NOS-NOC User's Manual

Armin Nurkanović and Moritz Diehl [1]

March, 2022

[1]Department of Microsystems Engineering (IMTEK) and Department of Mathematics, University of Freiburg, Germany
{armin.nurkanovic,moritz.diehl}@imtek.uni-freiburg.de

Contents

1 Introduction

`NOS-NOC` is an open source software package for Nonsmooth Numerical Optimal Control. It is a modular tool based on CasADi [1], IPOPT [2] and MATLAB, for numerically solving Optimal Control Problems (OCP) with piecewise smooth systems (PSS). It relies on the recently introduced Finite Elements with Switch Detection [3] which enables high accuracy optimal control and simulation of PSS. The time-freezing reformulation, which transforms several classes of systems with state jumps into PSS is supported as well. This enables the treatment of a broad class of nonsmooth systems in a unified way. The algorithms and reformulations yield mathematical programs with complementarity constraints. They can be solved with techniques of continuous optimization in a homotopy procedure, without the use of integer variables. The goal of the package is to automate all reformulations and to make nonsmooth optimal control problems practically solvable, without deep expert knowledge.

2 Problem Formulation

The goal is to high accuracy simulation and numerical optimal control algorthims for a general class of piecewise smooth systems (PSS) of the form:

$$\dot{x} = f_i(x,u), \text{ if } x \in R_i \subset \mathbb{R}^{n_x}, \ i \in \mathcal{I} \coloneqq \{1,\ldots,n_f\}, \tag{1}$$

where R_i are disjoint, nonempty, connected and open sets. The functions $f_i(\cdot)$ are assumed to be smooth on an open neighborhood of $\overline{R}_i$ and n_f is a positive integer. The event of $x(\cdot)$ reaching some boundary ∂R_i is called a *switch*. The right hand side (r.h.s.) of (1) is in general discontinuous in x and u is an externally chosen control function.

Several important classes of systems with state jumps can be reformulated into the form of (1) via the *time-freezing* reformulation [4, 5, 6]. Therefore, the focus on PPS enables a unified treatment of several different classes of nonsmooth systems.

Much more mathematical details can be found in paper that introduces `NOS-NOC` [7], the FESD paper [6] and the time-freezing papers [4, 5, 6].

2.1 Stewart's reformulation of the PSS into a Dynamic Complementarity System

To have a computationally more useful we transform the PSS (1) into a Dynamic Complementarity System (DCS). First, to have a meaningful notion of solution we regard the Filippov convexification of (1), which reads as:

$$\dot{x} \in F_{\mathrm{F}}(x,u) = \Big\{ F(x)\theta \mid \sum_{i\in\mathcal{I}} \theta_i = 1, \ \theta_i \geq 0, \ \theta_i = 0 \text{ if } x \notin \overline{R_i}, \forall i \in \mathcal{I} \Big\}, \tag{2}$$

with $\theta = (\theta_1,\ldots,\theta_{n_f}) \in \mathbb{R}^{n_f}$ and $F(x) \coloneqq [f_1(x),\ldots,f_{n_f}(x)] \in \mathbb{R}^{n_x \times n_f}$.

Second, we assume the sets R_i are define by the zero level sets of functions $c_i(x) = 0, i = 1, \ldots, n_c$, which are collected into the vector $c(x) = (c_1(x), \ldots, c_{n_c}(x)) \in \mathbb{R}^{n_c}$. Without lost of generality, the sets are defined as $R_1 = \{x \in \mathbb{R}^{n_x} \mid c_1(x) > 0, \ldots, c_{n_c}(x) > 0\}$, $R_2 = \{x \in \mathbb{R}^{n_x} \mid c_1(x) > 0, \ldots, c_{n_c}(x) < 0\}$, and so on. This can compactly encoded with a sign matrix $S \in \mathbb{R}^{n_f \times n_c}$ which has no repeating rows:

$$S = \begin{bmatrix} 1 & 1 & \ldots & 1 & 1 \\ 1 & 1 & \ldots & 1 & -1 \\ \vdots & \vdots & \ldots & \vdots & \\ -1 & -1 & \ldots & -1 & -1 \end{bmatrix}. \tag{3}$$

Thus the sets R_i are compactly denoted by:

$$R_i = \{x \in \mathbb{R}^{n_x} \mid \operatorname{diag}(S_{i,\bullet})c(x) > 0\}. \tag{4}$$

We use Stewart's reformulation of DCS into PSS [8]. In this case, it is assumed that the sets R_i are represent via the *discriminant functions* $g_i(\cdot)$:

$$R_i = \{x \in \mathbb{R}^{n_x} \mid g_i(x) < \min_{j \in \mathcal{I},\, j \neq i} g_j(x)\}. \tag{5}$$

Using the representation via the sign matrix S in Eq. (4), it can be shown that the function $g : \mathbb{R}^{n_x} \to \mathbb{R}^{n_f}$ whose components are $g_i(x)$ can be found as [3]:

$$g(x) = -Sc(x). \tag{6}$$

Using Stewart's definition of the sets via Eq. (5) the multipliers $\theta(\cdot)$ in the r.h.s. of (2) can be found as a solution of a Linear Program (LP) [8], and (2) is equivalent to

$$\dot{x} = F(x, u)\theta(x), \tag{7a}$$

$$\theta(x) \in \arg\min_{\tilde{\theta} \in \mathbb{R}^{n_f}} \quad g(x)^\top \tilde{\theta} \quad \text{s.t.} \quad e^\top \tilde{\theta} = 1,\ \tilde{\theta} \geq 0. \tag{7b}$$

Usinge the KKT conditions of the LP we can rewrite the last system into the following DCS:

$$\dot{x} = F(x, u)\theta \tag{8a}$$

$$0 = g(x) - \lambda - \mu(t)e, \tag{8b}$$

$$1 = e^\top \theta, \tag{8c}$$

$$0 \leq \theta \perp \lambda \geq 0, \tag{8d}$$

where $\lambda \in \mathbb{R}^{n_f}_{\geq 0}$ and $\mu \in \mathbb{R}$ are the Lagrange multipliers associated to the constraints of the LP (7b) and $z \coloneqq (\theta, \lambda, \mu) \in \mathbb{R}^{n_z}$, $n_z = 2n_f + 1$ are algebraic variables.

For compact notation, we use a C-function $\Phi(\cdot,\cdot)$ for the complementarity conditions and rewrite the KKT conditions of the LP as a nonsmooth equation

$$G_{\mathrm{LP}}(x,z) \coloneqq \begin{bmatrix} g(x) - \lambda - \mu e \\ 1 - e^\top \theta \\ \Phi(\theta,\lambda) \end{bmatrix} = 0. \tag{9}$$

Finally, the Filippov system is equivalent to the following DCS (which can be interpreted as a nonsmooth differential algebraic equation):

$$\dot{x} = F(x,u)\theta, \tag{10a}$$
$$0 = G_{\mathrm{LP}}(x,z). \tag{10b}$$

2.2 The continuous-time optimal control problem

We regard the continuous-time optimal control problem

$$\min_{x(\cdot),u(\cdot),z(\cdot)} \quad \int_0^T f_{\mathrm{q}}(x(t),u(t))\mathrm{d}t + f_{\mathrm{qT}}(x(T)) \tag{11a}$$
$$\text{s.t.} \quad x_0 = \bar{x}_0, \tag{11b}$$
$$\dot{x}(t) = F(x(t),u(t))\theta(t), \qquad t \in [0,T], \tag{11c}$$
$$0 = g(x(t)) - \lambda(t) - \mu(t)e, \qquad t \in [0,T], \tag{11d}$$
$$0 \le \theta(t) \perp \lambda(t) \ge 0, \qquad t \in [0,T], \tag{11e}$$
$$1 = e^\top \theta(t), \qquad t \in [0,T], \tag{11f}$$
$$0 \ge G_{\mathrm{ineq}}(x(t),u(t)), \qquad t \in [0,T], \tag{11g}$$
$$0 \ge G_{\mathrm{T}}(x(t)),, \tag{11h}$$

where $u(t) \in \mathbb{R}^{n_u}$ is the control function, $f_{\mathrm{qT}} : \mathbb{R}^{n_x} \to \mathbb{R}$ is the Mayer term and $f_{\mathrm{q}} : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}$ is the Lagrange objective term. The path and terminal constraints are collected in the functions $G_{\mathrm{ineq}} : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_{g1}}$ and $G_{\mathrm{term}} : \mathbb{R}^{n_x} \to \mathbb{R}^{n_{g2}}$, respectively.

2.3 Standard Implicit Runge-Kutta Discretization of a Single Control Interval

We briefly state how a single control interval is discretized in `NOS-NOC`. Latter we detail how this is extended to obtain FESD. Consider the single control interval $[0,T]$ with a constant externally choose control input q. Suppose the initial value $x_0 = s_0$ to be given. The control interval is divided into N_{fe} finite elements (i.e., integration intervals) $[t_n, t_{n+1}]$ via the grid points $0 = t_0 < t_1 < \ldots < t_{N_{\mathrm{fe}}} = T$.

On each of these intervals an n_{s}-stage Implicit Runge-Kutta (IRK) scheme is applied. An IRK scheme is defined by its Butcher tableau entries $A \in \mathbb{R}^{n_{\mathrm{s}} \times n_{\mathrm{s}}}$, $b \in \mathbb{R}^{n_{\mathrm{s}}}$, $c \in \mathbb{R}^{n_{\mathrm{s}}}$ [9]. The fixed step-size reads as $h_n = t_{n+1} - t_n$, $n = 0, \ldots, N_{\mathrm{fe}} - 1$. The approximation of the differential state at the grid points t_n is denoted by $x_n \approx x(t_n)$. The derivative of state at the stage

points $t_n + c_i h_n,\ i = 1, \ldots, n_\mathrm{s}$, for a single finite element are summarized in the vector $V_n \coloneqq (v_{n,1}, \ldots, v_{n,n_\mathrm{s}}) \in \mathbb{R}^{n_\mathrm{s} \cdot n_x}$. The stage values for the algebraic variables are collected in the vectors: $\Theta_n \coloneqq (\theta_{n,1}, \ldots, \theta_{n,n_\mathrm{s}}) \in \mathbb{R}^{n_\mathrm{s} \cdot n_f}$, $\Lambda_n \coloneqq (\lambda_{n,1}, \ldots, \lambda_{n,n_\mathrm{s}}) \in \mathbb{R}^{n_\mathrm{s} \cdot n_f}$, $M_n \coloneqq (\mu_{n,1}, \ldots, \mu_{n,n_\mathrm{s}}) \in \mathbb{R}^{n_\mathrm{s}}$ and $Z_n \coloneqq (\Theta_n, \Lambda_n, M_n) \in \mathbb{R}^{n_\mathrm{s} \cdot n_z}$. Therefore, the IRK equations for the DCS (10) can be written in a compact *differential* form. All *internal* IRK equations are summarized in in $G_\mathrm{irk}(x_n, Z_n, V_n, h_n, q) = 0$. Therefore, a single IRK integration step interpreted as a nonsmooth discrete-time system reads as

$$x_{n+1} = F_\mathrm{irk}(x_n, V_n, h_n, q), \tag{12a}$$
$$0 = G_\mathrm{irk}(x_n, Z_n, V_n, h_n, q), \tag{12b}$$

where

$$G_\mathrm{irk}(x_n, Z_n, V_n, h_n, q) \coloneqq \begin{bmatrix} v_{n,1} - F(x_n + h_n \sum_{j=1}^{n_\mathrm{s}} a_{1,j} v_{n,j}, q)\theta_{n,1} \\ \vdots \\ v_{n,n_\mathrm{s}} - F(x_n + h_n \sum_{j=1}^{n_\mathrm{s}} a_{n_\mathrm{s},j} v_{n,j}, q)\theta_{n,n_\mathrm{s}} \\ G_\mathrm{LP}(x_n + h_n \sum_{j=1}^{n_\mathrm{s}} a_{1,j} v_{n,j}, z_{n,1}) \\ \vdots \\ G_\mathrm{LP}(x_n + h_n \sum_{j=1}^{n_\mathrm{s}} a_{n_\mathrm{s},j} v_{n,j}, z_{n,n_\mathrm{s}}) \end{bmatrix},$$

$$F_\mathrm{irk}(x_n, V_n, h_n, q) \coloneqq x_n + h_n \sum_{i=1}^{n_\mathrm{s}} b_i v_{n,i}.$$

We remind the reader that h_n in (12) are implicitly fixed by the chosen discretization grid.

The next equations summarize all N_fe IRK steps over the control interval in the same discrete-time system style. The variables for all finite elements of a single control interval are collected in the following vectors $\mathbf{x} = (x_0, \ldots, x_{N_\mathrm{fe}}) \in \mathbb{R}^{(N_\mathrm{fe}+1)n_x}$, $\mathbf{V} = (V_0, \ldots, V_{N_\mathrm{fe}-1}) \in \mathbb{R}^{N_\mathrm{fe} n_\mathrm{s} n_x}$ and $\mathbf{h} \coloneqq (h_0, \ldots, h_{N_\mathrm{fe}-1}) \in \mathbb{R}^{N_\mathrm{fe}}$. For the algebraic variables, we collect the stage values and the newly introduced boundary values into the vectors $\mathbf{\Theta} = (\Theta_0, \ldots, \Theta_{N_\mathrm{fe}-1}) \in \mathbb{R}^{N_\mathrm{fe}(n_\mathrm{s}+1)n_f}$. The vectors $\mathbf{\Lambda} \in \mathbb{R}^{N_\mathrm{fe}(n_\mathrm{s}+1)n_f}$, $\mathbf{M} \in \mathbb{R}^{N_\mathrm{fe}(n_\mathrm{s}+1)}$ are defined accordingly and $\mathbf{Z} = (\mathbf{\Theta}, \mathbf{\Lambda}, \mathbf{M})$. We refer to this as the *standard discreteization*.

$$s_1 = F_\mathrm{std}(s_0, \mathbf{V}, q), \tag{13a}$$
$$0 = G_\mathrm{std}(\mathbf{x}, \mathbf{h}, \mathbf{V}, \mathbf{Z}, q), \tag{13b}$$

where

$$G_{\text{std}}(\mathbf{x},\mathbf{h},\mathbf{V},\mathbf{Z},q) \coloneqq \begin{bmatrix} x_1 - (x_0 + h_0 \sum_{i=1}^{n_\text{s}} b_i v_{0,i}) \\ G_{\text{irk}}(x_0, Z_0, V_0, h_0, q) \\ \vdots \\ G_{\text{irk}}(x_{N_\text{fe}-2}, Z_{N_\text{fe}-2}, V_{N_\text{fe}-2}, h_{N_\text{fe}-2}, q) \\ x_{N_\text{fe}-1} - (x_{N_\text{fe}-2} + h_{N_\text{fe}-2} \sum_{i=1}^{n_\text{s}} b_i v_{N_\text{fe}-2,i}) \\ G_{\text{irk}}(x_{N_\text{fe}-1}, Z_{N_\text{fe}-1}, V_{N_\text{fe}-1}, h_{N_\text{fe}-1}, q) \end{bmatrix},$$

$$F_{\text{std}}(s_0,\mathbf{V},q) \coloneqq x_{N_\text{fe}-1} + h_{N_\text{fe}-1} \sum_{i=1}^{n_\text{s}} b_i v_{N_\text{fe}-1,i}.$$

2.4 FESD Discretization of a Single Control Interval

In Finite Elements with Switch Detection (FESD) scheme the step-sizes h_n are left as degrees of freedom (as first proposed by [10]) such that the grid points t_n can coincide with the switching times. To exploit the additional degrees of freedom and to achieve exact switch detection we introduce additional conditions to the standard IRK equations (13) called *cross complementaries* and *step-equilibration*.

The cross complementaries Conditions for boundary values of $\lambda(\cdot)$ and $\mu(\cdot)$ for $n = 0,\dots,N_\text{fe}-2$ read as:

$$0 = G_{\text{LP}}(x_{n+1}, z_{n,n_\text{s}+1}), \tag{14}$$

The cross complementarity constraint for $n = 0,\dots,N_\text{fe}-1$ read as:

$$\begin{aligned} 0 &= \text{diag}(\theta_{n,m})\lambda_{n,m'}, \\ & m = 1,\dots,n_\text{s},\ m' = 0,\dots,n_\text{s}+1, m \neq m'. \end{aligned} \tag{15}$$

For brevity we have a slight abuse of notation since the first and last boundary point are not needed in this formulation. Due to the nonegativity of $\lambda_{n,m'}$ and $\theta_{n,m}$ the last constraint can be stated in many equivalent ways which are detailed in Section **??**.

With a slight abuse of notation we redefine the vectors that collect the algebraic variables such that the newly introduced boundary values are added as well: $\mathbf{\Theta} = (\Theta_0, \theta_{0,n_\text{s}+1}, \dots, \Theta_{N_\text{fe}-1}) \in \mathbb{R}^{N_\text{fe}(n_\text{s}+1)n_f}$. The vectors $\mathbf{\Lambda} \in \mathbb{R}^{N_\text{fe}(n_\text{s}+1)n_f}$, $\mathbf{M} \in \mathbb{R}^{N_\text{fe}(n_\text{s}+1)}$ are defined accordingly and $\mathbf{Z} = (\mathbf{\Theta}, \mathbf{\Lambda}, \mathbf{M})$. The cross complementarity conditions are collected in:

$$0 = G_{\text{cross}}(\mathbf{x},\mathbf{Z}) \coloneqq \begin{bmatrix} G_{\text{LP}}(x_1, z_{0,n_\text{s}+1}) \\ \vdots \\ G_{\text{LP}}(x_{N_\text{fe}}, z_{N_\text{fe}-2,n_\text{s}+1}) \\ \sum_{i=1}^{n_\text{s}} \sum_{j=1,j\neq i}^{n_\text{s}+1} \theta_{1,i}^\top \lambda_{1,j} \\ \vdots \\ \sum_{i=1}^{n_\text{s}} \sum_{\substack{j=0,\\ j\neq i}}^{n_\text{s}} \theta_{N_\text{fe}-1,i}^\top \lambda_{N_\text{fe}-1,j} \end{bmatrix}. \tag{16}$$

The step-equilibration To avoid spurious degrees of freedom in h_n if now switch occurs the following set of conditions is used:

$$0 = G_{\text{eq}}(\mathbf{h}, \mathbf{Z}) \coloneqq \begin{bmatrix} \eta_1(h_1 - h_0) \\ \vdots \\ \eta_{N_{\text{grid}}-1}(h_{N_{\text{fe}}-1} - h_{N_{\text{fe}}-2}) \\ \sum_{n=0}^{N_{\text{fe}}-1} h_n - T \end{bmatrix}. \tag{17}$$

where

$$\begin{aligned} \sigma_n^{\lambda,\text{B}} &= e^\top \Lambda_{n-1}, \\ \sigma_n^{\lambda,\text{F}} &= e^\top \Lambda_n, \\ \sigma_n^{\theta,\text{B}} &= e^\top \Theta_{n-1}, \\ \sigma_n^{\theta,\text{F}} &= e^\top \Theta_n, \\ \pi_n^{\lambda} &= \text{diag}(\sigma_n^{\lambda,\text{B}})\sigma_n^{\lambda,\text{F}}, \\ \pi_n^{\theta} &= \text{diag}(\sigma_n^{\theta,\text{B}})\sigma_n^{\theta,\text{F}}, \\ \upsilon_n &= \pi_n^{\lambda} + \pi_n^{\theta}. \end{aligned}$$

The joint effect of all components is collected in the product

$$\eta_n = \eta(\Theta_{n-1}, \Lambda_{n-1}, \Theta_n, \Lambda_n) = \prod_{i=1}^{n_f} (\upsilon_n)_i.$$

Heuristic step-equilibration The constraint (17) can due to its nonlinearity slow down the convergence and impair the progress of the homotopy loop. Therefore, we propose several heuristic approach to improve the convergence properties. Instead of having the bilinear terms of (17) as constraints we add for every k the objective the term:

$$\psi_{\text{eq}}(\mathbf{h}, \mathbf{Z}) = \rho_{\text{eq}} \sum_{k=0}^{N-1} \sum_{n=0}^{N_{\text{fe}}-1} \tanh(\eta_{k,n})(h_{k,n+1} - h_{k,n})^2,$$

where $\rho_{\text{eq}} > 0$ and the $\tanh(\cdot)$ bring all η_n to the same scale. An alternative simple heuristic is to add the quadratic cost term

$$\tilde{\psi}_{\text{eq}}(\mathbf{h}, \mathbf{Z}) = \rho_{\text{eq}} \sum_{k=0}^{N-1} \sum_{n=0}^{N_{\text{fe}}-1} (h_{k,n+1} - h_{k,n})^2.$$

However, in this case a too large ρ_{eq} biases towards selecting control inputs that lead to switches on an equidistant grid.

A single FESD step We summarize the developemnts of this subsection which result in the FESD discretization. We use again the same discrete-timer representation and refer to this as the *FESD discretization*.

$$s_1 = F_{\text{fesd}}(s_0, \mathbf{h}, \mathbf{V}, q), \tag{18a}$$
$$0 = G_{\text{fesd}}(\mathbf{x}, \mathbf{h}, \mathbf{V}, \mathbf{Z}, q), \tag{18b}$$

The equation $0 = G_{\text{fesd}}(\mathbf{x}, \mathbf{h}, \mathbf{V}, \mathbf{Z}, q)$ collects all other internal computations including all IRK steps within the regarded control interval:

$$G_{\text{fesd}}(\mathbf{x}, \mathbf{h}, \mathbf{V}, \mathbf{Z}, q) \coloneqq \begin{bmatrix} G_{\text{std}}(\mathbf{x}, \mathbf{h}, \mathbf{V}, \mathbf{Z}, q) \\ G_{\text{cross}}(\mathbf{x}, \mathbf{Z}) \\ G_{\text{eq}}(\mathbf{h}, \mathbf{Z}) \end{bmatrix},$$
$$F_{\text{fesd}}(s_0, \mathbf{h}, \mathbf{V}, q) = F_{\text{irk}}(x_{N_{\text{fe}}-1}, V_{N_{\text{fe}}-1}, h_{N_{\text{fe}}-1}, q)$$

2.5 Discretization of the OCP

Using the FESD (recommended) or standard discretization of a single control interval we can discreitized and OCP with $N \geq 1$ control intervals indexed by k. We assume piecewise constant controls collected in $\mathbf{q} = (q_0, \ldots, q_{N-1}) \in \mathbb{R}^{Nn_u}$. All internal variables of of every control interval are additionally equipped with an index k. On every control interval k we apply a discretization (18) with $N_{\text{fe},k}$ internal finite elements. The state values at the control interval boundaries are collected in $\mathbf{s} = (s_0, \ldots, s_N) \in \mathbb{R}^{(N+1)n_x}$. The following vectors collect all internal variables of all discretization steps: $\mathcal{H} = (\mathbf{h}_0, \ldots, \mathbf{h}_{N-1})$, $\mathcal{Z} = (\mathbf{Z}_0, \ldots, \mathbf{Z}_{N-1})$, $\mathcal{V} = (\mathbf{V}_0, \ldots, \mathbf{V}_{N-1})$, $\mathcal{X} = (\mathbf{x}_0, \ldots, \mathbf{x}_N)$. Finally the discretized version of the OCP (11) reads as:

$$\begin{aligned}
\min_{\mathbf{s},\mathbf{q},\mathcal{X},\mathcal{H},\mathcal{V},\mathcal{Z}} \quad & \sum_{k=1}^{N-1} \hat{f}_{\text{q}}(s_k, \mathbf{x}_k, q_k) + \hat{f}_{\text{qT}}(s_N) \\
\text{s.t.} \quad & s_0 = \bar{x}_0, \\
& s_{k+1} = F_{\text{fesd}}(s_k, \mathbf{V}_k, q_k),\ k = 0, \ldots, N-1, \\
& 0 = G_{\text{fesd}}(\mathbf{x}_k, \mathbf{V}_k, \mathbf{Z}_k, q_k),\ k = 0, \ldots, N-1, \\
& 0 \leq G_{\text{ineq}}(s_k, q_k),\ k = 0, \ldots, N-1, \\
& 0 \leq G_{\text{T}}(s_N),
\end{aligned}$$

where $\hat{f}_{\text{qT}} : \mathbb{R}^{n_x} \times \mathbb{R}^{(N_{\text{fe}}+1)n_s n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}$ and $\hat{f}_{\text{qT}} : \mathbb{R}^{n_x} \to \mathbb{R}$ are the discrteized stage and terminal costs. The path constraint $G_{\text{ineq}}(\cdot)$ in this formulation is evaluated only on the control discretization grid but the possibility to evaluate the constrain on a finer grid is supported as well.

3 Settings and Model

This sections provides details on the user settings and model input.

3.1 All `settings` at a glance

Name	Default value	Possible values	Description
General			
`solver_name`	'solver_fesd'	string	Name of the CasADi function calling the NLP solver.
`use_fesd`	1	$\{0,1\}$	Determine if the FESD scheme from **??** is used. If turned off, a standard IRK schemes for a DCS from Section **??** with a fixed step size is used.
IRK and FESD Settings			
`d`	2	$\{1,\dots,9\}$	Number of stages in IRK scheme.
`collocation_scheme`	'radau'	'radau','legendre'	Choose which IRK Scheme Familiy (and corresponding Butcher Tabelu) is used in the FESD scheme
`cross_complementartiy_mode`	3	$\{1,\dots,9\}$	Determines which form of the cross complementarity, cf. Section **??**.
`gamma_h`	1	$[0,1]$	Determines the lower and upper bound for h_n as $(1-\gamma_h)\frac{T}{N}$ and $(1+\gamma_h)\frac{T}{N}$, respectively.
`lp_initalization`	0	$\{0,1\}$	Solve a parametric LP (9) to get a feasible guess for the algebraic variables $\theta_{n,m}$,$\lambda_{n,m}$ and $\mu_{n,m}$.
`initial_theta`	1	$\mathbb{R}_{\geq 0}$	Initial guess for the convex multiplers θ.

initial_lambda	1	$\mathbb{R}_{\geq 0}$	Initial guess for the dual variable in the DCS λ.
initial_mu	1	$\mathbb{R}_{\geq 0}$	Initial guess for the dual variable in the DCS μ.
MPCC and Homotopy Settings			
mpcc_mode	5	$\{1, \ldots, 10\}$	Choose which MPCC approach is used, cf. Section **??**
comp_tol	10^{-16}	$\mathbb{R}_{>0}$	Stopping criterion for the homotopy loops in therms of the complementarity residual.
objective_scaling_direct	1	$\{0, 1\}$	objective scaling.
sigma_0	1	$\mathbb{R}_{>0}$ and $> \sigma_N$	Initial value for homotopy parameter.
sigma_N	10^{-14}	$\mathbb{R}_{>0}$ and $< \sigma_0$	Final value of the homotopy parameter.
kappa	0.1	$0, 1$	Constant in the homotopy parameter update rule $\sigma_{i+1} = \kappa\sigma i$.
N_homotopy	$\left\lceil \lvert \frac{\log(\frac{\sigma_N}{\sigma_0})}{\log \kappa} \rvert \right\rceil$	$\mathbb{N}$	Maximum number of homotopy parameters.
s_elastic_0	1	$\mathbb{R}_{>0}$	elastic mode.
s_elastic_max	100	$\mathbb{R}_{>0}$	elastic mode.
s_elastic_min	0	$\mathbb{R}_{>0}$	elastic mode.
store_all_homotopy_iterates	1	$\{0, 1\}$	Store the solution of every NLP in the homotopy loop.
Settings for Barrier Based MPCC Heuristic			
Step-Equilibration			
Time Settings			

`time_optimal_problem`	0	0,1	is the given OCP a time optimal OCP? If yes, removes the constraints that the sum of all h_n equals T and adds the parameter T to the objective and to the vector of optimization variables w.
`time_freezing`	0	$\{0, 1\}$	Is time freezing used, this is useful to isolate the clock state and other time transformation variables.
`time_rescaling`	0	$\{0, 1\}$	This enables to extend the bounds of h, so that a desired terminal time can be achieved. If only `time_freezing` is true, speed of time and step size all lumped together, e.g., $\hat{h}_{n,m} = s_n h_{n,m}$, hence the bounds need to be extended."

use_speed_of_time_variables	1	$\{0, 1\}$	"Introduce an extra control (prameter) variable s(.) for speed of time (e.g. for reaching a desreid terminal state for the clock state). This decouple the speed of time from the variable step size. It usually improves convergence. If only time_freezing_time_rescaling is true, then the h_n also make sure to addapt the length of the subintervals, if both. time_freezing and use_speed_of_time_variables are true, new control variables are introduecd, they can be per stage or one for the whole interval."
local_speed_of_time_variable	1	$\{0, 1\}$	Add a speed of time variable s(.) for every control interval.
stagewise_clock_constraint	1	$\{0, 1\}$	Determine is there after every stage a "terminal" constraint for the time that has passed so far, if on. Or if off, just add a final terminal constraint for the clock state.
s_sot0	1	> 0	Initial guess for the speed of time variables.

s_sot_max	25	¿0	Upper bound for the speed of time variables.
s_sot_min	s_sot_max^{-1}	≥ 0	Lower bound for the speed of time varaibles.
impose_terminal_phyisical_time	1	$\{0, 1\}$	This option is only avilable if time-freezing is used. It imposes that the final phyisical time is equl to the provided T (except in time optimal control problems where it is a degree of freedom, then it determines is the phyisical or numericla timize minimized). If it is off, it turns off time_rescaling during time_freezing.
IPOPT Settings			
Integrator Settings			

3.2 The model struct

This table gives an overview of the model data and what is mandatory user input.

Name	Type	Is it mandatory	Description
User input			
N_stages	double	yes	num of control intevrlas.
N_finte_elements	double	yes	num of finite elements.
T	double	yes	control horizon
x0	double	yes	control horizon
x	CasADI SX or MX	yes	state vectors.
u	CasADI SX or MX	yes	control vectors.

`F`	CasADi `sym expr`	yes	Matrix F with all modes.
`S`	MATLAB double	yes	Sign Matrix S modes.
`f_q`	CasADi `sym expr`	no	ggggg.
`f_qT`	CasADi `sym expr`	no	ggggg.
`g_ineq`	CasADi `sym expr`	no	ggggg.
`g_terminal`	CasADi `sym expr`	no	ggggg.
`lbx`	`double` of size $[n_x, 1]$	no	ggggg.
`ubx`	`double` of size $[n_x, 1]$	no	ggggg.
`lbu`	`double` of size $[n_x, 1]$	no	ggggg.
`ubu`	`double` of size $[n_x, 1]$	no	ggggg.
`lb_g_ineq`	`double` of size $[n_x, 1]$	no	ggggg.
`ub_g_ineq`	`double` of size $[n_x, 1]$	no	ggggg.
`lb_g_terminal`	`double` of size $[n_x, 1]$	no	ggggg.
`ub_g_terminal`	`double` of size $[n_x, 1]$	no	ggggg.
Auto generated			
`n_x`	`int`	·	dimension of state vector x.
`n_u`	`int`	·	dimension of state vector x.
`n_z`	`int`	·	dimension of state vector x.
`n_theta`	`int`	·	dimension of state vector x.

3.3 List of examples

In this subsection we give the current list of examples available in the NOS-NOC repository with some details about the problem.

4 Homotopy and MPCC settings

The discrete-time OCP from the last section obtained via direct transcription using FESD is an MPCC. It can be written more compactly as

$$\min_{w} \quad f(w) \tag{19a}$$
$$\text{s.t.} \quad 0 \leq h(w), \tag{19b}$$
$$0 \leq w_1 \perp w_2 \geq 0, \tag{19c}$$

where $w = (w_0, w_1, w_2) \in \mathbb{R}^{n_w}$ is a given decomposition of the problem variables. MPCC are difficult nonsmooth NLP which violate e.g., the MFCQ at all feasible points [11]. Fortunately, they can often be solved efficiently via reformulations

and homotopy approaches [11, 12, 13]. We briefly discuss the standard ways of solving MPCC that are implemented in `NOS-NOC`. They differ in how Eq. (19c) is handled. In all cases $w_1, w_2 \geq 0$ is kept unaltered and the bilinear constraint $w_1^\top w_2 = 0$ is treated differently.

In a homotopy procedure we solve a sequence of more regular, relaxed NLP related to (19) and parameterized by a homotopy parameter $\sigma_i \in \mathbb{R}_{\geq 0}$. Every new NLP is initialized with the solution of the previous one. In all approaches the homotopy parameter is updated via the rule: $\sigma_{i+1} = \kappa\sigma_i$, $\kappa \in (0, 1)$, $\sigma_0 > 0$, where i is the index of the NLP in the homotopy. In the limit as $\sigma_i \to 0$ (or often even for a finite i and σ_i) the solution of the relaxed NLP matches a solution of (19). `NOS-NOC` supports the following approaches:

Smoothing and Relaxation In smoothing the bilinear term is replaced by the simpler constraint $w_1^\top w_2 = \sigma_i$ and in *relaxation* by $w_1^\top w_2 \leq \sigma_i$. A sequence of NLP for a decreasing σ_i is solved and under certain assumptions for $\sigma_i \to 0$ a solution of the initial MPCC (19) is obtained [12].

ℓ_1-Penalty In this approach the bilinear constraint is discarded and the term $\frac{1}{\sigma_i} w_1^\top w_2$ is added to the objective, which is a penalized ℓ_1 norm of the complementarity residual. When the penalty $\frac{1}{\sigma_i}$ exceeds a certain (often finite) threshold we have $w_1^\top w_2 = 0$ and the solution of such an NLP is a solution to (19) [13].

Elastic Mode In *elastic mode* (sometimes called ℓ_∞-approach) [11] a bounded scalar slack variable $\gamma \in [0, \bar{\gamma}]$ is introduced. The relaxed bilinear constraint reads as $w_1^\top w_2 \leq \gamma$ and we add to the objective $\frac{1}{\sigma_i}\gamma$. Variants with $w_1^\top w_2 = \gamma$ and $-\gamma \leq w_1^\top w_2 \leq \gamma$ are supported as well. Once the penalty $\frac{1}{\sigma_i}$ exceeds a certain (often finite) threshold we have $\gamma = 0$ and we recover a solution of (19) [11].

Objective Scaling Furthermore, in the ℓ_1 and ℓ_∞ approaches, *direct* ($f(w) + \frac{1}{\sigma_i}\psi(w)$) and *indirect* ($\sigma_i f(w) + \psi(w)$) objective scaling are available, which are mathematically equivalent but often result in different performance.

Barrier Controlled Penalty Homotopy As controlling the homotopy parameters σ_i might sometimes be difficult we propose a heuristic approach, were the homotopy parameter is indirectly controlled by the barrier parameter control in `IPOPT` [2]. The formulation works as follows. We introduce two scalar slack variables τ and δ. We add the constraint

$$0 \leq \tau \leq \bar{\tau}, \tag{20}$$

$$0 \leq \delta \leq \bar{\delta}, \tag{21}$$

$$\tau \leq e^{-\delta}. \tag{22}$$

and to the objective the term $-\rho_\delta \delta^2$. This objective term will favor larger δ which imply very small values for τ. Intuitevly, as the interoor point interations proceed, τ will be implicitly controlled by the barrier parameter as the constraints get more "active", cf. Fig. **??**. The positive scalars $\bar{\tau}, \bar{\delta}, \rho_\delta$ are tuning parameters. We can now use τ instead of σ_i in all of the reformulations above and solve a single NLP.

5 Cross complementarities

5.1 "Everyone with everyone": full sparsity.

5.1.1 Case 1: Standard form full complementarity ("every one with everyone") vector valued form

$$0 = \mathrm{diag}(\theta_{0,m})\lambda_{0,m'},\ m,\ m' \in \mathcal{I}_{\mathrm{stg}}, \tag{23a}$$
$$0 = \mathrm{diag}(\theta_{n,m})\lambda_{n,m'},\ n \in \mathcal{I}_{\mathrm{grd}} \setminus \{0\}, m \in \mathcal{I}_{\mathrm{stg}}, m' \in \mathcal{I}_{\mathrm{stg}} \cup \{0\}. \tag{23b}$$

5.1.2 Case 2: Standard form full complementarity ("every one with everyone") scalar-valued form

$$0 = \theta_{0,m}^\top \lambda_{0,m'},\ m,\ m' \in \mathcal{I}_{\mathrm{stg}}, \tag{24a}$$
$$0 = \theta_{n,m}^\top \lambda_{n,m'},\ n \in \mathcal{I}_{\mathrm{grd}} \setminus \{0\}, m \in \mathcal{I}_{\mathrm{stg}}, m' \in \mathcal{I}_{\mathrm{stg}} \cup \{0\}. \tag{24b}$$

5.2 Constraints per collocation point: via sums of λ or θ.

5.2.1 Case 3: For every collocation point one vector-valued constraint via sum of all λ.

$$0 = \mathrm{diag}(\theta_{0,m}) \sum_{k=1}^{n_{\mathrm{s}}} \lambda_{0,k},\ m \in \mathcal{I}_{\mathrm{stg}}, \tag{25a}$$
$$0 = \mathrm{diag}(\theta_{n,m}) \sum_{k=0}^{n_{\mathrm{s}}} \lambda_{n,k},\ n \in \mathcal{I}_{\mathrm{grd}} \setminus \{0\}, m \in \mathcal{I}_{\mathrm{stg}}. \tag{25b}$$

Now we do the same via sum of θ for the remaining caseses. Note that here we have one more constraint as we make an extra constraint for $\lambda_{0,n}$.

5.3 Case 4: For every collocation point one vector-valued constraint via sum of all θ.

$$0 = \mathrm{diag}(\lambda_{0,m}) \sum_{k=1}^{n_{\mathrm{s}}} \theta_{0,k},\ m \in \mathcal{I}_{\mathrm{stg}}, \tag{26a}$$
$$0 = \mathrm{diag}(\lambda_{n,m}) \sum_{k=1}^{n_{\mathrm{s}}} \theta_{n,m},\ n \in \mathcal{I}_{\mathrm{grd}} \setminus \{0\}, m \in \mathcal{I}_{\mathrm{stg}} \cup \{0\}. \tag{26b}$$

5.3.1 Case 5: Per collocation point one scalar-valued constraint via sum of λ.

$$0 = \theta_{0,m}^\top \sum_{k=1}^{n_\mathrm{s}} \lambda_{0,k},\ m \in \mathcal{I}_\mathrm{stg}, \tag{27a}$$

$$0 = \theta_{n,m}^\top \sum_{k=0}^{n_\mathrm{s}} \lambda_{n,k},\ n \in \mathcal{I}_\mathrm{grd} \setminus \{0\}, m \in \mathcal{I}_\mathrm{stg}. \tag{27b}$$

5.4 Case 6: For every collocation point one scalar-valued constraint via sum of all θ.

$$0 = \lambda_{0,m}^\top \sum_{k=1}^{n_\mathrm{s}} \theta_{0,k},\ m \in \mathcal{I}_\mathrm{stg}, \tag{28a}$$

$$0 = \lambda_{n,m}^\top \sum_{k=1}^{n_\mathrm{s}} \theta_{n,m},\ n \in \mathcal{I}_\mathrm{grd} \setminus \{0\}, m \in \mathcal{I}_\mathrm{stg} \cup \{0\}. \tag{28b}$$

5.5 One Constraints per stage (Finite Element)

5.5.1 Case 7: Per stage (finite element) one vector valued constraint via sum of λ

$$0 = \sum_{k=1}^{n_\mathrm{s}} \mathrm{diag}(\theta_{n,k}) \sum_{m=1}^{n_\mathrm{s}} \lambda_{0,m}, \tag{29a}$$

$$0 = \sum_{k=1}^{n_\mathrm{s}} \mathrm{diag}(\theta_{n,k}) \sum_{m=0}^{n_\mathrm{s}} \lambda_{n,m},\ n \in \mathcal{I}_\mathrm{grd} \setminus \{0\}, m \in \mathcal{I}_\mathrm{stg}. \tag{29b}$$

5.5.2 Case 8: Per stage (finite element) one scalar constraint via sum of λ.

$$0 = \sum_{k=1}^{n_\mathrm{s}} \theta_{0,k}^\top \sum_{m=1}^{n_\mathrm{s}} \lambda_{0,m}, \tag{30a}$$

$$0 = \sum_{k=1}^{n_\mathrm{s}} \theta_{n,k}^\top \sum_{m=0}^{n_\mathrm{s}} \lambda_{n,m},\ n \in \mathcal{I}_\mathrm{grd} \setminus \{0\}. \tag{30b}$$

The cases one constraint per stage are the same no matter if they are performed via sum of θ or sum of λ.

5.5.3 Vector-Valued

$$0 = \sum_{k=1}^{n_\mathrm{s}} \mathrm{diag}(\theta_{n,k}) \sum_{m=1}^{n_\mathrm{s}} \lambda_{0,m} + \sum_{n=1}^{N_\mathrm{grid}-1} \sum_{k=1}^{n_\mathrm{s}} \mathrm{diag}(\theta_{n,k}) \sum_{m=0}^{n_\mathrm{s}} \lambda_{n,m}. \tag{31}$$

5.5.4 Scalar-Valued

$$0 = \sum_{k=1}^{n_s} \theta_{n,k}^{\top} \sum_{m=1}^{n_s} \lambda_{0,m} + \sum_{n=1}^{N_{\text{grid}}-1} \sum_{k=1}^{n_s} \theta_{n,k}^{\top} \sum_{m=0}^{n_s} \lambda_{n,m}. \tag{32}$$

Remark:The scalar valued constraint seem to work better with equality type relaxations and the vector vlaued ones with inequality type ones.

5.6 Step equilibration

5.7 FESD Integrator

6 A Tutorial Example

6.1 A MATLAB Example of an OCP with a System with State Jumps

```
import casadi.*
% Call this function to have an overview of all options.
[settings] = default_settings_fesd();
% Highlight that the problem is treated with time-freezing reformulation and change the number of IRK stages
settings.time_freezing = 1;
settings.n_s = 3;
% Discretization data
model.T = 5;
model.N_stages = 15;
model.N_finite_elements = 3;
% Define states, controls and inital values
q = MX.sym('q',2); v = MX.sym('v',2); t = MX.sym('t');
model.x = [q;v;t];
model.x0 = [0;0.5;0;0;0];
u = MX.sym('u',2);
model.u = u;
% Define regions of the PSS
model.S = [1; -1];
model.c = q(2);
% Define modes of PSS
f_1 = [v;u(1);u(2)-9.81;1];
f_2 = [0;v(2);0;-10*(q(2))-0.211989*v(2);0];
model.F = [f_1 f_2];
% Stage and terminal costs
model.f_q = u'*u; model.f_q_T = 10*v'*v;
% Path and terminal constraints
```

```
model.g_ineq = u'*u-7^2;
model.g_terminal = q-[4;0.25];
% Solve OCP with MPCC homotopy and plot results
[solver,solver_initalization, model,settings] =
    create_nlp_fesd(model,settings);
[results,stats] = homotopy_solver(solver,model,settings,
    solver_initalization);
plot_result_ball(model,settings,results,stats)
```

7 Structure of the Software

The aim of this section is to give more detail on the structure of `NOS-NOC` and to provide more insight on all functions shipped with this packages. (work in progress...)

References

[1] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi – a software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, vol. 11, no. 1, pp. 1–36, 2019.

[2] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical Programming*, vol. 106, no. 1, pp. 25–57, 2006.

[3] A. Nurkanović, M. Sperl, S. Albrecht, and M. Diehl, "Finite Elements with Switch Detection for Direct Optimal Control of Nonsmooth Systems," *to be submitted*, 2022.

[4] A. Nurkanović, T. Sartor, S. Albrecht, and M. Diehl, "A Time-Freezing Approach for Numerical Optimal Control of Nonsmooth Differential Equations with State Jumps," *IEEE Control Systems Letters*, vol. 5, no. 2, pp. 439–444, 2021.

[5] A. Nurkanović, S. Albrecht, B. Brogliato, and M. Diehl, "The Time-Freezing Reformulation for Numerical Optimal Control of Complementarity Lagrangian Systems with State Jumps," *arXiv preprint*, 2021.

[6] A. Nurkanović and M. Diehl, "Continuous optimization for control of hybrid systems with hysteresis via time-freezing," *Submitted to The IEEE Control Systems Letters (L-CSS)*, 2022.

[7] ——, "Nos-noc: A software package for numerical optimal control of nonsmooth systems," *Submitted to The IEEE Control Systems Letters (L-CSS)*, 2022.

[8] D. E. Stewart, "A high accuracy method for solving ODEs with discontinuous right-hand side," *Numerische Mathematik*, vol. 58, no. 1, pp. 299–328, 1990.

[9] E. Hairer and G. Wanner, *Solving Ordinary Differential Equations II – Stiff and Differential-Algebraic Problems*, 2nd ed. Berlin Heidelberg: Springer, 1991.

[10] B. T. Baumrucker and L. T. Biegler, "MPEC strategies for optimization of a class of hybrid dynamic systems," *Journal of Process Control*, vol. 19, no. 8, pp. 1248–1256, 2009.

[11] M. Anitescu, P. Tseng, and S. J. Wright, "Elastic-mode algorithms for mathematical programs with equilibrium constraints: global convergence and stationarity properties," *Mathematical Programming*, vol. 110, no. 2, pp. 337–371, 2007.

[12] S. Scholtes, "Convergence properties of a regularization scheme for mathematical programs with complementarity constraints," *SIAM Journal on Optimization*, vol. 11, no. 4, pp. 918–936, 2001.

[13] D. Ralph and S. J. Wright, "Some properties of regularization and penalization schemes for mpecs," *Optimization Methods and Software*, vol. 19, no. 5, pp. 527–556, 2004.