness: $\hat{u}$ is assumed to be Lipschitz.

Assumption **(A2)**

sufficient smoothness of functions φ, f, c

Assumptions

Assumption (**A3**): regularity condition

$$\exists \alpha, \beta > 0 \; \forall d, t \in [0,1] : \left\| C^{\alpha}(t)^{\top} d \right\| \geq \beta \|d\|$$

with

$$C^{\alpha}(t) := \left(c'_{j,u}[t] \right)_{j \in J^{\alpha}(t)}$$

and

$$J^{\alpha}(t) := \{ j \mid c_j[t] \geq -\alpha \}$$

Assumptions

Assumption (**A4**): controllability

For all e there exists (x, u) such that

$$
\begin{aligned}
x'(t) &= A(t)x(t) + B(t)u(t) \\
0 &= D^{\alpha}(t)x(t) + C^{\alpha}(t)u(t) \\
e &= \psi'_{x_0}x(0) + \psi'_{x_1}x(1)
\end{aligned}
$$

with

$$
A(t) := f'_x[t], \quad B(t) := f'_u[t], \quad D^{\alpha}(t) := \left(c'_{j,x}[t]\right)_{j \in J^{\alpha}(t)}, \quad C^{\alpha}(t) := \left(c'_{j,u}[t]\right)_{j \in J^{\alpha}(t)}
$$

Assumptions

Assumptions (**A5**): Coercivity

There exist $\nu, \gamma > 0$ such that

$$\begin{pmatrix} x(0) \\ x(1) \end{pmatrix}^{\top} \nabla^2 \left(\varphi(x(0), x(1)) + \sigma^{\top} \psi(x(0), x(1)) \right) \begin{pmatrix} x(0) \\ x(1) \end{pmatrix}$$

$$+ \int_0^1 \begin{pmatrix} x(t) \\ u(t) \end{pmatrix}^{\top} \nabla^2_{(x,u)} H[t] \begin{pmatrix} x(t) \\ u(t) \end{pmatrix} dt \geq \gamma \, \| (x, u) \|^2_{X_2 \times U_2}$$

for all $(x, u) \in X_2 \times U_2$ with

$$\begin{aligned} 0 &= x'(t) - A(t)\, x(t) - B(t)\, u(t) \\ 0 &= \psi'_{x_0}\, x(0) + \psi'_{x_1}\, x(1) \\ 0 &= \check{D}^{\nu}(t)\, x(t) + \check{C}^{\nu}(t)\, u(t) \end{aligned}$$

and

$$\left(\check{D}^{\nu}(t), \check{C}^{\nu}(t) \right) := \left(c'_{j,x}[t], c'_{j,u}[t] \right)_{j \in J_+^{\nu}(t)}, \quad J_+^{\nu}(t) := \{ j \in J^0(t) \mid \eta_j(t) > \nu \}$$

Implications

Assumptions (**A1**)– (**A4**) imply the following:

- existence of Lagrange multipliers such that KKT conditions hold with augmented Hamiltonian
$$H(x, u, \boldsymbol{\lambda}, \boldsymbol{\eta}) = \boldsymbol{\lambda}^\top f(x, u) + \boldsymbol{\eta}^\top c(x, u)$$
- smoothness: $\boldsymbol{\lambda} \in W^{n_x}_{2,\infty}([0, 1])$, $\boldsymbol{\eta} \in W^{n_c}_{1,\infty}([0, 1])$

Assumptions (**A1**)– (**A5**) imply the following:

- the discrete problem satisfies an analog coercivity condition
- the Legendre-Clebsch condition holds (continuous & discrete)
- the discrete problem has a locally unique solution
- discrete Legendre-Clebsch condition yields uniform strong regularity in the L^∞-norm (exploiting a parametric sensitivity analysis)

Final Result

Finally:

Convergence Theorem

If assumptions **(A1)** - **(A5)** are satisfied then for $h > 0$ sufficiently small there exists a locally unique KKT point $\hat{z}_h$ of **(DOCP)** and

$$\|\hat{z}_h - \Delta_h \hat{z}\|_Z \leq \ell \, h,$$

where ℓ is independent of h.

Details:

[1] Björn Martens:
Necessary Conditions, Sufficient Conditions, and Convergence Analysis for Optimal Control Problems with Differential-Algebraic Equations,
PhD thesis, Institute of Applied Mathematics and Scientific Computing, Universität der Bundeswehr, 2019.
https://athene-forschung.unibw.de/130232

Contents

Introduction

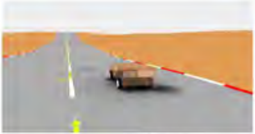
Direct Discretization : First Discretize – Then Optimize
- Overview
- Full Discretization (Collocation, Direct Transcription)
- Numerical Solution of Discretized Problem
- Reduced Discretization (Shooting)
- Adjoint Estimation
- Convergence

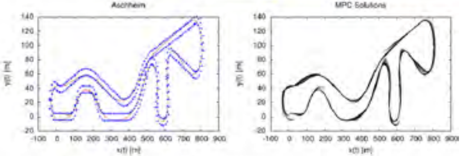
Applications

Research @ Engineering Mathematics

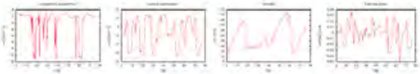
Application: Automatic Driving

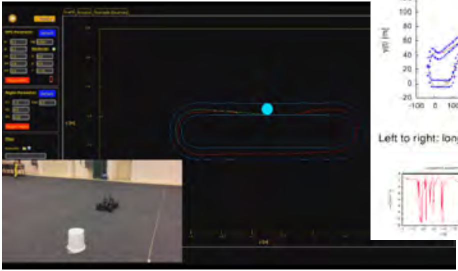
- Modelling of an “optimal” driver (time minimal, fuel efficient)
- Consideration of track bounds and obstacles
- Online optimization





Left to right: longitudinal acceleration, lateral acceleration, velocity, slip angle



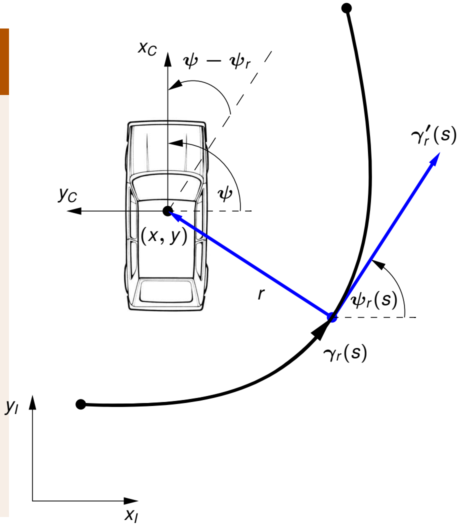


Nonlinear Kinematic Model

Motion in (s,r)-system along a reference curve

Given:

- reference curve $\gamma_r = (x_r, y_r)^\top$
- curvature κ_r



Nonlinear Kinematic Model

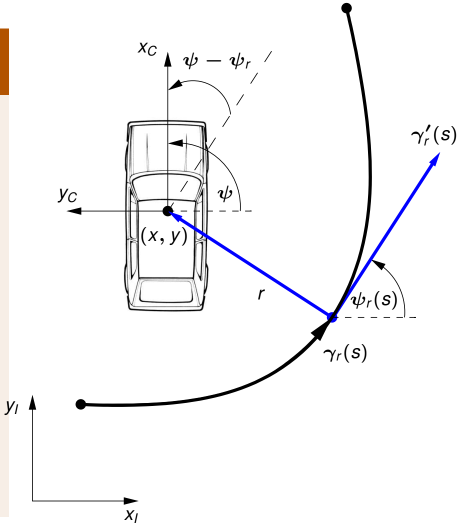
Motion in (s,r)-system along a reference curve

Given:

- reference curve $\gamma_r = (x_r, y_r)^\top$
- curvature κ_r

Motion in moving reference system aligned with γ_r:

$$\begin{aligned}
s' &= \frac{v\cos(\psi - \psi_r)}{1 - r \cdot \kappa_r(s)} \\
r' &= v\sin(\psi - \psi_r) \\
\psi' &= v \cdot \kappa \\
\kappa' &= u \\
\psi_r' &= \kappa_r(s) \cdot s'
\end{aligned}$$



Decoupling

Decoupling ...

Decoupling

Decoupling ...

Path Planning (yields parametrized curve w.r.t. arclength)

Minimize

$$-\alpha_1 s(L) + \alpha_2 \int_0^L \kappa(\ell)^2 d\ell + \alpha_3 \int_0^L u(\ell)^2 d\ell$$

s.t. dynamics with $v(t) \equiv 1$, initial conditions, and control/state constraints

$$(r, u, \kappa) \in [-r_{max}, r_{max}] \times [-u_{max}, u_{max}] \times [-\kappa_{max}, \kappa_{max}]$$

Decoupling

Decoupling ...

Path Planning (yields parametrized curve w.r.t. arclength)

Minimize

$$-\alpha_1 s(L) + \alpha_2 \int_0^L \kappa(\ell)^2 d\ell + \alpha_3 \int_0^L u(\ell)^2 d\ell$$

s.t. dynamics with $v(t) \equiv 1$, initial conditions, and control/state constraints

$$(r, u, \kappa) \in [-r_{max}, r_{max}] \times [-u_{max}, u_{max}] \times [-\kappa_{max}, \kappa_{max}]$$

and

Velocity Profile Generation

Find velocity profile $v(\ell)$ for $\ell \in [0, L]$ on computed path.

Decoupling

Decoupling ...

Path Planning (yields parametrized curve w.r.t. arclength)

Minimize

$$-\alpha_1 s(L) + \alpha_2 \int_0^L \kappa(\ell)^2 d\ell + \alpha_3 \int_0^L u(\ell)^2 d\ell$$

s.t. dynamics with $v(t) \equiv 1$, initial conditions, and control/state constraints

$$(r, u, \kappa) \in [-r_{max}, r_{max}] \times [-u_{max}, u_{max}] \times [-\kappa_{max}, \kappa_{max}]$$

and

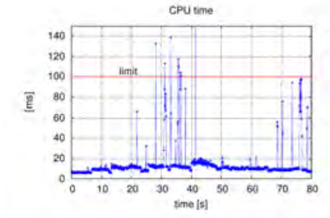
Velocity Profile Generation

Find velocity profile $v(\ell)$ for $\ell \in [0, L]$ on computed path.

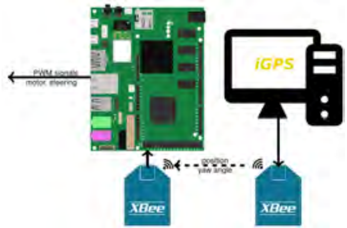
... increases robustness and flexibility.

Realization on Cars

CPU times: ($N = 18$)



Control architecture:



Path Planning of a UAV

Motion in a flight corridor

- reference ground curve

$$\gamma_r(s) = \begin{pmatrix} x_r(s) \\ y_r(s) \end{pmatrix},$$

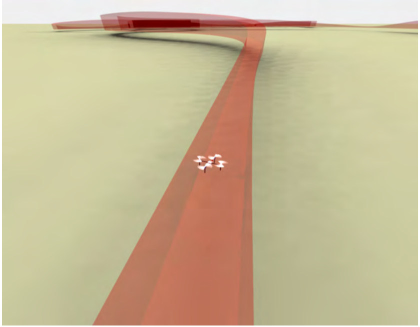
curvature κ_r, curve parameter s

- altitude bounds

$$z_{min}(s) \leq z(s) \leq z_{max}(s)$$

- width bounds

$$r_{min}(s) \leq r(s) \leq r_{max}(s)$$



[M. Burger, M. Gerdts: DAE Aspects in Vehicle Dynamics and Mobile Robotics, in Applications of Differential-Algebraic Equations: Examples and Benchmarks, Differential-Algebraic Equations Forum DAE-F, Eds. S. Campbell, A. Ilchmann, V. Mehrmann, T. Reis, Springer, pp. 37–80, 2019.]

Path Planning of a UAV

Motion in a flight corridor

$$
\begin{aligned}
s' &= \frac{v_{xy} \cdot \cos(\psi - \psi_r)}{1 - r \cdot \kappa_r(s)} \\
r' &= v_{xy} \cdot \sin(\psi - \psi_r) \\
z' &= v_z \\
m \cdot v_x' &= u_1 \cdot \cos\phi \cdot \sin\kappa - D_x \\
m \cdot v_y' &= -u_1 \cdot \sin\phi - D_y \\
m \cdot v_z' &= u_1 \cdot \cos(\phi) \cdot \cos(\kappa) - m \cdot g - D_z \\
\phi' &= \frac{u_2 - \phi}{\delta} \\
\kappa' &= \frac{u_3 - \kappa}{\delta}
\end{aligned}
$$

Notation:

- (s, r, z)=position in curvilinear coordinates
- u_1 = thrust
- u_2 = commanded roll angle
- u_3 = commanded pitch angle
- $v_{xy} = \sqrt{v_x^2 + v_y^2}$,
  $\psi = \arctan(v_y / v_x)$
- δ = delay factor

Path Planning of a UAV

Objective: (to be minimized)

$$\underbrace{\int_0^L \frac{1}{v(\ell)}\, d\ell}_{\text{flight time}} + \underbrace{\int_0^L u_1(\ell)^2 + u_2(\ell)^2 + u_3(\ell)^2\, d\ell}_{\text{control effort}}$$

State and control constraints:

$$z_{min}(s(\ell)) \leq z(\ell) \leq z_{max}(s(\ell)) \qquad \text{(altitude)}$$

$$r_{min}(s(\ell)) \leq r(\ell) \leq r_{max}(s(\ell)) \qquad \text{(offset)}$$

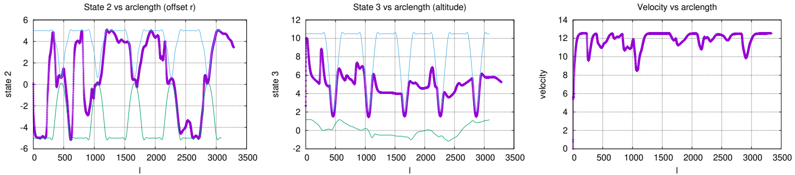
$$v_{min} \leq v \leq v_{max} \qquad \text{(velocity)}$$

$$|\phi| \leq \phi_{max}, \; |\kappa| \leq \kappa_{max} \qquad \text{(angles)}$$

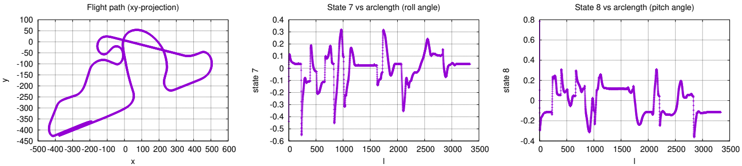
$$u_i \in [u_{i,min}, u_{i,max}], \; i = 1, 2, 3 \qquad \text{(controls)}$$

NMPC Results Quadrocopter

States: (offset r, altitude, velocity)

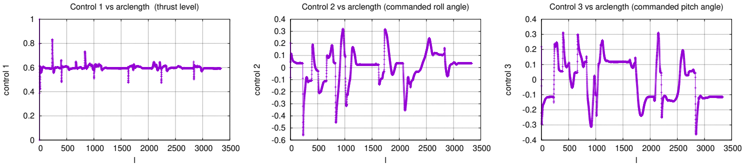


States: (xy-path, roll and pitch angle)



NMPC Results Quadrocopter

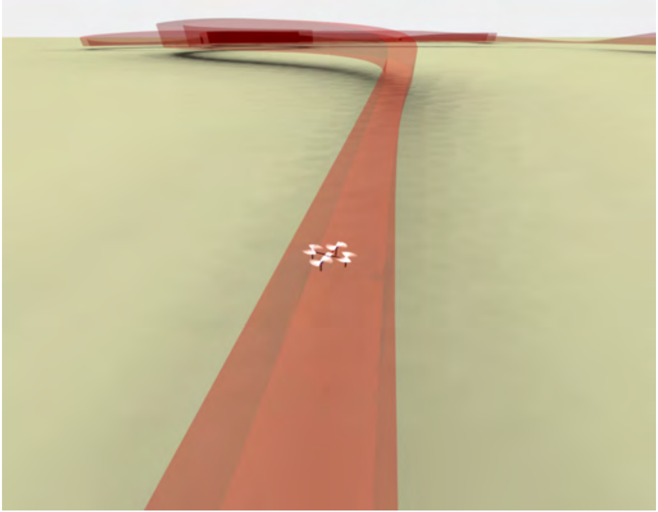
Controls: (thrust level, commanded roll and pitch)



$m = 3$ [kg], $\delta = 0.1$, $v_{max} = 15$ [m/s], $\phi_{max} = \kappa_{max} = 45^\circ$, $L = 20$ [m], $N = 30$, $T_{max} = 50$ [N]

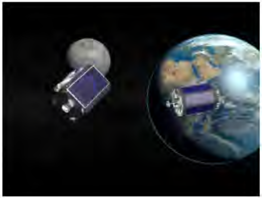
- total flight time: 284.59 [s]
- CPU time: 460.015 [s] for 5001 OCPs
- CPU time per OCP: 0.09 [s]

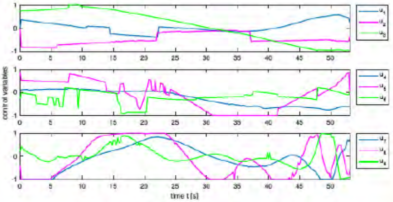
NMPC Results Quadrocopter



Example: Docking Maneuver

- Aim: Compute fuel efficient docking maneuvers w/o robotic manipulators to tumbling targets
- Multiple phases: Synchronization, docking, stabilization, transfer
- Space debris removal, landing on moving platforms





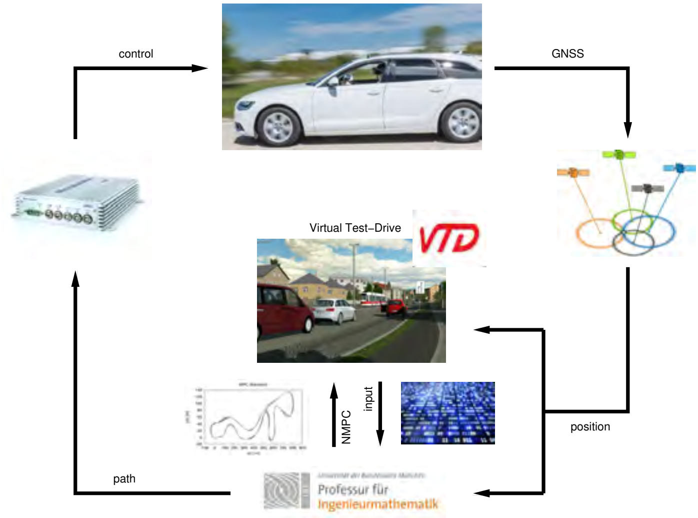
landing – docking, phase 1 – docking, phase 2

[M. Kreher: Optimal Docking Maneuvers for Space Debris Removal, Master thesis, UniBwM, 2017]

Automated Interconnected Vehicle-in-the-Loop (AN-VIL) @ Engineering Mathematics

- platform combining
  virtual reality & real driving & automated driving
- two experimental Audi A6 equipped with VTD, IMU, D-GPS
- versatile and safe tool in automated driving, cooperative driving, and human-machine interaction

Concept


control
GNSS
Virtual Test–Drive
VTD
NMPC
input
position
path
Professur für
Ingenieurmathematik

Testing Area @ UniBw M

Testing Area @ UniBw M

Research

Automated Driving

Cooperative Driving

Human-Machine-Interaction

- path planning and tracking
- MPC / online optimization
- obstacle avoidance

- distributed control
- hierarchies vs Nash equilibria
- obstacle avoidance

- many user studies performed by Prof. Färber and Prof. Nitsch, LRT-11
- identification of comfort criteria

Vision

- driving in the same virtual scenario ...
  ⇝ virtually dangerous scenarios possible
- ... but physically separated
  ⇝ physically safe at all times
- interactions
  human – human
  human – automated (real/virtual)
  automated – automated (real/virtual)

Extensions

Not discussed ...

- mixed-integer optimal control
  $\rightsquigarrow$ lectures by Christian Kirches, Sven Leyffer
- model-predictive control
  $\rightsquigarrow$ lectures by Karl Worthmann
- optimal control subject to differential-algebraic equations (DAEs)
- ... and many other topics in optimal control

Some Resources

Optimal control software:

- `CasADI`, `ACADO`: M. Diehl et al.; http://casadi.org; http://sourceforge.net/p/acado/
- `NUDOCCCS`: C. Büskens, University of Bremen
- `SOCS`: J. Betts, The Boeing Company, Seattle; http://www.boeing.com/boeing/phantom/socs/
- `DIRCOL`: O. von Stryk, TU Darmstadt; http://www.sim.informatik.tu-darmstadt.de/res/sw/dircol
- `MUSCOD-II`: H.G. Bock et al., IWR Heidelberg; http://www.iwr.uni-heidelberg.de/∼agbock/RESEARCH/muscod.php
- `MISER`: K.L. Teo et al., Curtin University, Perth; http://school.maths.uwa.edu.au/ les/miser/
- `PSOPT`: http://www.psopt.org/
- `FALCON.m`: https://www.fsd.lrg.tum.de/software/falcon-m/
- `GPOPS-II`: http://www.gpops2.com/
- ...

Optimization software:

- `WORHP` (sparse large-scale problems): C. Büskens/M. Gerdts, https://www.worhp.de
- `NPSOL` (dense problems), `SNOPT` (sparse large-scale problems): Stanford Business Software; http://www.sbsi-sol-optimize.com
- `KNITRO` (sparse large-scale problems): Ziena Optimization; http://www.ziena.com/knitro.htm
- `IPOPT` (sparse large-scale problems): A. Wächter: https://projects.coin-or.org/Ipopt
- `filterSQP`: R. Fletcher, S. Leyffer; http://www.mcs.anl.gov/ leyffer/solvers.html
- `ooQP`: M. Gertz, S. Wright; http://pages.cs.wisc.edu/ swright/ooqp/
- `qpOASES`: H.J. Ferreau, A. Potschka, C. Kirches; http://homes.esat.kuleuven.be/ optec/software/qpOASES/
- `OSQP`: B. Stellato, G. Banjac, P. Goulart, A. Bemporad, S. Boyd; https://osqp.org/
- ...

Links:

- Decision Tree for Optimization Software; http://plato.la.asu.edu/guide.html
- `CUTEr`: large collection of optimization test problems; http://www.cuter.rl.ac.uk/
- `COPS`: large-scale optimization test problems; http://www.mcs.anl.gov/∼more/cops/
- `MINTOC`: testcases for mixed-integer optimal control; http://mintoc.de/
- ...

Acknowledgement

The research was partly funded by:

Bundesministerium für Bildung und Forschung

DFG

AFRL
AFOSR

MUNICH AEROSPACE

Gefördert durch dtec.bw

(SeRANIS, MORE, EMERGENCY-VRD, MissionLab)

https://dtecbw.de

Thanks for your Attention!

Questions?

Further information:

matthias.gerdts@unibw.de
www.unibw.de/ingmathe
www.optimal-control.de

Fotos: http://de.wikipedia.org/wiki/München
Magnus Manske (Panorama), Luidger (Theatinerkirche), Kurmis (Chin. Turm), Arad Mojtahedi (Olympiapark), Max-k (Deutsches Museum), Oliver Raupach (Friedensengel), Andreas Praefcke (Nationaltheater)