Master thesis

Development of a Multi-phase Optimal Control Software for Aerospace applications (MPOPT)

Devakumar Thammisetty

Laboratoire d'automatique
École Polytechnique Fédérale de Lausanne – EPFL

Local Supervisor	Petr Listov, Doctoral assistant, EPFL
Supervisor	Prof. Colin Jones, EPFL
External examiner	Dr. UP Rajeev, VSSC, ISRO

July 16, 2020

Contents

Contents

Introduction

Optimal control problem (OCP) solvers

Compute control law satisfying the system dynamics and operation constraints while optimizing a given performance index.

Standard OCP in Bolza form

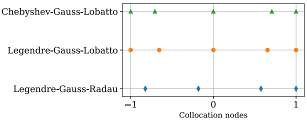
$$
\begin{aligned}
\min_{x,u,t_0,t_f,s} \quad & J = M(x_0, t_0, x_f, t_f, s) + \int_0^{t_f} L(x, u, t, s)dt \\
\text{subject to} \quad & \dot{x} = f(x, u, t, s) \\
& g(x, u, t, s) \leq 0 \\
& h(x_0, t_0, x_f, t_f, s) = 0
\end{aligned}
$$

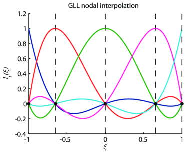
Methods to solve OCPs

	Direct methods	**Indirect methods**
	Finite dimensional	Infinite dimensions
Principle	Convert to NLP using transcription methods • **Collocation** • Single, Multiple shooting	Convert to Two-point boundary value problem using Pontryagin maximum principle
Pros	Easy to solve non-linear and complex OCPs	Possible to derive closed form analytical solutions for certain OCP formulations
Cons	Difficult to transcribe accurately	Difficult to solve complex OCPs, sensitive to initial guess

Pseudo-spectral collocation

- Satisfy the system dynamics and constraints exactly at predefined collocation nodes by approximating states and controls using interpolation polynomials
- Choice of collocation nodes $\rightarrow$ Jacobi polynomials (Spectral property)





$$\mathbf{x}^s(\tau) = \sum_{i=0}^{d_s^p} \mathbf{x}_i \phi_i(\tau); \qquad \mathbf{u}(\tau) = \sum_{i=0}^{d_s^p} \mathbf{u}_i \phi_i(\tau)$$

$$\phi_i(\tau_j) = \delta_{ij}; \qquad \phi_i(\tau) = \prod_{j=0;j\neq i}^{d_s^p} \frac{\tau - \tau_j}{\tau_i - \tau_j}$$

Multi-phase OCP solvers

- Number of software available. e.g. GPOPS-II, PSOPT, SOCS, DIDO.

Sl.no	**Solver**	**Remarks** [Source: Opty]
1	GPOPS -II	2014, MATLAB, Commercial, Automatic Differentiation, Pseudospectral, Grid refinement, IPOPT, SNOPT
2	PSOPT	2010, C++, Open source, Automatic Differentiation, Pseudospectral, Grid refinement, IPOPT, non-intuitive interface
3	SOCS	2010, Fortran, Commercial, Finite differences, Euler/RK, built-in NLP solver
4	DIDO	2002, MATLAB, Commercial, Analytic Differentiation, Pseudospectral, IPOPT, SNOPT

- Need fast, extendable, easy to use, open-source solver

Contents

Transcription : Continuous time OCP $\rightarrow$ NLP

- Collocation: Selection of collocation nodes, Approximation of derivative and integral operators
- Discretize system dynamics, operation constraints and objective interms of optimization variables

Multi-phase continuous time OCP → NLP

$$\min_{\mathbf{x}^p,\mathbf{u}^p,t_0^p,t_f^p,\mathbf{a}^p} \quad \sum_{p=1}^{N_p}\left[\mathcal{M}^{(p)}\left(\mathbf{x}^p(t_0^p),\mathbf{x}^p(t_f^p),t_0^p,t_f^p,\mathbf{a}^p\right)+\int_{t_0^p}^{t_f^p}\mathcal{L}^{(p)}\left(\mathbf{x}^p,\mathbf{u}^p,t^p,\mathbf{a}^p\right)dt\right]$$

$$\begin{aligned}
\text{subject to} \quad & \dot{\mathbf{x}}^p = \mathbf{f}^{(p)}(\mathbf{x}^p,\mathbf{u}^p,t^p,\mathbf{a}^p)\\
& \mathbf{g}^{(p)}(\mathbf{x}^p,\mathbf{u}^p,t^p,\mathbf{a}^p) \le 0\\
& \mathbf{h}^{(p)}(\mathbf{x}^p(t_0^p),\mathbf{x}^p(t_f^p),t_0^p,t_f^p,\mathbf{a}^p) = 0\\
& \underline{\mathbf{e}} \le \mathbf{e}(\mathbf{x}^1(t_0^1),\mathbf{x}^1(t_f^1),t_0^1,t_f^1,\mathbf{a}^1,\dots \mathbf{x}^{N_p}(t_0^{N_p}),\mathbf{x}^{N_p}(t_f^{N_p}),t_0^{N_p},t_f^{N_p},\mathbf{a}^{N_p}) \le \overline{\mathbf{e}}\\
& \mathbf{x}^p \in \mathcal{X}^p; \quad \mathbf{u}^p \in \mathcal{U}^p; \quad t_0^p \in \mathcal{T}_0^p; \quad t_f^p \in \mathcal{T}_f^p; \quad \mathbf{a}^p \in \mathcal{A}^p;
\end{aligned}$$

↓

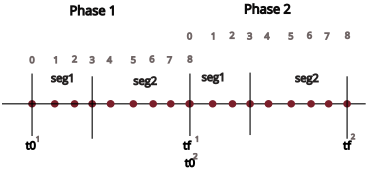
$$\min_{\mathcal{X},\mathcal{U},\mathcal{T},\mathcal{A},\mathcal{P}} \quad \sum_{p=1}^{N_p}\left[\mathbb{M}^p + \mathbf{W}^{\mathbf{p}\,T}\mathbf{Q}^p\right]$$

$$\begin{aligned}
\text{s. t.} \quad & 0 = \mathbf{D}^p\mathbf{X}^p - \mathbf{F}^p = 0 && \forall p = 1,\dots,N_p\\
& -\infty \le \mathbf{G}^p \le 0 && \forall p = 1,\dots,N_p\\
& 0 = \mathbf{H}^p = 0 && \forall p = 1,\dots,N_p\\
& \underline{\mathbf{E}}^l \le \mathbf{E}^l \le \overline{\mathbf{E}}^l && \forall\, l \in \mathcal{P}\\
& \mathbf{X}^p \in \mathcal{X}^p; \quad \mathbf{U}^p \in \mathcal{U}^p; && \forall p = 1,\dots,N_p\\
& t_0^p \in \mathcal{T}_0^p; \quad t_{N_c^p}^p \in \mathcal{T}_f^p; \quad \mathbf{a}^p \in \mathcal{A}^p; && \forall p = 1,\dots,N_p
\end{aligned}$$

Continuous time OCP → NLP : States

$$\min_{\mathcal{X},\mathcal{U},\mathcal{T},\mathcal{A},\mathcal{P}} \quad \sum_{p=1}^{N_p} \left[\mathbb{M}^p + \mathbf{W^p}^T \mathbf{Q}^p \right]$$

$$\begin{aligned}
\text{s. t.} \quad 0 &= \mathbf{D}^p \mathbf{X}^p - \mathbf{F}^p && = 0 \\
-\infty &\le \mathbf{G}^p && \le 0 \\
0 &= \mathbf{H}^p && = 0 \\
\underline{\mathbf{E}} &\le \mathbf{E} && \le \overline{\mathbf{E}} \\
& \mathbf{X}^p \in \mathcal{X}^p; && \mathbf{U}^p \in \mathcal{U}^p; \\
& t_0^p \in \mathcal{T}_0^p; && t_{N_c^p}^p \in \mathcal{T}_f^p; \\
& \mathbf{a}^p \in \mathcal{A}^p;
\end{aligned}$$



Degree of polynomial in segment (s) $= d^s$

$$\text{Define } N_c^p = \sum_{j=1}^{N_s} d^j$$

$$\mathbf{X}^p = \begin{bmatrix} x_0^1 & \dots & x_0^{n_x} \\ x_1^1 & \dots & x_1^{n_x} \\ \vdots & \ddots & \vdots \\ x_{N_c^p}^1 & \dots & x_{N_c^p}^{n_x} \end{bmatrix}^p$$

Continuous time OCP $\rightarrow$ NLP : Controls

$$\min_{\mathcal{X},\mathcal{U},\mathcal{T},\mathcal{A},\mathcal{P}} \quad \sum_{p=1}^{N_p} \left[\mathbb{M}^p + \mathbf{W^p}^T \mathbf{Q}^p \right]$$

$$\begin{aligned} \text{s. t.} \qquad 0 &= \mathbf{D}^p \mathbf{X}^p - \mathbf{F}^p && = 0 \\ -\infty &\leq \mathbf{G}^p && \leq 0 \\ 0 &= \mathbf{H}^p && = 0 \\ \underline{\mathbf{E}} &\leq \mathbf{E} && \leq \overline{\mathbf{E}} \\ & \mathbf{X}^p \in \mathcal{X}^p; && \mathbf{U}^p \in \mathcal{U}^p; \\ & t_0^p \in \mathcal{T}_0^p; && t_{N_c^p}^p \in \mathcal{T}_f^p; \\ & \mathbf{a}^p \in \mathcal{A}^p; \end{aligned}$$

$$\mathbf{U}^p = \begin{bmatrix} u_0^1 & \dots & u_0^{n_u} \\ u_1^1 & \dots & u_1^{n_u} \\ \vdots & \ddots & \vdots \\ u_{N_c^p}^1 & \dots & u_{N_c^p}^{n_u} \end{bmatrix}^p$$

Continuous time OCP → NLP : Time and Phase links

$$
\begin{aligned}
\min_{\mathcal{X},\mathcal{U},\mathcal{T},\mathcal{A},\mathcal{P}} \quad & \sum_{p=1}^{N_p} \left[\mathbb{M}^p + \mathbf{W^p}^T \mathbf{Q}^p \right] \\
\text{s. t.} \quad & 0 = \mathbf{D}^p \mathbf{X}^p - \mathbf{F}^p \quad = 0 \\
& -\infty \leq \mathbf{G}^p \quad \leq 0 \\
& 0 = \mathbf{H}^p \quad = 0 \\
& \underline{\mathbf{E}} \leq \mathbf{E} \quad \leq \overline{\mathbf{E}} \\
& \mathbf{X}^p \in \mathcal{X}^p; \quad \mathbf{U}^p \in \mathcal{U}^p; \\
& t_0^p \in \mathcal{T}_0^p; \quad t_{N_c^p}^p \in \mathcal{T}_f^p; \\
& \mathbf{a}^p \in \mathcal{A}^p;
\end{aligned}
$$

$$\mathcal{T} = \left\{ \left(t_0^p, t_{N_c^p}^p \right); \quad \forall p = 1 \dots N_p \right\}$$

$$\mathcal{A} = \left\{ a^p; \quad \forall p = 1 \dots N_p \right\}$$

$$\mathcal{P} = \left\{ (p_i, p_j) \mid p_i, p_j \in [1 \dots N_p]; p_i < p_j \right\}$$

Continuous time OCP $\rightarrow$ NLP : Mayer term

$$
\begin{aligned}
\min_{\mathcal{X},\mathcal{U},\mathcal{T},\mathcal{A},\mathcal{P}} \quad & \sum_{p=1}^{N_p} \left[\mathbb{M}^p + \mathbf{W^p}^T \mathbf{Q}^p\right] \\
\text{s. t.} \quad & 0 = \mathbf{D}^p \mathbf{X}^p - \mathbf{F}^p = 0 \\
& -\infty \leq \mathbf{G}^p \leq 0 \\
& 0 = \mathbf{H}^p = 0 \\
& \underline{\mathbf{E}} \leq \mathbf{E} \leq \overline{\mathbf{E}} \\
& \mathbf{X}^p \in \mathcal{X}^p; \quad \mathbf{U}^p \in \mathcal{U}^p; \\
& t_0^p \in \mathcal{T}_0^p; \quad t_{N_c^p}^p \in \mathcal{T}_f^p; \\
& \mathbf{a}^p \in \mathcal{A}^p;
\end{aligned}
$$

$$
M^p = \mathcal{M}^{(p)}\left(\mathbf{x}_0^p, \mathbf{x}_{N_c^p}^p, t_0^p, t_{N_c^p}^p, \mathbf{a}^p\right)
$$

Continuous time OCP $\rightarrow$ NLP : Segment width fraction

$$\min_{\mathcal{X},\mathcal{U},\mathcal{T},\mathcal{A},\mathcal{P}} \quad \sum_{p=1}^{N_p} \left[\mathbb{M}^p + \mathbf{W^p}^T \mathbf{Q}^p \right]$$

$$\begin{aligned}
\text{s. t.} \quad 0 &= \mathbf{D}^p \mathbf{X}^p - \mathbf{F}^p &= 0 \\
-\infty &\leq \mathbf{G}^p &\leq 0 \\
0 &= \mathbf{H}^p &= 0 \\
\underline{\mathbf{E}} &\leq \mathbf{E} &\leq \overline{\mathbf{E}} \\
& \mathbf{X}^p \in \mathcal{X}^p; \quad \mathbf{U}^p \in \mathcal{U}^p; \\
& t_0^p \in \mathcal{T}_0^p; \quad t_{N_c^p}^p \in \mathcal{T}_f^p; \\
& \mathbf{a}^p \in \mathcal{A}^p;
\end{aligned}$$

$$\sum_{s=1}^{N_s^p} h_s^p = 1; \qquad 0 < h_s^p \leq 1$$

$$\text{Segment width} = \left(t_{N_c^p}^p - t_0^p \right) \times h_s^p$$

$$t_{\text{start}}^s = \left(t_{N_c^p}^p - t_0^p \right) \times \sum_{i=1}^{s-1} h_i^p$$

$$t_{\text{end}}^s = \left(t_{N_c^p}^p - t_0^p \right) \times \sum_{i=1}^{s} h_i^p$$

$$t^p(\tau) = \frac{t_{\text{end}}^s - t_{\text{start}}^s}{2} \tau + \frac{t_{\text{end}}^s + t_{\text{start}}^s}{2}$$

Continuous time OCP $\rightarrow$ NLP : Quadrature

$$
\begin{aligned}
\min_{\mathcal{X},\mathcal{U},\mathcal{T},\mathcal{A},\mathcal{P}} \quad & \sum_{p=1}^{N_p} \left[\mathbb{M}^p + \mathbf{W^p}^T \mathbf{Q}^p \right] \\
\text{s. t.} \quad & 0 = \mathbf{D}^p \mathbf{X}^p - \mathbf{F}^p = 0 \\
& -\infty \leq \mathbf{G}^p \leq 0 \\
& 0 = \mathbf{H}^p = 0 \\
& \underline{\mathbf{E}} \leq \mathbf{E} \leq \overline{\mathbf{E}} \\
& \mathbf{X}^p \in \mathcal{X}^p; \quad \mathbf{U}^p \in \mathcal{U}^p; \\
& t_0^p \in \mathcal{T}_0^p; \quad t_{N_c^p}^p \in \mathcal{T}_f^p; \\
& \mathbf{a}^p \in \mathcal{A}^p;
\end{aligned}
$$

$$
\begin{aligned}
& \int_{t_0^p}^{t_f^p} \mathcal{L}^{(p)} \left(\mathbf{x}^p, \mathbf{u}^p, t^p, \mathbf{a}^p \right) dt \\
&= \sum_{s=1}^{N_s^p} \frac{\left(t_{N_c^p}^p - t_0^p \right) \times h_s^p}{2} \int_{-1}^{1} \mathcal{L}^{(p)}(\tau) d\tau \\
&= \sum_{s=1}^{N_s^p} \frac{\left(t_{N_c^p}^p - t_0^p \right) \times h_s^p}{2} \sum_{i=0}^{d_s^p} \mathbf{w}_{i,s} \mathcal{L}^{(p)}(\tau)
\end{aligned}
$$

$$
\mathbf{W}^p = \begin{bmatrix} \mathbf{w}_1 \\ \vdots \\ \mathbf{w}_{N_s^p} \end{bmatrix}^p_{(N_c^p+1)\times 1} \quad ; \mathbf{w}_s = [\mathbf{w}_{i,s}]; \; i \in [1, d_s^p] \forall s > 1
$$

Continuous time OCP → NLP : Lagrange term

$$\min_{\mathcal{X},\mathcal{U},\mathcal{T},\mathcal{A},\mathcal{P}} \quad \sum_{p=1}^{N_p} \left[\mathbb{M}^p + \mathbf{W}^{\mathbf{p}\,T} \mathbf{Q}^p\right]$$

$$\begin{aligned}
\text{s. t.} \quad 0 &= \mathbf{D}^p\mathbf{X}^p - \mathbf{F}^p && = 0\\
-\infty &\leq \mathbf{G}^p && \leq 0\\
0 &= \mathbf{H}^p && = 0\\
\underline{\mathbf{E}} &\leq \mathbf{E} && \leq \overline{\mathbf{E}}\\
&\mathbf{X}^p \in \mathcal{X}^p; \quad \mathbf{U}^p \in \mathcal{U}^p;\\
&t_0^p \in \mathcal{T}_0^p; \quad t_{N_c^p}^p \in \mathcal{T}_f^p;\\
&\mathbf{a}^p \in \mathcal{A}^p;
\end{aligned}$$

$$\mathbb{Q}^p = [q_i]$$

$$q_i = \frac{\left(t_{N_c^p}^p - t_0^p\right) \times r_i^p \times \mathcal{L}^{(p)}(t_i)}{2}$$

$$r_i^p = h_s^p; \qquad \text{if} \qquad \sum_{j=1}^{s-1} d_j^p \leq i < \sum_{j=1}^{s} d_j^p$$

Continuous time OCP $\rightarrow$ NLP : Differential operator

$$\dot{\mathbf{x}}^s(\tau) = \begin{bmatrix} \dot{\phi}_0(\tau_0) & \dots & \dot{\phi}_{d_s^p}(\tau_0) \\ \vdots & \ddots & \vdots \\ \dot{\phi}_0(\tau_{d_s^p}) & \dots & \dot{\phi}_{d_s^p}(\tau_{d_s^p}) \end{bmatrix} \begin{bmatrix} \mathbf{x}_0 \\ \vdots \\ \mathbf{x}_{d_s^p} \end{bmatrix} = \mathbf{D}_s \mathbf{x}^s$$

$$\min_{\mathcal{X},\mathcal{U},\mathcal{T},\mathcal{A},\mathcal{P}} \quad \sum_{p=1}^{N_p} \left[\mathbb{M}^p + \mathbf{W}^{\mathbf{p}\,T} \mathbf{Q}^p \right]$$

$$\begin{aligned} \text{s. t.} \quad 0 = {} & \color{blue}{\mathbf{D}^p}\color{black}\mathbf{X}^p - \mathbf{F}^p && = 0 \\ -\infty \leq {} & \mathbf{G}^p && \leq 0 \\ 0 = {} & \mathbf{H}^p && = 0 \\ \underline{\mathbf{E}} \leq {} & \mathbf{E} && \leq \overline{\mathbf{E}} \\ & \mathbf{X}^p \in \mathcal{X}^p; && \mathbf{U}^p \in \mathcal{U}^p; \\ & t_0^p \in \mathcal{T}_0^p; && t_{N_c^p}^p \in \mathcal{T}_f^p; \\ & \mathbf{a}^p \in \mathcal{A}^p; \end{aligned}$$

$$\mathbf{D}^p = \begin{bmatrix} \begin{matrix} \bullet&\bullet&\bullet&\bullet&\bullet \\ \bullet&\bullet&\bullet&\bullet&\bullet \\ \bullet&\bullet&\bullet&\bullet&\bullet \\ \bullet&\bullet&\bullet&\bullet&\bullet \\ \bullet&\bullet&\bullet&\bullet&\bullet \end{matrix} & & & & \\ & \begin{matrix} \bullet&\bullet&\bullet \\ \bullet&\bullet&\bullet \\ \bullet&\bullet&\bullet \end{matrix} & & & \\ & & \ddots & & \\ & & & \begin{matrix} \bullet&\bullet&\bullet&\bullet \\ \bullet&\bullet&\bullet&\bullet \\ \bullet&\bullet&\bullet&\bullet \\ \bullet&\bullet&\bullet&\bullet \end{matrix} & \\ & & & & \begin{matrix} \bullet&\bullet&\bullet \\ \bullet&\bullet&\bullet \\ \bullet&\bullet&\bullet \end{matrix} \end{bmatrix}$$

Continuous time OCP → NLP : Dynamics

$$
\begin{aligned}
\min_{\mathcal{X},\mathcal{U},\mathcal{T},\mathcal{A},\mathcal{P}} \quad & \sum_{p=1}^{N_p} \left[\mathbb{M}^p + \mathbf{W^p}^T \mathbf{Q}^p \right] \\
\text{s. t.} \quad & 0 = \mathbf{D}^p \mathbf{X}^p - \mathbf{F}^p \quad = 0 \\
& -\infty \leq \mathbf{G}^p \quad \leq 0 \\
& 0 = \mathbf{H}^p \quad = 0 \\
& \underline{\mathbf{E}} \leq \mathbf{E} \quad \leq \overline{\mathbf{E}} \\
& \mathbf{X}^p \in \mathcal{X}^p; \quad \mathbf{U}^p \in \mathcal{U}^p; \\
& t_0^p \in \mathcal{T}_0^p; \quad t_{N_c^p}^p \in \mathcal{T}_f^p; \\
& \mathbf{a}^p \in \mathcal{A}^p;
\end{aligned}
$$

$$
\mathbf{F}^p = \begin{bmatrix} f_0^1 & \cdots & f_0^{n_x} \\ f_1^1 & \cdots & f_1^{n_x} \\ \vdots & \ddots & \vdots \\ f_{N_c^p}^1 & \cdots & f_{N_c^p}^{n_x} \end{bmatrix}^p
$$

Continuous time OCP $\rightarrow$ NLP : Path constraints

$$
\begin{aligned}
\min_{\mathcal{X},\mathcal{U},\mathcal{T},\mathcal{A},\mathcal{P}} \quad & \sum_{p=1}^{N_p} \left[\mathbb{M}^p + \mathbf{W^p}^T \mathbf{Q}^p \right] \\
\text{s. t.} \quad & 0 = \mathbf{D}^p \mathbf{X}^p - \mathbf{F}^p \quad = 0 \\
& -\infty \leq \mathbf{G}^p \quad \leq 0 \\
& 0 = \mathbf{H}^p \quad = 0 \\
& \underline{\mathbf{E}} \leq \mathbf{E} \quad \leq \overline{\mathbf{E}} \\
& \mathbf{X}^p \in \mathcal{X}^p; \quad \mathbf{U}^p \in \mathcal{U}^p; \\
& t_0^p \in \mathcal{T}_0^p; \quad t_{N_c^p}^p \in \mathcal{T}_f^p; \\
& \mathbf{a}^p \in \mathcal{A}^p;
\end{aligned}
$$

$$
\mathbf{G}^p = \begin{bmatrix} g_0^1 & \cdots & g_0^{n_{pc}} \\ g_1^1 & \cdots & g_1^{n_{pc}} \\ \vdots & \ddots & \vdots \\ g_{N_c^p}^1 & \cdots & g_{N_c^p}^{n_{pc}} \end{bmatrix}^p
$$

Continuous time OCP → NLP : Terminal constraints

$$
\begin{aligned}
\min_{\mathcal{X},\mathcal{U},\mathcal{T},\mathcal{A},\mathcal{P}} \quad & \sum_{p=1}^{N_p} \left[\mathbb{M}^p + \mathbf{W^p}^T \mathbf{Q}^p \right] \\
\text{s. t.} \quad & 0 = \mathbf{D}^p \mathbf{X}^p - \mathbf{F}^p \quad = 0 \\
& -\infty \leq \mathbf{G}^p \quad \leq 0 \\
& 0 = \mathbf{H}^p \quad = 0 \\
& \underline{\mathbf{E}} \leq \mathbf{E} \quad \leq \overline{\mathbf{E}} \\
& \mathbf{X}^p \in \mathcal{X}^p; \quad \mathbf{U}^p \in \mathcal{U}^p; \\
& t_0^p \in \mathcal{T}_0^p; \quad t_{N_c^p}^p \in \mathcal{T}_f^p; \\
& \mathbf{a}^p \in \mathcal{A}^p;
\end{aligned}
$$

$$
\mathbf{H}^p = \begin{bmatrix} h_0^1 & \dots & h_0^{n_{tc}} \\ h_1^1 & \dots & h_1^{n_{tc}} \\ \vdots & \ddots & \vdots \\ h_{N_c^p}^1 & \dots & h_{N_c^p}^{n_{tc}} \end{bmatrix}^p
$$

Continuous time OCP → NLP : Event constraints

$$\min_{\mathcal{X},\mathcal{U},\mathcal{T},\mathcal{A},\mathcal{P}} \quad \sum_{p=1}^{N_p} \left[\mathbb{M}^p + \mathbf{W^p}^T \mathbf{Q}^p\right]$$

$$\begin{aligned}
\text{s. t.} \quad & 0 = \mathbf{D}^p \mathbf{X}^p - \mathbf{F}^p && = 0 \\
& -\infty \leq \mathbf{G}^p && \leq 0 \\
& 0 = \mathbf{H}^p && = 0 \\
& \underline{\mathbf{E}} \leq \mathbf{E} && \leq \overline{\mathbf{E}} \\
& \mathbf{X}^p \in \mathcal{X}^p; \quad \mathbf{U}^p \in \mathcal{U}^p; \\
& t_0^p \in \mathcal{T}_0^p; \quad t_{N_c^p}^p \in \mathcal{T}_f^p; \\
& \mathbf{a}^p \in \mathcal{A}^p;
\end{aligned}$$

$$\mathbf{E} = \left[\mathbf{e}^l\right]_{n_e \times 1}; \qquad \forall\ l \in \mathcal{P}$$

$$\mathbf{e}^l = \begin{bmatrix} \mathbf{x}_{N_c^{p_i}}^{p_i} - \mathbf{x}_0^{p_j} \\ \mathbf{u}_{N_c^{p_i}}^{p_i} - \mathbf{u}_0^{p_j} \\ t_f^{p_i} - t_0^{p_j} \end{bmatrix} \qquad \forall\ (p_i, p_j) = l \in \mathcal{P}$$

$$\underline{\mathbf{E}} = \left[\underline{\mathbf{e}}^l\right]; \qquad \underline{\mathbf{e}}^l = \begin{bmatrix} \underline{\mathbf{e}_x^l} \\ \underline{\mathbf{e}_u^l} \\ \underline{\mathbf{e}_t^l} \end{bmatrix} \qquad \forall\ l \in \mathcal{P}$$

$$\overline{\mathbf{E}} = \left[\overline{\mathbf{e}}^l\right]; \qquad \overline{\mathbf{e}}^l = \begin{bmatrix} \overline{\mathbf{e}_x^l} \\ \overline{\mathbf{e}_u^l} \\ \overline{\mathbf{e}_t^l} \end{bmatrix} \qquad \forall\ l \in \mathcal{P}$$

Interpolation

- Residual in dynamics at non-collocation nodes is a good estimate of quality of solution.
- States and controls at any given nodes $[\tau_1 \dots \tau_n]$ can be computed via projection operation.

$$\mathbf{X}_I^p = \mathbf{C}_I^p \mathbf{X}^p; \quad \mathbf{U}_I^p = \mathbf{C}_I^p \mathbf{U}^p; \qquad \dot{\mathbf{X}}_I^p = \mathbf{D}_I^p \mathbf{X}^p; \quad \dot{\mathbf{U}}_I^p = \mathbf{D}_I^p \mathbf{U}^p;$$

- Composite interpolation matrix

$$\begin{bmatrix} \mathbf{x}_1^I \\ \vdots \\ \mathbf{x}_{I_s^p}^I \end{bmatrix} = \begin{bmatrix} \phi_0(\tau_1^I) & \dots & \phi_{d_s^p}(\tau_1^I) \\ \vdots & \ddots & \vdots \\ \phi_0(\tau_{I_s^p}^I) & \dots & \phi_{d_s^p}(\tau_{I_s^p}) \end{bmatrix}_{I_s^p \times (d_s^p+1)} \begin{bmatrix} \mathbf{x}_0 \\ \vdots \\ \mathbf{x}_{d_s^p} \end{bmatrix} = \mathbf{C}_s \mathbf{x}^s$$

NLP creation algorithm

Algorithm 1 NLP formulation of multi-phase continuous time OCP

procedure $\mathrm{NLP}(N_p, N_s^p, d_s^p)$

  for Each phase (p) in $[1, \ldots, N_p]$ **do**

    $(\mathbf{X}^p, \mathbf{U}^p, t_0^p, t_{N_c^p}^p, \mathbf{a}^p) \leftarrow$ Create optimization variables ▷ Variables

    $(\mathbf{D}^p, \mathbf{W}_p) \leftarrow$ Compute collocation approximation ▷ Collocation matrices

    $(\mathbf{F}^p, \mathbf{LB}_f^p, \mathbf{UB}_f^p) \leftarrow$ Construct system dynamics matrix ▷ Dynamics

    $(\mathbf{G}^p, \mathbf{LB}_g^p, \mathbf{UB}_g^p) \leftarrow$ Create path constraints matrix ▷ Path constraints

    $(\mathbf{H}^p, \mathbf{LB}_h^p, \mathbf{UB}_h^p) \leftarrow$ Create terminal constraints vector ▷ Terminal con.

    $(\mathbf{A}^p, \mathbf{LB}_a^p, \mathbf{UB}_a^p) \leftarrow$ Additional constraints (if any) ▷ Additional con.

    $\mathbf{Q}^p \leftarrow$ Compute running costs matrix ▷ Lagrangian

    $\mathbf{F} \leftarrow (\mathbf{D}^p\mathbf{X}^p - \mathbf{F}^p, \mathbf{G}^p, \mathbf{H}^p, \mathbf{A}^p)$ ▷ Update NLP constraint vector

    $\mathbf{LB} \leftarrow (\mathbf{LB}_f^p = 0, \mathbf{LB}_g^p = -\infty, \mathbf{LB}_h^p = 0, \mathbf{LB}_a^p)$ ▷ Update NLP cons. lower bound

    $\mathbf{UB} \leftarrow (\mathbf{UB}_f^p = 0, \mathbf{UB}_g^p = 0, \mathbf{UB}_h^p = 0, \mathbf{UB}_a^p)$ ▷ Update NLP cons. upper bound

    $J \leftarrow (M^p + \mathbf{W^p}^T \mathbf{Q}^p)$ ▷ Update cost variable

    $\mathbf{X} \leftarrow (\mathbf{X}^p, \mathbf{U}^p, t_0^p, t_{N_c^p}^p, \mathbf{a}^p)$ ▷ Update NLP variables vector

    $\underline{\mathbf{X}} \leftarrow \mathrm{LowerBound}(\mathbf{X}^p, \mathbf{U}^p, t_0^p, t_{N_c^p}^p, \mathbf{a}^p)$ ▷ Update NLP variables lower bound vector

    $\overline{\mathbf{X}} \leftarrow \mathrm{UpperBound}(\mathbf{X}^p, \mathbf{U}^p, t_0^p, t_{N_c^p}^p, \mathbf{a}^p)$ ▷ Update NLP variables upper bound vector

  $(\mathbf{E}, \mathbf{LB}_e, \mathbf{UB}_e) \leftarrow$ Construct event constraints matrix ▷ Event constraints

  $\mathbf{F} \leftarrow \mathbf{E}$ ▷ Update NLP constraint vector

  $\mathbf{LB} \leftarrow \mathbf{LB}_e$ ▷ Update NLP constraint lower bound vector

  $\mathbf{UB} \leftarrow \mathbf{UB}_e$ ▷ Update NLP constraint upper bound vector

  return $(\mathbf{F}, \mathbf{LB}, \mathbf{UB}, J, \mathbf{X}, \underline{\mathbf{X}}, \overline{\mathbf{X}})$

Implementation and features

- Collocation nodes supported : Legendre-Gauss-Radau, Legendre-Gauss-Lobatto, Chebyshev-Gasuss-Lobatto

Implementation and features

- Collocation nodes supported : Legendre-Gauss-Radau, Legendre-Gauss-Lobatto, Chebyshev-Gasuss-Lobatto
- Initial solution : Interpolated from user defined initial and final guess

Implementation and features

- Collocation nodes supported : Legendre-Gauss-Radau, Legendre-Gauss-Lobatto, Chebyshev-Gasuss-Lobatto
- Initial solution : Interpolated from user defined initial and final guess
- OCP definition : User defined functions for system dynamics, constraints, running costs and terminal costs.

Implementation and features

- Collocation nodes supported : Legendre-Gauss-Radau, Legendre-Gauss-Lobatto, Chebyshev-Gasuss-Lobatto
- Initial solution : Interpolated from user defined initial and final guess
- OCP definition : User defined functions for system dynamics, constraints, running costs and terminal costs.

- Supports variable scaling, additional constraints on control slope, box constraints on variables at non-collocation nodes, continuity of slope of control across segments

Implementation and features

- Collocation nodes supported : Legendre-Gauss-Radau, Legendre-Gauss-Lobatto, Chebyshev-Gasuss-Lobatto
- Initial solution : Interpolated from user defined initial and final guess
- OCP definition : User defined functions for system dynamics, constraints, running costs and terminal costs.

- Supports variable scaling, additional constraints on control slope, box constraints on variables at non-collocation nodes, continuity of slope of control across segments
- post-processing : Plotting and retrieving the trajectories of states, controls.

Algorithm

Algorithm 2 MPOPT algorithm

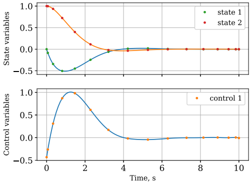
1: **procedure** MPOPT(OCP, N_s^p, d_s^p, collocation_scheme)
2: $(\mathbf{F}, \mathbf{LB}, \mathbf{UB}, J, \mathbf{Z}, \underline{\mathbf{Z}}, \overline{\mathbf{Z}}) \leftarrow$ Initialize NLP matrices ▷ Create NLP
3: $Z_0 \leftarrow$ Create initial solution estimate ▷ Initial solution
4: $h_s^p \leftarrow \dfrac{1}{N_s^p}$ ▷ Initialize segment width fractions
5: $\mathcal{Z} \leftarrow$ Solve NLP$((\mathbf{F}, \mathbf{LB}, \mathbf{UB}, J, \mathbf{Z}, \underline{\mathbf{Z}}, \overline{\mathbf{Z}}),\ Z_0,\ h_s^p)$ ▷ Solve NLP
6: $\{\mathbf{X}^i, \mathbf{U}^i, t_0^i, t_f^i, a^i;\ i = 1, \ldots, p\} \leftarrow$ Retrieve OCP solution from the NLP solution $\mathcal{Z}$
7: **return** $\{\mathbf{X}^i, \mathbf{U}^i, t_0^i, t_f^i, a^i\}$ ▷ Return OCP solution

Single phase: Van der Pol 2D Oscillator OCP

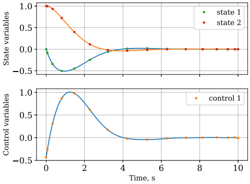
$$
\begin{aligned}
\min_{x,u} \quad & J = 0 + \int_{t_0}^{t_f} \left(x_0^2 + x_1^2 + u^2\right) dt \\
\text{subject to} \quad & \dot{x}_0 = (1 - x_1^2)x_0 - x_1 + u \\
& \dot{x}_1 = x_0 \\
& x_1 \geq -0.25 \\
& -1 \leq u \leq 1 \\
& x_0(t_0) = 0;\ \ x_1(t_0) = 1; \\
& t_0 = 0.0;\ \ t_f = 10
\end{aligned}
$$

```python
from mpopt import mp
ocp = mp.OCP(n_states=2, n_controls=1, n_phases=1)
ocp.dynamics[0]=lambda x,u,t:[(1-x[1]*x[1])*x[0]-x[1]+u,x[0]]
ocp.running_costs[0]=lambda x,u,t: x[0]*x[0]+x[1]*x[1]+u*u
ocp.lbx[0][1] = -0.25
ocp.lbu[0], ocp.ubu[0] = -1, 1
ocp.x00[0] = [0, 1]
ocp.lbtf[0], ocp.ubtf[0] = 10, 10
mpo, post = mp.solve(ocp, n_segments=1, poly_orders=15,"LGR")
```

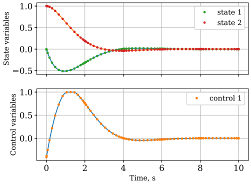
Single phase: Van der Pol 2D Oscillator OCP … Contd.



(a) Single segment collocation (LGR)



(b) Single segment collocation (CGL)



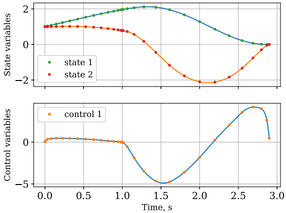
(c) Multiple-segment collocation (LGR)

- Global collocation, Multi-segment collocation with different nodes is demonstrated for single phase OCP with the example.

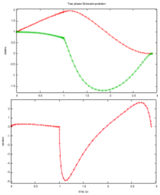
Case	Optimal cost	NLP var.	NLP cons.	Time total	Time NLP sol.
LGR, 15 nodes	2.87373	50	47	85.2 ms $\pm$ .842 ms	$\approx 9.5ms$
CGL, 15 nodes	2.87397	50	47	84.2 ms $\pm$ 1.14 ms	$\approx 8.7ms$
LGR, 5 seg, 75 nodes	2.87332	230	227	326 ms $\pm$ .872 ms	$\approx 30ms$

Multi phase: Two-phase Schwartz problem

$$
\begin{aligned}
\min_{x,u} \quad & J = 5(x_0(t_f)^2 + x_1(t_f)^2) + \int_{t_0}^{t_f} 0 dt \\
\text{subject to} \quad & \dot{x_0} = x_1 \\
& \dot{x_1} = u - 0.1(1 + 2x_0^2)x_1 \\
\textit{Phase 1:} \quad & 1 - 9(x_0 - 1)^2 - \left(\frac{x_1 - 0.4}{0.3}\right)^2 \leq 0 \\
& x_1 \geq -0.8 \\
& -1 \leq u \leq 1 \\
& x_0(t_0) = 1; \; x_1(t_0) = 1; \\
& t_0 = 0; \; t_f = 1 \\
\textit{Phase 2:} \quad & t_0 = 1; \; t_f = 2.9 \\
& x \in \mathbb{R}^2; \; u \in \mathbb{R}
\end{aligned}
$$



(a) Single segment collocation



(b) Solution from PSOPT

Multi-stage launch vehicle ascent trajectory

$$\min_{x,u} \quad J = -m(t_f)$$

subject to

Dynamics:

$$\dot{\mathbf{r}} = \mathbf{v}$$

$$\dot{\mathbf{v}} = -\frac{\mu}{\|\mathbf{r}\|^3}\mathbf{r} + \frac{T^p}{m}\mathbf{u} - \frac{C_d A^{\text{ref}} \rho_0 \left\|\mathbf{v} - (\omega_e \times \mathbf{r})\right\| (\mathbf{v} - (\omega_e \times \mathbf{r})) e^{\frac{\|\mathbf{r}\| - R_e}{H_0}}}{2m}$$

$$\dot{m} = -\dot{m}^p \qquad p = 0, 1, 2, 3$$

Path constraints:

$$\|\mathbf{u}\| = 1$$

$$\|\mathbf{r}\| \geq R_e$$

Terminal constraints:

$$t_0^0 = 0;\ t_0^1 = 75.2;\ t_0^2 = 150.4;\ t_0^3 = 261$$

$$a_f(\mathbf{r_f}, \mathbf{v_f}) = 24361140; e_f(\mathbf{r_f}, \mathbf{v_f}) = 0.7308;$$

$$i_f(\mathbf{r_f}, \mathbf{v_f}) = \frac{28.5\pi}{180}; \Omega_f(\mathbf{r_f}, \mathbf{v_f}) = \frac{269.8\pi}{180};$$

$$\omega_f(\mathbf{r_f}, \mathbf{v_f}) = \frac{130.5\pi}{180};$$

Event constraints:

$$\mathbf{r}(t_f^p) - \mathbf{r}(t_0^{p+1}) = 0;\ \mathbf{v}(t_f^p) - \mathbf{v}(t_0^{p+1}) = 0; \quad p = 0, 1, 2$$

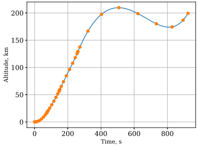
$$m(t_f^p) - m(t_0^{p+1}) = [13680, 6840, 8830] \quad p = 0, 1, 2$$

Initial conditions:

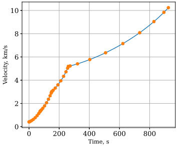
$$\mathbf{r}(t_0^0) = [5605222.973, 0, 3043387.761];\ \mathbf{v}(t_0^0) = [0, 408.74, 0];$$

$$m(t_0^0) = 301454$$

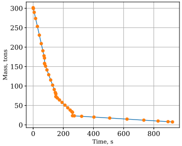
Multi-stage launch vehicle ascent trajectory ... contd.



(a) Altitude profile



(b) Inertial velocity



(c) Mass decay profile

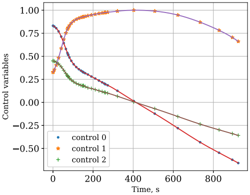


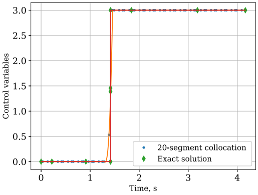
Figure: Control profile

Solver	**Payload, kg**	**Tf, s**	solve time, s
MPOPT	7529.7129	924.1393	3.20
PSOPT	7529.6610	924.1413	3.35
GPOPS-II	7529.7123	-	-
SOCS	7529.7125	-	-

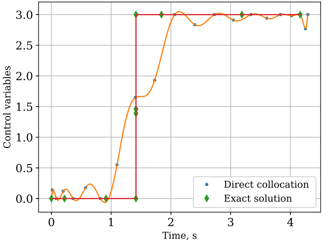
Contents

Moon lander example

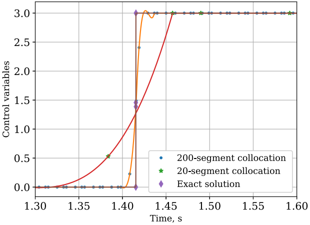
$$
\begin{aligned}
\min_{x,u} \quad & J = \int_{t_0}^{t_f} u \; dt \\
\text{subject to} \quad & \dot{x_0} = x_1 \\
& \dot{x_1} = u - 1.5 \\
& x_0 \geq 0 \\
& 0 \leq u \leq 3 \\
& x_0(t_0) = 10;\; x_1(t_0) = -2;\; t_0 = 0.0; \\
& x_0(t_f) = 0;\; x_1(t_f) = 0;\; t_f = \text{free variable}
\end{aligned}
$$



Multi-segment collocation



Global collocation with 20 nodes



Non-adaptive vs exact solution

Adaptive segment width refinement

- Quality of the solution is generally assessed by computing the residual in dynamics at non-collocation points

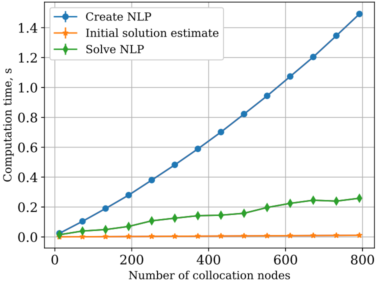
$$\mathbf{R}^p = \mathbf{D}_l^p \mathbf{X}^p - \mathbf{F}_l^p$$
$$\epsilon = \max \|\mathbf{R}^p\|$$

- Adaptive grid refinement improves the quality of solution especially when the solution is discontinuous.

Solver	**Method**
GPOPS-II	Customized iterative grid refinement strategy by merging and splitting the segments based on the estimates of residuals
PSOPT	Non-linear least square fit to estimate the required number of collocation points (Iterative)

Adaptive grid refinement

- Existing solvers refine the grid by adding new collocation nodes which requires re-initialization of NLP
- Iteratively increasing the number of nodes makes computations expensive in addition to being indeterministic in computational time.



- In this thesis, three methods are proposed for scaling the given grid.

Adaptive schemes implemented in MPOPT

- Iterative schemes : Adaptive schemes - I & II
  - Heuristic method
  - Residual based method
    1. Equal area residual segments
    2. Merge or split segments (preserve total no. of segments)
- Non-iterative : Adaptive scheme-III

Algorithm for iterative adaptive grid refinement

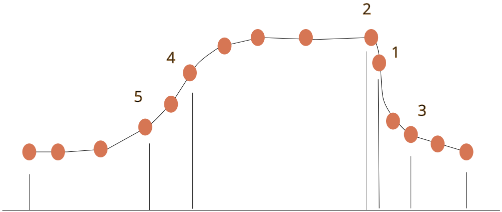
Algorithm 3 Iterative segment width refinement: Adaptive solver

procedure H-ADAPTIVE(N_p, N_s^p, d_s^p)
    NLP(N_p, N_s^p, d_s^p) $\leftarrow$ Initialize NLP ▷ Algorithm
    $\mathrm{h}_s^p \leftarrow \dfrac{1}{N_s^p}$ ▷ Equal width segments
    $\mathcal{Z}^0 = \{\mathbf{X}^i, \mathbf{U}^i, t_0^i, t_f^i, a^i;\ i = 1, \ldots, p\} \leftarrow$ Solve NLP with h_s^p ▷ Reference NLP solution
    for Each iteration k in $[1, \ldots, Iter_{max}]$ **do**
        $\mathbf{R} \leftarrow$ Estimate of max. residual in $\mathcal{Z}^{(k-1)}$ ▷ At non-collocation nodes
        if $(\mathbf{R} < \epsilon^r)$ or $(\dfrac{dh_s^p}{dk} \leq \epsilon^h)$ **then**
            break;
        else
            $h_s^p \leftarrow$ Refine segment widths using solution $\mathcal{Z}^{(k-1)}$ ▷ **Adaptive schemes - I II**
        $\mathcal{Z}^k = \{\mathbf{X}^i, \mathbf{U}^i, t_0^i, t_f^i, a^i;\ i = 1, \ldots, p\} \leftarrow$ Solve NLP with h_s^p ▷ Refined NLP solution
    return $\mathcal{Z}$ ▷ Return updated solution

Adaptive scheme-I : Heuristic method

- Scale the segments such that nodes are dense near the discontinuity
- Generalizing, define a measure of denseness of the collocation nodes in terms of local variables themselves

Sl. no	Criteria ($\mathcal{C}$)	Remarks
1	$\|\dot{\mathbf{u}}\|$ or $\|\dot{\mathbf{x}}\|$	Norm of slope of control/state variables
2	$\dot{u}_i;\ i = 1, \ldots, n_u$	Slope of any control variable of choice (i)
3	$\|\ddot{\mathbf{u}}\|$ or $\|\ddot{\mathbf{x}}\|$	Norm of second derivative of control/state variables
4	$\dot{x}_i;\ i = 1, \ldots, n_x$	Slope of any state variable of choice (i)
5	$f(\mathbf{x}, \mathbf{u}) \in \mathbb{R}$	Custom criteria



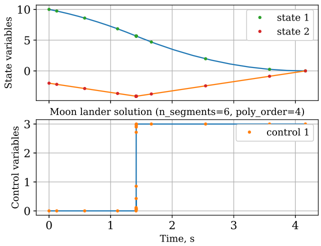
Control input profile

Adaptive scheme-I: Algorithm

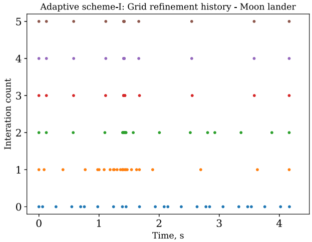
Algorithm 4 Iterative grid refinement: Adaptive scheme-I

procedure H-ADAPTIVE-I($\mathcal{Z}^k, h_s^p, \mathcal{T}^{\text{grid}}$)
  $\mathcal{C} \leftarrow$ Select criteria for measure of discontinuity
  $\mathcal{T}^{\text{pivot}} = \{t_i^p \in \mathcal{T}^{\text{grid}};\ \mathcal{C}(t_i^p) > \epsilon;\ \mathcal{C}(t_i^p) > \mathcal{C}(t_j^p)\ \forall\ i > j\}$ ▷ Compute pivot locations in $\mathcal{Z}^k$
  if No. of pivots $> N_s^p - 2$ **then** ▷ Enough pivots to define new segments
    $\mathcal{T}^{\text{pivot}} \leftarrow$ Select first $N_s^p - 1$ pivots from $\mathcal{T}^{\text{pivot}}$
    $\mathcal{T}^{\text{pivot}} \leftarrow [t_0^p, t_f^p]$ ▷ Add start and final times to pivots
    $\mathcal{T}^{\text{pivot}} \leftarrow$ Sort pivots in ascending order
  else ▷ Not enough pivots to define new segments
    $\mathcal{T}^{\text{pivot}} \leftarrow [t_0^p, t_f^p]$ ▷ Add start and final times to pivots
    $\mathcal{T}^{\text{pivot}} \leftarrow$ Sort pivots in ascending order
    $N \leftarrow$ Number of pivots in $\mathcal{T}^{\text{pivot}}$
    $\mathcal{T}^{\text{pivot}} \leftarrow$Split first segment in $\mathcal{T}^{\text{pivot}}$ into $\left\lfloor \dfrac{N_s^p - N}{2} \right\rfloor$ of equal segments
    $\mathcal{T}^{\text{pivot}} \leftarrow$ Split last segment in $\mathcal{T}^{\text{pivot}}$ into remaining number of equal segments
  $h_s^p \leftarrow$ Estimate segment width fractions from $\mathcal{T}^{\text{pivot}}$
  return h_s^p ▷ Return refined segment width fractions

Validation: Scheme-I



(a) Adaptive solution : Scheme-I



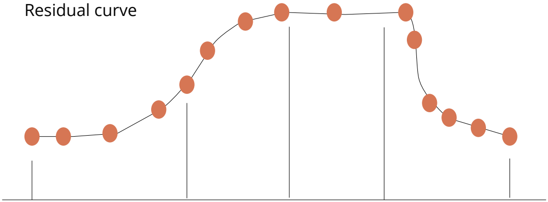
(b) Grid refinement history

- The exact solution is achieved with as low as 15 nodes compared to 600 nodes in non-adaptive scheme for the moon-lander OCP
- Convergence is not guaranteed for all problems

Adaptive scheme-II : Residual based method

- Residual error in dynamics is used to refine segment widths. Let curve $R(t)$ represent 2-norm of the residual error in the whole domain.
- **Sub-method-1: Equal residual segments**

$$\int_{t_0^s}^{t_f^s} R(t)\ dt = \int_{t_0^{s+1}}^{t_f^{s+1}} R(t)\ dt; \qquad s \in [1, \dots, N_s^p - 1]$$
$$t_f^s = t_0^{s+1}$$

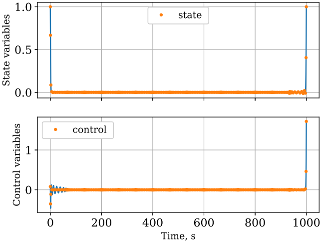


Equal residual segments : Adaptive scheme-II

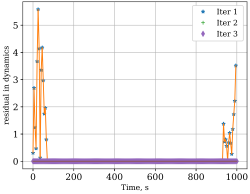
Validation : Adaptive Scheme-II (Equal residual segments)

$$\min_{x,u} \quad J = 0 + \frac{1}{2}\int_{t_0}^{t_f} (x^2 + u^2)\ dt$$

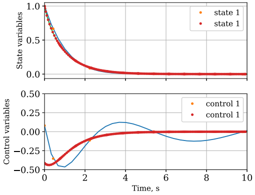
subject to
$$\dot{x} = -x^3 + u$$
$$x_0(t_0) = 1;\ t_0 = 0;$$
$$x_1(t_f) = 0;\ t_f = 1000$$



Non-adaptive solution (15 seg.)



Residual evolution

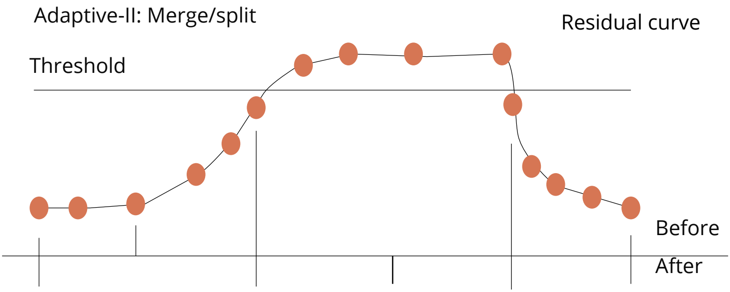


Comparison

Adaptive scheme-II : Residual based method... contd.

- **Sub-method-2: Merge/split segments**

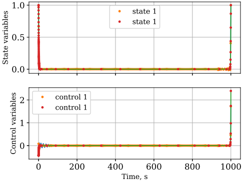
Merge two consecutive segments if they have acceptable tolerance and split remaining segments while conserving the total number of segments.



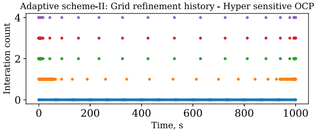
Residual based merge-split adaptive scheme

Validation : Adaptive Scheme-II

- **Submethod-1: Equal residual segments**

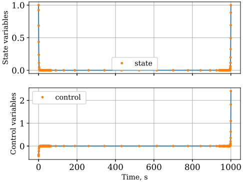


(a) Adaptive solution

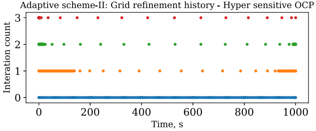


(b) Grid refinement history

- **Submethod-2: Merge/Split method**



(a) Adaptive solution



(b) Grid refinement history

Adaptive scheme-III : Direct optimization

- The segment width fractions are added to the optimization variables in NLP formulation
- Additional constraints on residual.

$$\sum_{i=1}^{N_s^p} h_i^p = 1$$

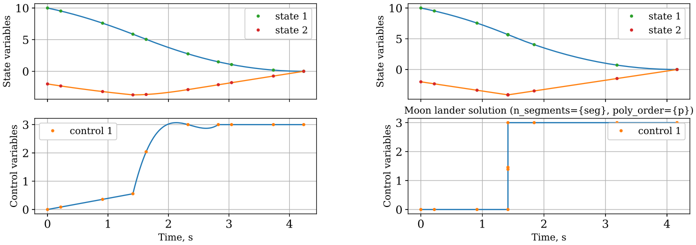
$$-\epsilon^{tol} \leq h_s^p \times R_s^p \leq \epsilon^{tol} \qquad \epsilon^{tol} > 0;$$

$$\epsilon^h < h_s^p \leq 1; \qquad \epsilon^h > 0;$$

$$\rightarrow$$

$$\begin{aligned}
\min_{\mathcal{X},\mathcal{U},\mathcal{T},\mathcal{A},\mathcal{P},\mathcal{H}} \quad & \sum_{p=1}^{N_p} \left[\mathbb{M}^p + \mathbf{W}_p^T \mathbf{Q}^p\right] \\
\text{s. t.} \quad & 0 = \mathbf{D}^p\mathbf{X}^p - \mathbf{F}^p \quad = 0 \\
& -\infty = \mathbf{G}^p \quad \leq 0 \\
& 0 = \mathbf{H}^p \quad = 0 \\
& \underline{\mathbf{E}}^l = \mathbf{E}^l \quad = \overline{\mathbf{E}}^l \\
& -\epsilon^{tol} = \mathbf{R}^p \quad = \epsilon^{tol} \\
& \mathbf{X}^p \in \mathcal{X}^p; \quad \mathbf{U}^p \in \mathcal{U}^p; \\
& t_0^p \in \mathcal{T}_0^p; \quad t_{N_c^p}^p \in \mathcal{T}_f^p; \quad \mathbf{a}^p \in \mathcal{A}^p;
\end{aligned}$$

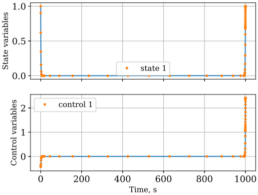
Validation: Adaptive scheme-III - Moon lander OCP



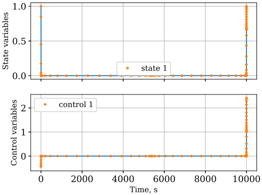
(a) Non-adaptive solution (b) Adaptive-III solution

Scheme	Terminal time	Optimal cost	Computational time
Non-adaptive	4.17081	8.25622	15.8 ms ± 0.121 ms
Adaptive-I: Heuristic	4.16415	8.24622	267 ms ± 1.91 ms
Adaptive-II: Equal residual	4.16414	8.24621	262 ms ± 2.1 ms
Adaptive-II: Merge/split	4.16416	8.24624	233 ms ± 0.899 ms
Adaptive-III: Direct optimization	4.16414	8.24621	92.9 ms ± 0.313 ms

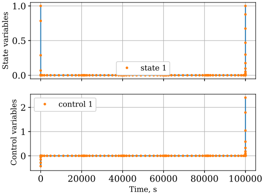
Validation: Adaptive scheme-III - Hyper sensitive OCP



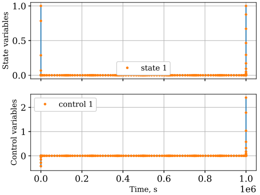
(a) Tf $=$ 1000, 5 segments



(b) Tf $=$ **10000**, 5 segments



(c) Tf $=$ **100000**, 7 segments



(d) Tf $=$ **1000000**, 10 segments

Case study of two-stage rockets : Lorenz T. Biegler

$\min_{x,u}$ $J = -M(t_f)$

subject to

Define $\mathbf{r} = [x_0, x_1, x_2];\ \mathbf{v} = [x_3, x_4, x_5];\ m = x_6;\ \mathbf{u} = [u_0, u_1, u_2, u_3];$

$\mu = 3.986012e14;\ \omega_e = [0, 0, 7.29211585e-5]^T;\ R_e = 6378145;$

$\rho_0 = 1.225;\ H_0 = 7200;\ C_d = 0.5;\ A^{\text{ref}} = 4\pi;$

$T^0 = 9 \times 934e3;\ T^1 = 934e3;\ T^2 = 934e3 \textit{ to } 360e3;$

$\dot{m}^0 = 2521.1;\ \dot{m}^1 = 280.122;\ \dot{m}^2 = 106.45 \textit{ to } 280.122;$

Dynamics: $\dot{\mathbf{r}} = \mathbf{v}$

$$\dot{\mathbf{v}} = -\frac{\mu}{\|\mathbf{r}\|^3}\mathbf{r} + \frac{u_3 T^p}{m}\mathbf{u}_{)}^{\mathbf{2}} - \frac{C_d A^{\text{ref}} \rho_0 \left\|\mathbf{v} - (\omega_e \times \mathbf{r})\right\| (\mathbf{v} - (\omega_e \times \mathbf{r})) e^{\frac{\|\mathbf{r}\| - R_e}{H_0}}}{2m}$$

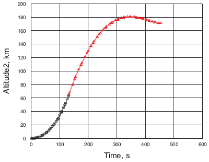
$$\dot{m} = -\dot{m}^p \qquad p = 0, 1, 2$$

Path constraints: $\left\|\mathbf{u_0^2}\right\| = 1; \quad \|\mathbf{r}\| \geq R_e;\ \frac{\rho\|v\|^2}{2} \leq q_{max};\left\|\frac{T_i + D_i}{m_i}\right\| \leq acc_{max}$

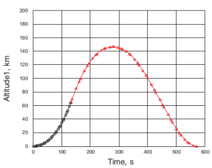
Terminal constraints: $t_0^0 = 0;\ a_f = 6593145;\ e_f = 0.0076;\ i_f = \frac{28.5\pi}{180};\ \Omega_f = \frac{269.8\pi}{180};\ \omega_f = \frac{130.5\pi}{180};$

$r^2(t_f^2) = r^0(t_0^0);\ v^2(t_f^2) = 0;$ ← **Booster back at launch pad**

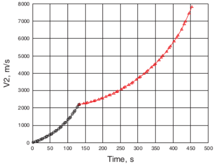
Case study of two-stage rockets: Reference solution



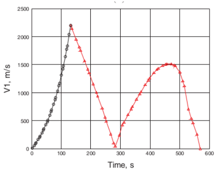
(a) Orbiter altitude



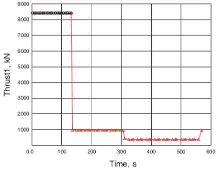
(b) Booster altitude



(c) Orbiter velocity



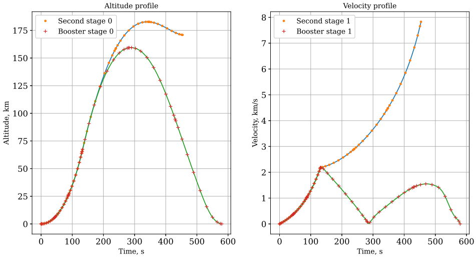
(d) Booster velocity



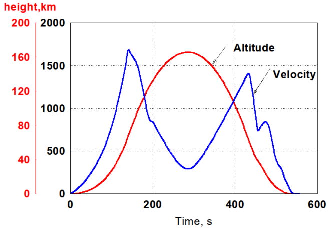
(e) Thrust profile

Parameter	Biegler
Payload, kg	17310
mass at landing, kg	22100
MECO, s	131.4
SECO, s	453.4
Landing, s	569.7
Solve time, s	1.9

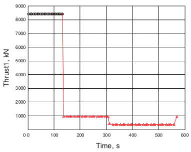
Case study of two-stage rockets: Non-adaptive solution



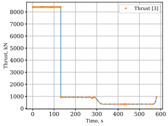
(a) Altitude and velocity



(b) Falcon9 NROL-76 telemetry



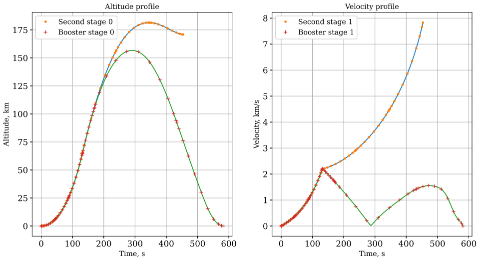
(c) Biegler

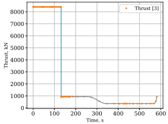


(d) MPOPT

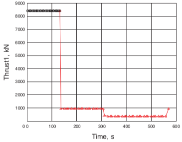
Parameter	Biegler	non-adaptive
Payload, kg	17310	17139.5
mass at landing, kg	22100	22100
MECO, s	131.4	131.4
SECO, s	453.4	453.98
Landing, s	569.7	579.7
Solve time, s	1.9	2.3

Case study of two-stage rockets: Adaptive-I - Heuristic





(b) MPOPT

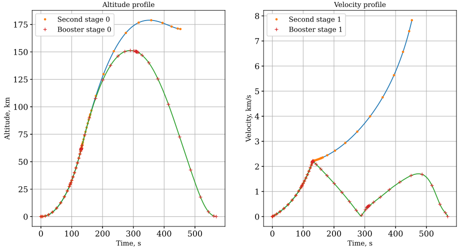


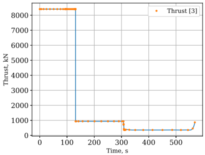
(c) Biegler

(a) Altitude and velocity

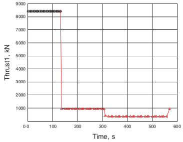
Parameter	Biegler	non-adaptive	Adaptive:Heuristic
Payload, kg	17310	17139.5	17267.4
mass at landing, kg	22100	22100	22100
MECO, s	131.4	131.4	131.4
SECO, s	453.4	453.98	453.52
Landing, s	569.7	579.7	581.3
Solve time, s	1.9	2.3	2.86

Case study of two-stage rockets: Adaptive-III





(a) MPOPT



(b) Biegler

Altitude and velocity

Parameter	Reference	non-adaptive	Adaptive:Heuristic	Adaptive-III
Payload, kg	17310	17139.5	17267.4	17451.4
mass at landing, kg	22100	22100	22100	22100
MECO, s	131.4	131.4	131.4	131.4
SECO, s	453.4	453.98	453.52	452.86
Landing, s	569.7	579.7	581.3	569.1
Solve time, s	1.9	2.3	2.86	17.3

Discussion: Adaptive methods

- Work with standard test cases, convergence is not guaranteed
- Heuristic method works best for all cases involving discontinuity.
- Residual based methods work best if error is uniformly distributed over the domain.
- Direct optimization is found to be effective in all cases, however at higher computational cost

Contents

Source code and packaging

- Source code hosted in GitHub `https://github.com/mpopt`
  - Continuous integration using TravisCI
  - Test coverage report using coverage.io (94%)
  - Documentation pages are auto-generated

- The package is hosted on Python Package Index (PyPI)

  `$ pip install mpopt`

Contents

Limitations of package

- Need to input dynamics, constraints and objective in terms of continuous and differentiable functions.
- Need for user to initialize the collocation scheme and approximating polynomials

Further improvements

- OCP problem definition (User friendly)
- Initial guess improvement (Interpolation)
- Collocation approximation (Derivatives, support for additional schemes)
- Additional constraints implementation (Higher order derivatives)
- Additional NLP solvers support (Testing and validation)
- Extend the package to work with DAEs
- Additional adaptive schemes from literature

Contents

Conclusion

- An open-source, extensible multi-phase optimal control solver is developed, validated.
- Three different adaptive grid refinement schemes are developed, implemented in the solver.
- Adaptive schemes are tested and validated using standard test cases.
- A case study of launch vehicle trajectory optimization with first stage recovery is presented.
- Package is hosted on Python Package Index (PyPI) and source code on GitHub

Thank you

References I

G. Elnagar, M. Kazemi, and M. Razzaghi, "The pseudospectral Legendre method for discretizing optimal control problems", *IEEE Transactions on Automatic Control*, vol. 40, no. 10, pp. 1793–1796, Oct. 1995, ISSN: 1558-2523. DOI: 10.1109/9.467672.

G. Molnárka, "Fornberg, B.: A Practical Guide to Pseudospectral Methods. Cambridge, Cambridge University Press 1996. X. 231 pp., £37.50. ISBN 0-521-49582-2 (Cambridge Monographs on Applied and Computational Mathematics 1)", en, *ZAMM - Journal of Applied Mathematics and Mechanics / Zeitschrift für Angewandte Mathematik und Mechanik*, vol. 77, no. 10, pp. 798–798, 1997, ISSN: 1521-4001. DOI: 10.1002/zamm.19970771017. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/zamm.19970771017 (visited on 06/18/2020).

References II

J. T. Betts, "Survey of Numerical Methods for Trajectory Optimization", *Journal of Guidance, Control, and Dynamics*, vol. 21, no. 2, pp. 193–207, 1998. DOI: 10.2514/2.4231. [Online]. Available: https://doi.org/10.2514/2.4231 (visited on 06/18/2020).

F. Fahroo and I. M. Ross, "Direct Trajectory Optimization by a Chebyshev Pseudospectral Method", *Journal of Guidance, Control, and Dynamics*, vol. 25, no. 1, pp. 160–166, 2002. DOI: 10.2514/2.4862. [Online]. Available: https://doi.org/10.2514/2.4862 (visited on 06/18/2020).

References III

S. L. Campbell, "Practical methods for optimal control using nonlinear programming, John T. Betts, SIAM, Philadelphia, PA, 2001, ISBN 0-89871-488-5", en, *International Journal of Robust and Nonlinear Control*, vol. 14, no. 11, pp. 1019–1021, 2004, ISSN: 1099-1239. DOI: 10.1002/rnc.874. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/rnc.874 (visited on 06/18/2020).

P. E. Gill, W. Murray, and M. A. Saunders, "SNOPT: An SQP Algorithm for Large-Scale Constrained Optimization", *SIAM Review*, vol. 47, no. 1, pp. 99–131, Jan. 2005, ISSN: 0036-1445. DOI: 10.1137/S0036144504446096. [Online]. Available: https://doi.org/10.1137/S0036144504446096 (visited on 06/15/2020).

References IV

D. A. Benson, G. T. Huntington, T. P. Thorvaldsen, and A. V. Rao, "Direct Trajectory Optimization and Costate Estimation via an Orthogonal Collocation Method", *Journal of Guidance, Control, and Dynamics*, vol. 29, no. 6, pp. 1435–1440, 2006. DOI: 10.2514/1.20478. [Online]. Available: https://doi.org/10.2514/1.20478 (visited on 06/18/2020).

J. Nocedal and S. Wright, *Numerical Optimization*, en, 2nd ed., ser. Springer Series in Operations Research and Financial Engineering. New York: Springer-Verlag, 2006, ISBN: 9780387303031. DOI: 10.1007/978-0-387-40065-5. [Online]. Available: https://www.springer.com/de/book/9780387303031 (visited on 11/09/2019).

References V

A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming", en, *Mathematical Programming*, vol. 106, no. 1, pp. 25–57, Mar. 2006, ISSN: 1436-4646. DOI: 10.1007/s10107-004-0559-y. [Online]. Available: https://doi.org/10.1007/s10107-004-0559-y (visited on 06/15/2020).

S. Jain and P. Tsiotras, "Trajectory Optimization Using Multiresolution Techniques", *Journal of Guidance, Control, and Dynamics*, vol. 31, no. 5, pp. 1424–1436, 2008. DOI: 10.2514/1.32220. [Online]. Available: https://doi.org/10.2514/1.32220 (visited on 06/18/2020).

References VI

S. Kameswaran and L. T. Biegler, "Convergence rates for direct transcription of optimal control problems using collocation at Radau points", en, *Computational Optimization and Applications*, vol. 41, no. 1, pp. 81–126, Sep. 2008, ISSN: 1573-2894. DOI: 10.1007/s10589-007-9098-9. [Online]. Available: https://doi.org/10.1007/s10589-007-9098-9 (visited on 06/18/2020).

V. M. Becerra, "Solving complex optimal control problems at no cost with PSOPT", in *2010 IEEE International Symposium on Computer-Aided Control System Design*, ISSN: 2165-302X, Sep. 2010, pp. 1391–1396. DOI: 10.1109/CACSD.2010.5612676.

References VII

J. T. Betts, *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming*, ser. Advances in Design and Control. Society for Industrial and Applied Mathematics, Jan. 2010, ISBN: 9780898716887. DOI: 10.1137/1.9780898718577. [Online]. Available: https://epubs.siam.org/doi/book/10.1137/1.9780898718577 (visited on 06/15/2020).

D. Garg, M. Patterson, W. W. Hager, A. V. Rao, D. A. Benson, and G. T. Huntington, "A unified framework for the numerical solution of optimal control problems using pseudospectral methods", en, *Automatica*, vol. 46, no. 11, pp. 1843–1851, Nov. 2010, ISSN: 0005-1098. DOI: 10.1016/j.automatica.2010.06.048. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0005109810002980 (visited on 06/18/2020).

References VIII

A. V. Rao, D. A. Benson, C. Darby, M. A. Patterson, C. Francolin, I. Sanders, and G. T. Huntington, "Algorithm 902: GPOPS, A MATLAB software for solving multiple-phase optimal control problems using the gauss pseudospectral method", *ACM Transactions on Mathematical Software*, vol. 37, no. 2, 22:1–22:39, Apr. 2010, ISSN: 0098-3500. DOI: 10.1145/1731022.1731032. [Online]. Available: https://doi.org/10.1145/1731022.1731032 (visited on 06/15/2020).

References IX

B. A. Steinfeldt, M. J. Grant, D. A. Matz, R. D. Braun, and G. H. Barton, "Guidance, Navigation, and Control System Performance Trades for Mars Pinpoint Landing", *Journal of Spacecraft and Rockets*, vol. 47, no. 1, pp. 188–198, Jan. 2010, ISSN: 0022-4650. DOI: 10.2514/1.45779. [Online]. Available: https://arc.aiaa.org/doi/10.2514/1.45779 (visited on 06/18/2020).

C. L. Darby, W. W. Hager, and A. V. Rao, "An hp-adaptive pseudospectral method for solving optimal control problems", en, *Optimal Control Applications and Methods*, vol. 32, no. 4, pp. 476–502, 2011, ISSN: 1099-1514. DOI: 10.1002/oca.957. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/oca.957 (visited on 06/15/2020).

References X

J. Andersson, J. Åkesson, and M. Diehl, "Dynamic optimization with CasADi", in *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, ISSN: 0743-1546, Dec. 2012, pp. 681–686. DOI: 10.1109/CDC.2012.6426534.

T. Fujikawa and T. Tsuchiya, "Enhanced Mesh Refinement in Numerical Optimal Control Using Pseudospectral Methods", *SICE Journal of Control, Measurement, and System Integration*, vol. 7, no. 3, pp. 159–167, 2014. DOI: 10.9746/jcmsi.7.159.

References XI

M. A. Patterson and A. V. Rao, "GPOPS-II: A MATLAB Software for Solving Multiple-Phase Optimal Control Problems Using hp-Adaptive Gaussian Quadrature Collocation Methods and Sparse Nonlinear Programming", *ACM Transactions on Mathematical Software*, vol. 41, no. 1, 1:1–1:37, Oct. 2014, ISSN: 0098-3500. DOI: 10.1145/2558904. [Online]. Available: https://doi.org/10.1145/2558904 (visited on 06/15/2020).

F. Liu, W. W. Hager, and A. V. Rao, "Adaptive mesh refinement method for optimal control using nonsmoothness detection and mesh size reduction", en, *Journal of the Franklin Institute*, vol. 352, no. 10, pp. 4081–4106, Oct. 2015, ISSN: 0016-0032. DOI: 10.1016/j.jfranklin.2015.05.028. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0016003215002045 (visited on 06/15/2020).

References XII

N. A. o. Engineering, *Frontiers of Engineering: Reports on Leading-Edge Engineering from the 2016 Symposium*, en. Jan. 2017, ISBN: 9780309450362. DOI: 10.17226/23659. [Online]. Available: https://www.nap.edu/catalog/23659/frontiers-of-engineering-reports-on-leading-edge-engineering-from-the (visited on 06/18/2020).

D. Garg, M. Patterson, W. Hager, A. Rao, D. R. Benson, and G. T. Huntington, "An overview of three pseudospectral methods for the numerical solution of optimal control problems", Oct. 2017, [Online]. Available: https://hal.archives-ouvertes.fr/hal-01615132 (visited on 06/18/2020).

References XIII

J. K. Moore and A. D. v. Bogert, "Opty: Software for trajectory optimization and parameter identification using direct collocation", en, *Journal of Open Source Software*, vol. 3, no. 21, p. 300, Jan. 2018, ISSN: 2475-9066. DOI: 10.21105/joss.00300. [Online]. Available: https://joss.theoj.org/papers/10.21105/joss.00300 (visited on 06/15/2020).

J. Zhao and T. Shang, "Dynamic Optimization Using Local Collocation Methods and Improved Multiresolution Technique", en, *Applied Sciences*, vol. 8, no. 9, p. 1680, Sep. 2018. DOI: 10.3390/app8091680. [Online]. Available: https://www.mdpi.com/2076-3417/8/9/1680 (visited on 06/15/2020).

References XIV

J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi: A software framework for nonlinear optimization and optimal control", en, *Mathematical Programming Computation*, vol. 11, no. 1, pp. 1–36, Mar. 2019, ISSN: 1867-2957. DOI: 10.1007/s12532-018-0139-4. [Online]. Available: https://doi.org/10.1007/s12532-018-0139-4 (visited on 06/15/2020).

P. Listov and C. Jones, "PolyMPC: An efficient and extensible tool for real-time nonlinear model predictive tracking and path following for fast mechatronic systems", en, *Optimal Control Applications and Methods*, vol. 41, no. 2, pp. 709–727, 2020, ISSN: 1099-1514. DOI: 10.1002/oca.2566. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/oca.2566 (visited on 06/16/2020).

References XV

I. M. Ross, "Enhancements to the DIDO Optimal Control Toolbox", *arXiv:2004.13112 [cs, math]*, Jun. 2020, arXiv: 2004.13112. [Online]. Available: `http://arxiv.org/abs/2004.13112` (visited on 06/18/2020).

CasADi - Docs, [Online]. Available: `https://web.casadi.org/docs/` (visited on 02/18/2020).

A. Lee, "Fuel-efficient Descent and Landing Guidance Logic for a Safe Lunar Touchdown", in *AIAA Guidance, Navigation, and Control Conference*, American Institute of Aeronautics and Astronautics. DOI: 10.2514/6.2011-6499. [Online]. Available: `https://arc.aiaa.org/doi/abs/10.2514/6.2011-6499` (visited on 06/18/2020).

PSOPT/psopt, en. [Online]. Available: `https://github.com/PSOPT/psopt` (visited on 06/15/2020).