

Master's Thesis

---

**Evaluating Methods for Solving  
Mathematical Programs With  
Complementarity Constraints Arising  
From Nonsmooth Optimal Control**

---

Anton E. Pozharskiy

Examiner: Prof. Dr. Moritz Diehl

Advisor: Armin Nurkanović

Albert-Ludwigs-University Freiburg

Faculty of Engineering

Department of Microsystems Engineering

Chair for Systems Control and Optimization

November 6, 2023

**Writing Period**

06. 04. 2023 – 06. 11. 2023

**Examiner**

Prof. Dr. Moritz Diehl

**Second Examiner**

Prof. Dr. Lars Pastewka

**Advisor**

Armin Nurkanović

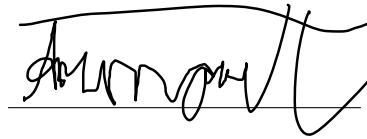
# Declaration

I hereby declare, that I am the sole author and composer of my thesis and that no other sources or learning aids, other than those listed, have been used. Furthermore, I declare that I have acknowledged the work of others by providing detailed references of said work.

I hereby also declare, that my Thesis has not been prepared for another examination or assignment, either wholly or excerpts thereof.

Freiburg, 06.11.2023

Place, Date

A handwritten signature in black ink, consisting of a series of loops and a long horizontal stroke at the top, written over a horizontal line.

Signature



# Abstract

The recently introduced Finite Elements with Switch Detection (FESD) discretization for nonsmooth dynamical systems provides a new source of Mathematical Programs with Complementarity Constraints (MPCCs), which are a particularly difficult form of nonlinear program. This approach has multiple powerful properties that are particularly useful when applied to Optimal Control Problems (OCPs). However in order to be used in this context effectively, fast, robust solvers for MPCCs must exist. This thesis evaluates the performance of various solution methods for MPCCs that arise from the application of FESD to OCPs. The approach of interest is a generally successful class of solution methods which solves MPCCs via a sequence of relaxed nonlinear programs in a homotopy procedure where the relaxation is governed by a parameter that is driven to zero. This thesis introduces a novel benchmark suite, NOSBENCH, with a total set of 603 MPCCs, and uses it to evaluate the performance of various relaxations and other reformulation parameters. It is observed that only 73.8% of problems in the benchmark are solved by the best approaches, often requiring minutes to tens of minutes to converge even for relatively small OCPs. These results highlight the need for further work to be done to improve the state of the art for MPCC solvers. Finally, this thesis evaluates several sparsities of the complementarity constraints generated by FESD and proves that the sparsest form violates certain constraint qualifications at all feasible points. This fact is then augmented by empirically showing that the degeneracy of the sparsest mode indeed affects practical performance.



# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Notation . . . . .	3
1.2	Related Work . . . . .	5
1.2.1	MPCC Solution Methods . . . . .	5
1.2.2	Other Benchmarks . . . . .	7
1.3	Outline of the Thesis . . . . .	8
<b>2</b>	<b>Mathematical Programs with Complementarity Constraints</b>	<b>9</b>
2.1	MPCC Theory . . . . .	9
2.1.1	Stationarity Definitions . . . . .	10
2.1.2	Bouligand Stationarity . . . . .	10
2.2	MPCC Solution Methods . . . . .	15
2.2.1	Relaxation Functions . . . . .	16
2.2.2	“The Price of Inexactness” . . . . .	21
2.2.3	Driving the Relaxation Parameter to Zero . . . . .	22
2.2.4	Lifting Methods . . . . .	25
2.2.5	Parameter Scaling . . . . .	26
<b>3</b>	<b>Nonsmooth Systems</b>	<b>29</b>
3.1	Types of Nonsmooth Systems . . . . .	29
3.1.1	Piecewise Smooth Systems . . . . .	30
3.1.2	Filippov Systems . . . . .	32

3.2	Continuous Time Optimal Control with Dynamic Complementarity Systems . . . . .	33
3.3	FESD and Time-Freezing . . . . .	34
3.3.1	FESD . . . . .	35
3.3.2	The Time-Freezing Reformulation . . . . .	48
3.4	Theoretical Properties of FESD . . . . .	50
3.4.1	Sparse Cross-Complementarity . . . . .	52
3.4.2	Dense Cross-Complementarity . . . . .	54
3.5	NOSNOC . . . . .	54
3.5.1	Tool-chain . . . . .	54
3.5.2	Software Structure . . . . .	58
<b>4</b>	<b>NOSBENCH - An MPCC Benchmark</b>	<b>61</b>
4.1	Problem Format . . . . .	61
4.2	Problem Set . . . . .	62
4.2.1	Original ODEs and OCPs . . . . .	62
4.2.2	Discretization Options . . . . .	65
4.2.3	Problem Naming Scheme . . . . .	68
4.2.4	Problem Subsets . . . . .	68
<b>5</b>	<b>Numerical Experiments</b>	<b>71</b>
5.1	Performance Profiles . . . . .	72
5.2	General Experiment Setup . . . . .	73
5.3	Stopping Criteria and Solution Quality . . . . .	74
5.4	Verification of Methods Against MacMPEC . . . . .	75
5.5	Cross Complementarity Modes . . . . .	77
5.6	Comparing $\sigma$ Scaling vs No $\sigma$ Scaling . . . . .	78
5.7	Improved Relaxation Solver Parameters . . . . .	88
5.7.1	Relaxation Benchmark . . . . .	89
5.7.2	Homotopy Parameters . . . . .	93



5.7.3	NLP Solvers . . . . .	95
5.8	Heaviside Step vs Stewart . . . . .	100
5.9	Lifting vs Direct . . . . .	101
<b>6</b>	<b>Conclusions &amp; Outlook</b>	<b>103</b>
6.1	Conclusions . . . . .	103
6.2	Future Work . . . . .	104
<b>7</b>	<b>Acknowledgments</b>	<b>107</b>
	<b>Bibliography</b>	<b>118</b>
	<b>Appendices</b>	<b>119</b>
<b>A</b>	<b>Additional NOSBENCH Description and Results</b>	<b>121</b>



# 1 Introduction

Nonsmooth dynamical systems make up an extremely common class of models in a variety of fields, but particularly in robotics applications where they can be used to model contacts between rigid objects. However, their use extends beyond that into modeling of electronic circuits, and biological systems [1, 2]. Such systems often pose challenges both for accurate simulation as well as optimal control. For the former, time stepping and switch detection methods are commonly used techniques, but these are not of much use when applied in an optimal control context. This is due to the fact that such techniques are not generally capable of providing accurate sensitivities used in direct optimal control algorithms based on Newton-type methods.

This thesis builds on the work of Nurkanović and co-workers and the Finite Elements with Switch Detection (FESD) method which is a discretization approach for simulation and optimal control problems subject to nonsmooth dynamical systems [3]. The FESD framework transforms Optimal Control Problems (OCPs) subject to Filippov Differential Inclusions (DI) first into OCPs subject to Dynamic Complementarity Systems (DCS), and then into Mathematical Programs with Complementarity Constraints (MPCCs).

MPCCs are a particularly nasty form of Nonlinear Program (NLP):

$$\min_{w \in \mathbb{R}^{n_w}} F(w) \tag{1a}$$

$$\text{s.t.} \quad 0 = g(w), \tag{1b}$$

$$0 \leq h(w), \tag{1c}$$

$$0 \leq G(w) \perp H(w) \geq 0, \tag{1d}$$

which contains constraints of the form in Equation (1d) which lead to significant difficulties. This form contains five functions over the decision variable  $w$ : the objective  $F : \mathbb{R}^{n_w} \rightarrow \mathbb{R}$ , the equality and inequality constraint functions  $g : \mathbb{R}^{n_w} \rightarrow \mathbb{R}^{n_g}$ ,  $h : \mathbb{R}^{n_w} \rightarrow \mathbb{R}^{n_h}$ , and the complementarity functions  $G : \mathbb{R}^{n_w} \rightarrow \mathbb{R}^{n_c}$ ,  $H : \mathbb{R}^{n_w} \rightarrow \mathbb{R}^{n_c}$ . In this case, the Equation (1d) is a combinatorial constraint that implies that only at most one of each corresponding component  $G_i(w)$  and  $H_i(w)$  can be non-negative and the other must be zero. This can algebraically be rewritten as the set of constraints:

$$0 \leq G(w), \tag{2a}$$

$$0 \leq H(w), \tag{2b}$$

$$0 = G_i(w)H_i(w), \quad i = 1, \dots, n_c. \tag{2c}$$

This kind of problem violates the Linear Independence Constraint Qualification (LICQ) at all feasible points. This can be proven via the fact that at any feasible point we have linearly dependent gradients of two of the inequalities described above. The same holds true for the Mangasarian-Fromovitz Constraint Qualification (MFCQ), though the proof for that is marginally more involved and can be found in [4].

The study of such problems in the past has come primarily in the service of bi-level optimization problems, structural optimization, and occasionally in dynamic systems as described by Ferris and Pang [5]. These include work by Bard on convex two-level

optimization [6], packaging problems that minimize membrane surfaces, examples of which can be found in [7], and examples of Stackelberg games such as the one found in [8]. The problems obtained by discretization of an optimal control problem (OCP) via FESD are often much larger and have a higher number of complementarity constraints than those found other MPCCs. They also contain many nonlinear constraints in the form of discretized system dynamics. FESD in its standard, sparse, form also produces problems that may further violate even specifically MPCC tailored constraint qualifications. This makes the problems degenerate, which may complicate their analysis. We discuss some of the theoretical properties of FESD problems in future chapters.

In this thesis, we introduce a new benchmark that is made up of problems generated via the FESD discretization implemented in the software package NOSNOC [9]. This benchmark is available publicly in its entirety in two formats at <https://github.com/apozharski/nosbench>. We then run several experiments across various parameters of the NOSNOC solver, as well as a comparison between existing NLP solvers' performance on the current iteration of the benchmark.

## 1.1 Notation

We now introduce some of the notation we will use in the remaining chapters of this thesis:

**Table 1:** Notation

Symbol	Meaning
$w$	Primal optimization variable
$x$	Differential state
$z$	Algebraic state
$n_w$	Number of primal variables

**Table 1:** Notation

Symbol	Meaning
$n_c$	Number of complementarity pairs
$n_g$	Number of algebraic constraints
$n_x$	Number of differential states
$n_u$	Number of control variables
$n_z$	Number of algebraic states
$n_q$	Number of generalized position states
$n_v$	Number of generalized velocity states
$n_f$	Number of regions with unique dynamics
$N_s$	Number of control stages
$N_{fe}$	Number of Finite Elements per control stage
$n_s$	Number of Runge-Kutta stages per Finite Element
$\theta$	Filippov multipliers
$\alpha$	Heaviside step function values
$R_i$	Regions of a PSS
$f_i(x, u)$	System dynamics in region $R_i$
$F_F(x, u)$	System dynamics in terms of a Filippov differential inclusion
$\phi(x)$	Piecewise smooth system switching functions
$G(w), H(w)$	Complementarity functions
$\psi(a, b, \sigma)$	Relaxation function used to reformulate complementarities
$a, b$	Complementarity arguments $0 \leq a \perp b \leq 0$
$\sigma$	Relaxation homotopy parameter
$\kappa$	Linear update rate of $\sigma$
$\zeta$	Exponential update rate of $\sigma$
$\mathcal{I}_{00}$	Index set of bi-active complementarities
$\mathcal{I}_{+0}$	Index set of complementarities with positive $G$ and zero $H$

**Table 1:** Notation

Symbol	Meaning
$\mathcal{I}_{0+}$	Index set of complementarities with zero $G$ and positive $H$
$\mathcal{F}_{\Omega_{\text{MPCC}}}^{\text{MPCC}}(w)$	MPCC linearized feasible cone
$\mathcal{L}(w, \lambda, \mu, \nu, \xi)$	MPCC Lagrangian

## 1.2 Related Work

This thesis' primary contribution is the introduction of a new benchmark consisting of a particular class of MPCCs. This is done primarily to evaluate deficiencies in existing solution methods when comes to solving the problems that arise from discretizing OCPs via the FESD method. In this section, we cover related work on solution methods for MPCCs and benchmarking efforts for other classes of problem.

### 1.2.1 MPCC Solution Methods

There exists significant literature on various solution methods for MPCCs, though often this comes in the form of methods for Mathematical Programs with Equilibrium Constraints (MPECs) which are NLPs with Variational Inequalities (VIs). A vast array of approaches using Nonlinear Complementarity Functions (NCPs) can be found in Facchinei and Pang [8]. Other approaches include direct penalty methods such as those suggested in [10]. So called direct methods that solve the MPCC as an NLP are also often applied, where the complementarity constraints in Equation (1d) are replaced with an equality constraint ( $G_i(x)H_i(x) = 0$ ) or a relaxation of that constraint to an inequality ( $G_i(x)H_i(x) \leq 0$ ) [11]. There is also existing work by Anitescu that proposes elastic mode style algorithms (as are implemented in SQP methods) for solving MPCCs [12].

The primary method explored in this thesis is solving MPCCs via solving multiple, related, NLPs. When it comes to regularization/relaxation methods there is a variety of proposed methods that have differing theoretical properties that have been proposed in [13, 14, 15, 16, 17], to name a few. These can all be used to generate relaxed NLPs with a regularization parameter  $\sigma_k$  and solved in a homotopy with  $\sigma_k \rightarrow 0$ . These methods include the use of various regularization techniques with varying levels of theoretical guarantees on what points they converge to. However, due to the relative ease of existing benchmarks (which are discussed in the next section) there is not a significant amount of comparison of these approaches on larger MPCCs. We also discuss briefly the lifting methods as proposed by Stein [18] and extended by Izmailov et al. [19] which attempt to project the degenerate complementarity constraints into a 3rd dimension via an auxiliary variable and additional equality constraints. If one limits themselves to just quadratic programs with only linear complementarity constraints, i.e.,  $G(w) = Aw$ , and  $H(w) = Bw$ , there is a specifically tailored solver, LCQPow, developed by Hall et al [20] that uses a sequential convex programming approach.

Beyond those solution methods directly benchmarked there are several other classes of approaches that have been proposed. Interesting recent work related to solving MPECs with a Non-Interior Point approach has been done by Lin and Ohtsuka [21] which extends similar work done by Hotta and Yoshise [22] for non-linear MPCCs and earlier by Chen and Harker [23] which focused on solving Linear Programs with Complementarity Constraints. A separate class of approaches has also been proposed, most recently by Kirches et al. [24] with an algorithm that uses sequential linearization of an MPCC as a sub-problem for an Augmented Lagrangian scheme. This builds upon a filterSQP extension proposed by Leyffer and Munson [25] which introduces an additional element to the standard filterSQP to specifically handle the complementarity conditions. An even earlier proposed algorithm is an “ $\epsilon$ -active-set” method by Fukushima and Tseng [26]. This approach solves a series of MPCCs within a fixed but slightly relaxed active set at each iterate to generate a descent



direction. A common factor that makes evaluation of these algorithms difficult is that for many of these proposed algorithms there often does not exist a performant, publicly available, implementation.

Finally, there is a completely separate set of approaches that attempt to take advantage of the inherent combinatorial structure of MPCCs and attempt to solve them by branching on the active sets of each complementarity and attempting to solve these as Mixed-Integer Nonlinear Programming problems (MINLP). These are somewhat similar to the work in [24], [25], and [27], as they also use the combinatorial structure to fix the active set of a given problem.

### 1.2.2 Other Benchmarks

There does not exist a particularly large cohort of existing benchmarks focused on MPCCs or MPECs. The primary set of problems used is a benchmark introduced by Sven Leyffer in 2000 called MacMPEC which contains 193 problems of various sizes [28]. Less commonly used and cited is the MPEC library published by GAMS [29]. Both of these benchmarks consist primarily of problems taken from operations research and economics along with problems coming from elasto-statics and structural engineering. Conversely, one can find a wide variety of benchmarks for other classes of problems such as QPs [30], Semi-definite programming [31, 32], and NLPs [33, 34].

In present literature there are several examples of solvers that use MacMPEC as a source for validation of performance on MPCCs and MPECs including filterSQP [35], and several Augmented Lagrangian (AL) approaches to solving such problems such as ALGENCAN as benchmarked by Izmailov et al. and more recently a variety of AL methods by Guo and Deng [27, 36]. These benchmarks all tend to follow the interpretation guidelines set forth by Dolan and Moré in their article on performance profiles [37]. The article introduces an approach to interpreting the results of optimization benchmarks via a more quantitative approach to data visualization which

we describe in Section 5.1 and utilize during our discussion of experimental results in Chapter 5.

### 1.3 Outline of the Thesis

This thesis is structured as follows. We first describe sufficient mathematical background on the theory underpinning MPCCs and their solution methods in Chapter 2. This is followed by a brief discussion of the theory behind nonsmooth dynamical systems, their difficulties, and the details of FESD, in Chapter 3. This chapter also briefly mentions the structure of NOSNOC itself and the tool-chain that it implements. Chapter 4 then first describes the structure of the benchmark as well as a brief explanation of all included problems. Chapter 5 begins with a discussion of performance profiles introduced by Dolan and Moré and their use in visualizing the results of the various experiments run [37]. This is followed by a description of experiments run against this benchmark, and their results are discussed in detail. We conclude with an overview of the results along with proposed future work in Chapter 6.

## 2 Mathematical Programs with Complementarity Constraints

In this chapter, we cover the background on Mathematical Programs with Complementarity Constraints (MPCCs) starting with tailored stationarity concepts and constraint qualifications. We then move on to a description of the primary solution method evaluated in this thesis: solving MPCCs via a relaxation of the complementarity constraints. To this end, we cover a myriad of existing relaxation functions and several approaches to driving the relaxation parameter to zero. We then include a brief description of lifting methods. Finally we discuss an attempt at re-scaling the relaxed region of the problem, such that it's relative size is normalized for all of the approaches.

### 2.1 MPCC Theory

Due to the previously discussed degeneracy of MPCCs as a class of NLPs we cannot use the standard first-order necessary conditions and Karush-Kuhn-Tucker (KKT) conditions to describe solutions to such problems. Therefore, more specialized conditions have been devised to describe the stationarity for MPCCs. In this section we describe those stationarity conditions as well as the set of auxiliary NLPs that can be used to classify the stationary points of the MPCC. We also discuss the inherently

combinatorial nature of MPCCs as well as the standard combinatorial stationarity conditions known as Bouligand Stationarity (B-Stationarity).

### 2.1.1 Stationarity Definitions

It is important to note that most of the theoretical difficulties in solving MPCCs stem from bi-active constraints, i.e. when for  $0 \leq a \perp b \leq 0$  the solution lies in the point where  $a = 0$  and  $b = 0$ . On the other hand, when a solution to an MPCC lies in points where either  $a > 0$  or  $b > 0$  for all complementarity conditions (a condition named “strict complementarity”), standard KKT-conditions can be shown to be necessary for optimality. The stationarity conditions of MPCCs have been described as both a “zoo” and “alphabet-soup” by researchers in the field. This is due to the variety of conditions which can cause solvers to halt on NLP formulations of the MPCC, even if those points have trivial descent directions.

In order to treat the various conditions that follow we first define the index sets:

$$\mathcal{I}_{00} = \mathcal{I}_{00}(w) = \{ i | G_i(w) = 0, H_i(w) = 0 \}, \quad (3a)$$

$$\mathcal{I}_{0+} = \mathcal{I}_{0+}(w) = \{ i | G_i(w) = 0, H_i(w) > 0 \}, \quad (3b)$$

$$\mathcal{I}_{+0} = \mathcal{I}_{+0}(w) = \{ i | G_i(w) > 0, H_i(w) = 0 \}, \quad (3c)$$

for a  $w$  that is feasible for the original form of the MPCC in Equation (1).

### 2.1.2 Bouligand Stationarity

In order to define the most generic stationarity concept for MPCCs, Bouligand-Stationarity (B-Stationarity) we first define the MPCC specific first order linearized

feasible cone at a feasible point  $w$ :

$$\begin{aligned}\mathcal{F}_{\Omega_{\text{MPCC}}}^{\text{MPCC}}(w) = \{d \in \mathbb{R}^{n_w} \mid & \nabla g(w)^\top d = 0, \\ & \nabla h_i(w)^\top d \geq 0, \text{ for all } i \in \mathcal{A}(w), \\ & \nabla G_i(w)^\top d = 0, \text{ for all } i \in \mathcal{I}_{0+}(w), \\ & \nabla H_i(w)^\top d = 0, \text{ for all } i \in \mathcal{I}_{+0}(w), \\ & 0 \leq \nabla G_i(w)^\top d \perp \nabla H_i(w)^\top d \leq 0, \text{ for all } i \in \mathcal{I}_{00}(w)\}.\end{aligned}$$

We note that, unlike the standard linearized feasible cone used in the theory for NLPs, this cone is necessarily non-convex.  $w^*$  is considered algebraically B-stationary if it is feasible and it holds that:

$$\nabla f(w^*)^\top d \geq 0, \quad \forall d \in \mathcal{F}_{\Omega_{\text{MPCC}}}^{\text{MPCC}}(w^*). \quad (4)$$

It turns out that if  $\mathcal{F}_{\Omega_{\text{MPCC}}}^{\text{MPCC}}(w^*)$  is equal to the true tangent cone at  $w^*$ , then this condition is necessary for optimality. This is called “geometric B-stationarity” but this is less computationally useful.

### Other Stationarity Concepts for MPCCs

In this section we will discuss multiplier based stationarity concepts for an MPCC of the form in Equation (1). To handle this problem the literature introduces several auxiliary NLPs, the KKT conditions of which are useful in characterizing the stationarity conditions of the MPCC. These are the so called tightened NLP (TNLP):

$$\min_w F(w) \tag{5a}$$

$$\text{s.t. } 0 = g(w), \tag{5b}$$

$$0 \leq h(w), \tag{5c}$$

$$0 = G_i(w), \quad i \in \mathcal{I}_{00} \cup \mathcal{I}_{0+}, \tag{5d}$$

$$0 \leq G_i(w), \quad i \in \mathcal{I}_{+0}, \tag{5e}$$

$$0 = H_i(w), \quad i \in \mathcal{I}_{00} \cup \mathcal{I}_{+0}, \tag{5f}$$

$$0 \leq H_i(w), \quad i \in \mathcal{I}_{0+}, \tag{5g}$$

and the relaxed NLP (RNLP):

$$\min_w F(x) \tag{6a}$$

$$\text{s.t. } 0 = g(w), \tag{6b}$$

$$0 \leq h(w), \tag{6c}$$

$$0 = G_i(w), \quad i \in \mathcal{I}_{0+}, \tag{6d}$$

$$0 = H_i(w), \quad i \in \mathcal{I}_{+0}, \tag{6e}$$

$$0 \leq G_i(w), \quad i \in \mathcal{I}_{00} \cup \mathcal{I}_{+0}, \tag{6f}$$

$$0 \leq H_i(w), \quad i \in \mathcal{I}_{00} \cup \mathcal{I}_{0+}. \tag{6g}$$

### Stationarity Conditions

The above, along with a definition of an MPCC Lagrangian, which is the standard Lagrangian for the TNLP/RNLP:

$$\mathcal{L}(w, \lambda, \mu, \nu, \xi) = F(w) - \lambda \cdot g(w) - \mu \cdot h(w) - \nu \cdot G(w) - \xi \cdot H(w), \tag{7}$$

allow for the the definition of “weak stationarity” (W-stationarity) conditions for Equation (1):

$$\nabla_w \mathcal{L}(w^*, \lambda^*, \mu^*, \nu^*, \xi^*) = 0, \quad (8a)$$

$$g(w^*) = 0, \quad (8b)$$

$$0 \leq \mu^* \perp h(w^*) \geq 0, \quad (8c)$$

$$G(w^*) \geq 0, \nu_i^* = 0, \quad i \in \mathcal{I}_{+0}, \quad (8d)$$

$$H(w^*) \geq 0, \xi_i^* = 0, \quad i \in \mathcal{I}_{0+}, \quad (8e)$$

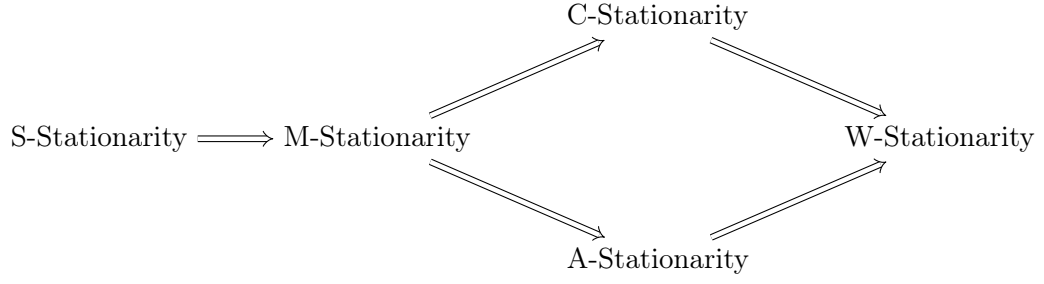
$$G(w^*) = 0, \nu_i^* \geq 0, \quad i \in \mathcal{I}_{00} \cup \mathcal{I}_{0+}, \quad (8f)$$

$$H(w^*) = 0, \xi_i^* \geq 0, \quad i \in \mathcal{I}_{00} \cup \mathcal{I}_{+0}. \quad (8g)$$

Via simple comparison of the KKT conditions we see that this W-stationarity is equivalent to stationarity in the TNLP. We can also see that the KKT conditions of the RNLP correspond to Equation (8a) with non-negativity constraints on  $\nu^*$  and  $\xi^*$  (Equation (8f) and Equation (8g)). This condition is called “strong stationarity” (S-Stationarity).

With both ends of the spectrum of stationarity described we can dive deeper into the alphabet soup of stationarity conditions. All of these: Clarke (C), Mordukhovich (M), and Abadie (A) stationarity conditions can be defined as constraints on the multipliers  $\nu^*$  and  $\xi^*$ , along with the other W-stationarity conditions.

1. Abadie stationarity is characterized by the condition that for all  $i \in \mathcal{I}_{00}$  either  $\nu_i \geq 0$  or  $\nu_i \geq 0$ .
2. Clarke stationarity is characterized by the condition that for all  $i \in \mathcal{I}_{00}$ ,  $\nu_i \xi_i \geq 0$ .
3. Mordukhovich stationarity is characterized by the condition that for all  $i \in \mathcal{I}_{00}$  either  $\nu_i > 0 \wedge \xi_i > 0$  or  $\nu_i \xi_i = 0$ .



**Figure 1:** Stationarity implication graph.

It is important to note that for problems that exhibit strict complementarity we see that each of these definitions collapses to the same conditions as  $\mathcal{I}_{00}$  is the empty set. In cases without strict complementarity, the implications in Figure 1 hold which can easily be observed from the inequality conditions on the multipliers.

### Constraint Qualifications

Here we briefly mention the constraint qualifications that are relevant to the relaxation methods we discuss in the next section.

- (i) MPCC-LICQ is satisfied for an MPCC of the form Equation (1) at point  $w^*$  iff for the inequality active set  $\mathcal{I}_h(w^*)$  the gradients:

$$\nabla g_i(w^*), \quad i = 1, \dots, n_g, \quad (9a)$$

$$\nabla h_i(w^*), \quad i \in \mathcal{I}_h(w^*), \quad (9b)$$

$$\nabla G_i(w^*), \quad i \in \mathcal{I}_{00}(w^*) \cup \mathcal{I}_{0+}(w^*), \quad (9c)$$

$$\nabla H_i(w^*), \quad i \in \mathcal{I}_{00}(w^*) \cup \mathcal{I}_{+0}(w^*), \quad (9d)$$

are linearly independent.

- (ii) MPCC-MFCQ is satisfied for an MPCC of the form Equation (1) at point  $w^*$



iff for the inequality active set  $\mathcal{I}_h(w^*)$  the gradients:

$$\nabla g_i(w^*), \quad i = 1, \dots, n_g, \quad (10a)$$

$$\nabla G_i(w^*), \quad i \in \mathcal{I}_{00}(w^*) \cup \mathcal{I}_{0+}(w^*), \quad (10b)$$

$$\nabla H_i(w^*), \quad i \in \mathcal{I}_{00}(w^*) \cup \mathcal{I}_{+0}(w^*), \quad (10c)$$

are linearly independent and there exists a vector  $d \in \mathbb{R}^{n_w}$  such that:

$$\nabla g_i(w^*)^\top d = 0, \quad i = 1, \dots, n_g, \quad (11a)$$

$$\nabla h_i(w^*)^\top d < 0, \quad i \in \mathcal{I}_h(w^*), \quad (11b)$$

$$\nabla G_i(w^*)^\top d = 0, \quad i \in \mathcal{I}_{00}(w^*) \cup \mathcal{I}_{0+}(w^*), \quad (11c)$$

$$\nabla H_i(w^*)^\top d = 0, \quad i \in \mathcal{I}_{00}(w^*) \cup \mathcal{I}_{+0}(w^*), \quad (11d)$$

These definitions can be found in more detail, along with other weaker constraint qualifications, in [38].

## 2.2 MPCC Solution Methods

In this section, we discuss the solution methods evaluated in the experiments described in Chapter 5. To this end we first describe the relaxation methods that are implemented in NOSNOC and briefly mention their theoretical properties. This is followed by a discussion of the unfortunate reality that many of these theoretical properties do not necessarily exist in practice due to the limitations of imprecise algebras. We then discuss several approaches to driving the relaxation parameter to zero, and a normalization approach which we use to reduce the biases caused by the different relaxations. Finally the lifting methods of Stein, and Izmailov et al. are presented.

### 2.2.1 Relaxation Functions

The primary solution method tackled in this thesis is the relaxation approach which solves a series of relaxed NLPs:

$$\min_w F(w) \tag{12a}$$

$$\text{s.t. } 0 = g(w), \tag{12b}$$

$$0 \leq h(w), \tag{12c}$$

$$0 \leq G(w), \tag{12d}$$

$$0 \leq H(w), \tag{12e}$$

$$0 \leq \psi(G(w), H(w), \sigma), \tag{12f}$$

which we call  $\text{NLP}(\sigma)$ . Note this is not the same NLP as the RNLP which is used in the stationairity definitions. The relaxation function  $\psi(G(w), H(w), \sigma)$  is assumed to approach the true feasible set of  $0 \leq G(w) \perp H(w) \geq 0$  as  $\sigma$  approaches 0. There have been many such functions proposed in the literature with various theoretical properties proven about convergence of this homotopy procedure under various assumptions. In the remainder of this section we discuss the relaxation functions that are implemented in the NOSNOC software package and that are evaluated in Chapter 5.

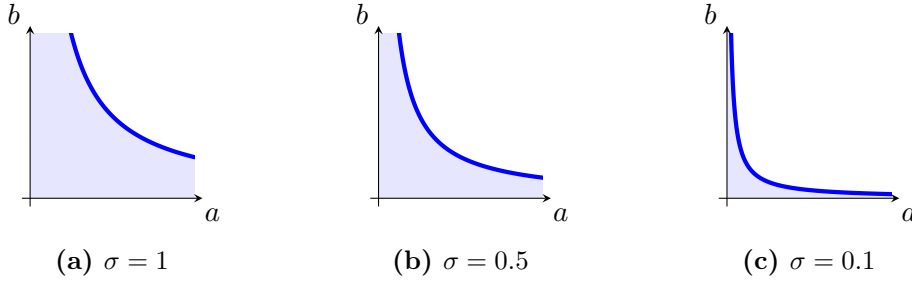
#### Scholtes' Global Relaxation

One of the earliest and simplest global relaxations was studied by Scholtes in [13]. The relaxation function in this case is defined as:

$$\psi_S(a, b, \sigma) = ab - \sigma \tag{13}$$

Theorem 3.1 of the Scholtes paper states that a sequence  $x_k$  which are solutions of  $\text{NLP}(\sigma_k)$  converging to  $x^*$ , which is guaranteed to be a C-stationary point under the

assumption that MPCC-LICQ holds at  $x^*$ . This sequence is also guaranteed to have unique multipliers for each  $k$ . If the MPCC-LICQ requirement is relaxed to only include MPCC-MFCQ it can still be shown that that  $x^*$  is a C-stationary point, but we lose the multiplier uniqueness. It can be further shown that if strict complementarity holds then the points converged to by this sequence will be B-Stationary points of the MPCC.



**Figure 2:** Visualization of the Scholtes relaxation of  $0 \leq a \perp b \leq 0$ .

### Smoothed Nonlinear Complementarity Functions

These three functions fall into the class of smooth nonlinear complementarity functions (NCPs). It is unclear if the convergence properties derived by Scholtes necessarily hold for any arbitrary NCP, however some work has been done by Jiang and Ralph that shows convergence for particular formulations using these functions [39]. [8] further proves convergence to C-stationary points. However these results rely on similar, relatively strong, assumptions on the non-degeneracy of the point that the sequence converges to as the proof for the Scholtes relaxation does. It is important to note that given a rescaling of the  $\sigma$  parameter the level sets of these three NCPs are equivalent to that of the Scholtes relaxation.

$$\psi_{\text{NR}}(a, b, \sigma) = \frac{a + b - \sqrt{(a - b)^2 + \sigma^2}}{2} \quad (14)$$

$$\psi_{\text{FB}}(a, b, \sigma) = a + b - \sqrt{a^2 + b^2 + \sigma^2} \quad (15)$$

$$\psi_{\text{CCK}}(a, b, \sigma) = \alpha \left( a + b - \sqrt{a^2 + b^2 + \sigma^2} \right) + (1 - \alpha)(ab - \sigma) \quad (16)$$

For the Chen-Chen-Kanzow function we use  $\alpha = 0.5$  as is done in the paper that introduced this function [40].

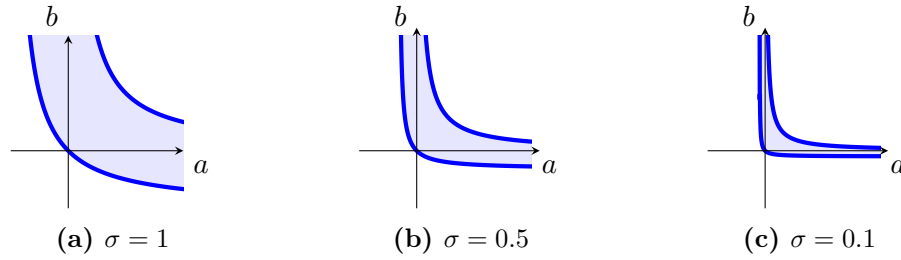
### The Lin-Fukushima Relaxation

The Lin-Fukushima relaxation relaxes both sides of the inequality pair that makes up the complementarity set by replacing  $0 \leq a \perp b \geq 0$  with the constraints:

$$ab \leq \sigma^2 \quad (17)$$

$$(a + \sigma)(b + \sigma) \geq \sigma^2 \quad (18)$$

This relaxation is shown to have the same theoretical guarantees as the Scholtes relaxation under MPCC-LICQ and MPCC-MFCQ [41].



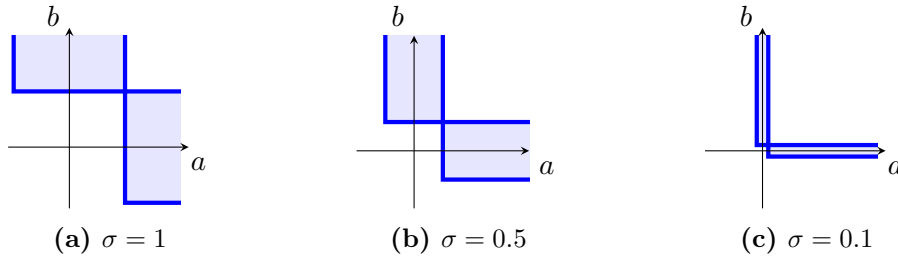
**Figure 3:** Visualization of the Lin-Fukushima relaxation of  $0 \leq a \perp b \geq 0$ .

### Kadrani's Nonsmooth Relaxation

The relaxation proposed by Kadrani et. al [16] produces a non-smooth feasible set with a combination of the function:

$$\psi_K(a, b, \sigma) = (a - \sigma)(b - \sigma), \quad (19)$$

and relaxing the strict non-negativity of the complementarities  $G(x) \geq 0$  and  $H(x) \geq 0$  with  $G(x) \geq -\sigma$ ,  $H(x) \geq -\sigma$ . From simple observation it is clear that the near disjointness of this approach may lead to poor results in cases where the initial solution is not on the correct orthant. However this work has some of the best theoretical guarantees available and concludes with a convergence guarantee to M-stationary points under MPCC-LICQ of the accumulating point which can actually be reduced to only MPCC-CPLD.



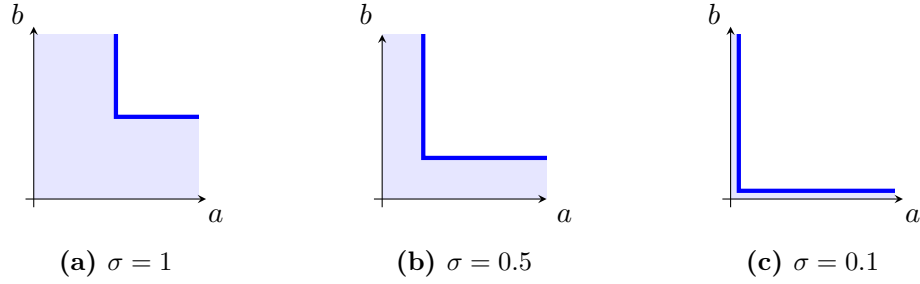
**Figure 4:** Visualization of the Kadrani relaxation of  $0 \leq a \perp b \geq 0$ .

### The Nonsmooth Kanzow-Schwartz Relaxation

The Kadrani relaxation was further improved by Kanzow and Schwartz [17] by using a similarly non-smooth relaxation, but dealing with the near-disjointness problem of the original. This is accomplished with a piecewise function that one can essentially view as a shift of the original set along the  $(1, 1)$  ray:

$$\psi_{\text{KS}}(a, b, \sigma) = \begin{cases} (a - \sigma)(b - \sigma), & a + b - 2\sigma \geq 0 \\ -0.5((a - \sigma)^2 + (b - \sigma)^2), & a + b - 2\sigma < 0 \end{cases} \quad (20)$$

This relaxation maintains the theoretical properties of the Kadrani relaxation and can be shown to converge to M-stationary points under MPCC-CPLD.



**Figure 5:** Visualization of the Kanzow-Schwartz relaxation of  $0 \leq a \perp b \leq 0$ .

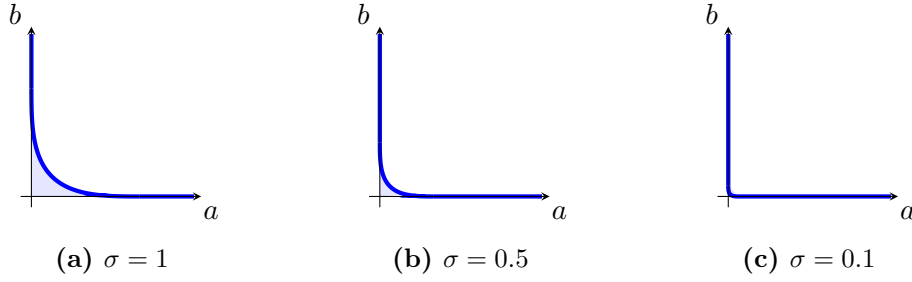
### The Steffensen-Ulbrich Local Relaxation

As opposed to the global relaxation of all the prior options, the approach proposed by Steffensen and Ulbrich relaxes only the “problematic” region of the problem, i.e., only the origin. This is done by generating a function that maintains equality with the complementarity constraints for any point  $a + b \geq \sigma$ , and smoothly transitions to the relaxed portion (i.e. has continuous derivatives) at  $(a, b) = (0, \sigma)$  and  $(a, b) = (\sigma, 0)$ . The local relaxation is then formed by a function in Cartesian coordinates  $q = a + b$ ,  $v = a - b$  and satisfy certain conditions including convexity and differentiability which can be found in [42]. This allows for a whole family of relaxations including the two we implement in NOSNOC which we describe below: Let  $x = a - b$  and  $z = \frac{x}{\sigma}$ .

$$\psi_{\text{SU1}}(a, b, \sigma) = a + b - \begin{cases} |x| & |x| \geq \sigma, \\ \sigma \left[ \frac{2}{\pi} \sin \left( (z + 3) \frac{\pi}{2} \right) + 1 \right], & |x| < \sigma \end{cases} \quad (21)$$

$$\psi_{\text{SU1}}(a, b, \sigma) = a + b - \begin{cases} |x| & |x| \geq \sigma, \\ \frac{\sigma}{8} (-z^4 + 6 * z^2 + 3), & |x| < \sigma \end{cases} \quad (22)$$

This relaxation can be shown to converge to a C-stationary point under MPCC-CPLD assumptions, which is a marginally stronger statement than that which can be made for the Scholtes relaxation [42].



**Figure 6:** Visualization of the Steffensen-Ulbrich relaxation of  $0 \leq a \perp b \leq 0$ .

### 2.2.2 “The Price of Inexactness”

It is important to note that all of the above theoretical guarantees come from proofs reliant on solving the problem,  $\text{NLP}(\sigma_k)$ , to an exact KKT point. The effects of solving KKT systems inexactly, in finite arithmetic, and using that as a sequence of points converging to a true solution to the MPCC  $w^*$  was studied in detail by Kanzow and Schwartz [43]. This is a crucial caveat to much of the theory when applied to implemented solvers due to the general inability of NLP solvers to provide anything other than an  $\epsilon$  approximation to a KKT point as encoded by termination conditions. It has also played out frequently in practice, that the simpler Scholtes and Scholtes-style NCP relaxations have performed better in practical applications than their theory would suggest, which is a result that we will echo in Section 5.7.1.

Kanzow and Schwartz provide a proof that, under some mild assumptions on the contraction of  $\epsilon_k$  (the tolerance to which the approximate KKT point is solved to) with regards to  $\sigma_k$ , both the Scholtes and Lin-Fukushima [44] relaxations still converge to C-stationary points under MPCC-MFCQ. In the case of the Steffensen-Ulbrich relaxation, the convergence to even a C-stationary point relies on further conditions being placed on the intermediate points of the homotopy. In particular these conditions geometrically suggest that near-bi-active points treated by the local relaxation must disappear after some point in the sequence  $x_k$ . Similar (but more complex) conditions on  $x_k$  are also derived for both the Kadrani and Kanzow-Schwartz

kinked relaxations, without which only W-stationarity is guaranteed for  $x^*$ .

### 2.2.3 Driving the Relaxation Parameter to Zero

This section covers three different methods of driving the relaxation parameter to zero, which allows us to recover the solution to the original MPCC. We note that for methods other than the standard homotopy there is only limited theory for our exact use of these methods, but we mention existing convergence properties when relevant.

#### Standard Homotopy

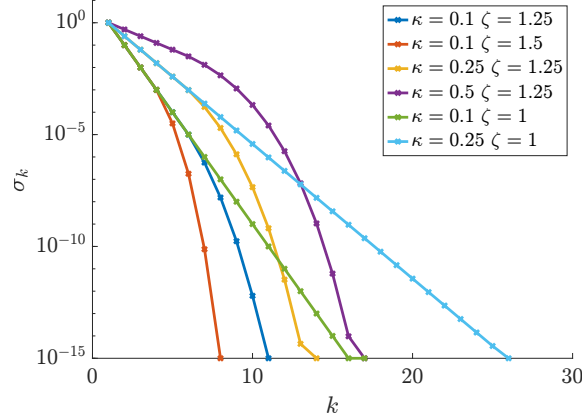
The standard approach to getting a solution of the MPCC, which is equivalent to the solution of  $\text{NLP}(0)$ , is to solve a sequence of  $\text{NLP}(\sigma_k)$  with the sequence  $\sigma_k \downarrow 0$ . This is the approach taken by the vast majority of the published relaxation approaches discussed in Section 2.2.1, and is the basis of the theory that leads to the convergence properties described in that section. The exact method of obtaining  $\sigma_{k+1}$  from  $\sigma_k$  is generally not restricted by the theory, however, in the literature, the most common implementation that exists is the update rule:  $\sigma_{k+1} = \kappa\sigma_k$ . In this scheme,  $\kappa \in (0, 1)$  is a free meta-parameter which determines the rate at which  $\sigma_k$  approaches zero in a geometric fashion. Examples are  $\kappa = 0.2$  in Kadrani et. al [16],  $\kappa = 10^{-2}$  in Scholtes [13], and  $\kappa = 0.1$  in Steffensen and Ulbrich [42].

There exists a further improvement to this approach that involves a second parameter  $\zeta \in \mathbb{R}^+$  and a new update rule:

$$\sigma_{k+1} = \min(\kappa\sigma_k, \sigma_k^\zeta), \tag{23}$$

which provides an “acceleration” to the convergence rate of  $\sigma_k$  to zero. We call this the “superlinear” update rule, and the original  $\sigma_{k+1} = \kappa\sigma_k$  the “linear” update rule.





**Figure 7:** Different sequences of  $\sigma_k$  using the superlinear update rule.

In practice we occasionally observe better performance from this as the active set of the solution is often identified and no longer changes after  $\sigma_k$  reaches a small enough value. We plot the trajectory of  $\sigma_k$  for several values of  $\kappa$  and  $\zeta$  in Figure 7.

### $\ell_\infty$ -Mode

The first alternative approach to the standard homotopy is what we will call the  $\ell_\infty$ -mode, due to its similarity to the  $\ell_\infty$ -exact penalty formulation. The NLPs we solve in this case come in the form:

$$\min_{w \in \mathbb{R}^w, s \in \mathbb{R}} F(w) + \frac{1}{\sigma} s \quad (24a)$$

$$\text{s.t.} \quad 0 = g(w), \quad (24b)$$

$$0 \leq h(w), \quad (24c)$$

$$0 \leq G(w), \quad (24d)$$

$$0 \leq H(w), \quad (24e)$$

$$0 \leq \psi(G_i(w), H_i(w), s), \quad i = 1, \dots, n_c, \quad (24f)$$

$$0 \leq s \leq \sigma. \quad (24g)$$

Note that when  $\psi = \psi_S$  we obtain the  $\ell_\infty$ -exact penalty formulation of the MPCC [45]. For other choices of  $\psi$  the NLP still uses the  $\ell_\infty$ -exact penalty formulation for a problem where the orthogonality constraint is replaced with the equality constraint  $\psi(a, b, 0) = 0$ . As such, we would expect that for all the relaxations, this would be a valid approach to steer the relaxation parameter to 0. In some similar formulations, such as the elastic mode NLP proposed by Anitescu [12], there is also an upper bound set on the slack variable  $s$  as a guiding constraint. We combine the ideas of the standard homotopy approach and  $\ell_\infty$ -mode approach by adding the upper bound constraint  $s \leq \sigma$  to sequentially shrink the upper bound along with increasing the penalty at each iteration.

### $\ell_1$ -Mode

The second alternative steering mode we call the  $\ell_1$ -mode, once again due to its similarity to the corresponding exact penalty formulation:

$$\min_{w, s \in \mathbb{R}^{n_c}} F(w) + \frac{1}{\sigma} \sum_{i=1}^{n_c} s_i \quad (25a)$$

$$\text{s.t.} \quad 0 = g(w), \quad (25b)$$

$$0 \leq h(w), \quad (25c)$$

$$0 \leq G(w), \quad (25d)$$

$$0 \leq H(w), \quad (25e)$$

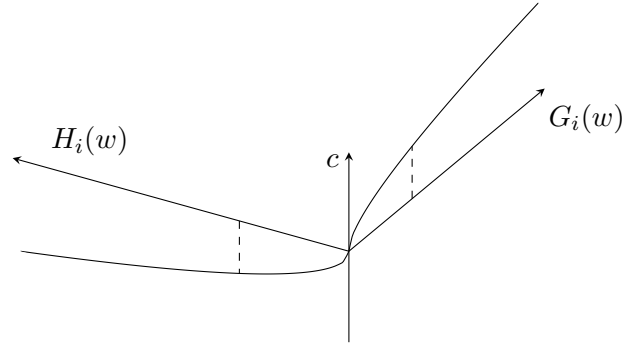
$$0 \leq \psi(G_i(w), H_i(w), s_i), \quad i = 1, \dots, n_c, \quad (25f)$$

$$0 \leq s \leq \sigma. \quad (25g)$$

Again this is equivalent to the standard  $\ell_1$ -exact penalty reformulation when using the Scholtes relaxation. This approach also has more examples in the literature than the  $\ell_\infty$  version, for example Fukushima et al. present an algorithm using an  $\ell_1$  penalized Fischer Burmeister function [46]. The exact  $\ell_1$ -penalty formulation for MPCCs was

described by Anitescu and has been proven to converge to S-stationary points of the MPCC for a large enough penalty parameter  $\frac{1}{\sigma}$  under MPCC-LICQ [11]. In the same paper it is however noted that a theoretical difficulty arises when the B-stationary points of the MPCC do not coincide with S-stationary points (i.e. stationary points of the RNLP). In such cases, the required penalty parameter grows infinitely and as such makes the numerical properties of this method suspect. It is also shown in the literature by Leyffer et al. [47] that the iterates of a penalty method such as this can be shown to correspond to that of the Scholtes relaxation method following the same pattern of  $\{\sigma_k\}$ .

### 2.2.4 Lifting Methods



**Figure 8:** Representation of the lifted feasible set.

We also briefly experiment with an alternate approach to handling the degeneracy caused by the orthogonality constraint as proposed by Stein and extended by Izmailov et al. [18, 19]. The cornerstone of this approach is to replace the nonsmooth, ill-conditioned, orthogonality constraint in 2 dimensions, with a set of constraints in 3 dimensions that encode the same behavior when projected back down into two dimensions. The proposed lifting involves a function  $s : \mathbb{R} \rightarrow \mathbb{R}$  which satisfies:

- (P1)  $s(c) = 0$  over the negative reals,
- (P2)  $s(c)$  is non-negative over the positive reals,

(P3)  $s$  is surjective from the set of non-negative reals to the set of non-negative reals,

(P4)  $s$  is in at least  $\mathcal{C}^1$ .

An example of such a function would be  $s(c) = \max(0, c)^k$  for any  $k$ . In his paper Stein suggests  $k = 2$  as an option and in Izmailov et al.  $k = 3$  is used. This  $s(c)$  can then be used to replace  $0 \leq G_i(w) \perp H_i(w) \geq 0$  with one additional variable  $c$  and 2 constraints:

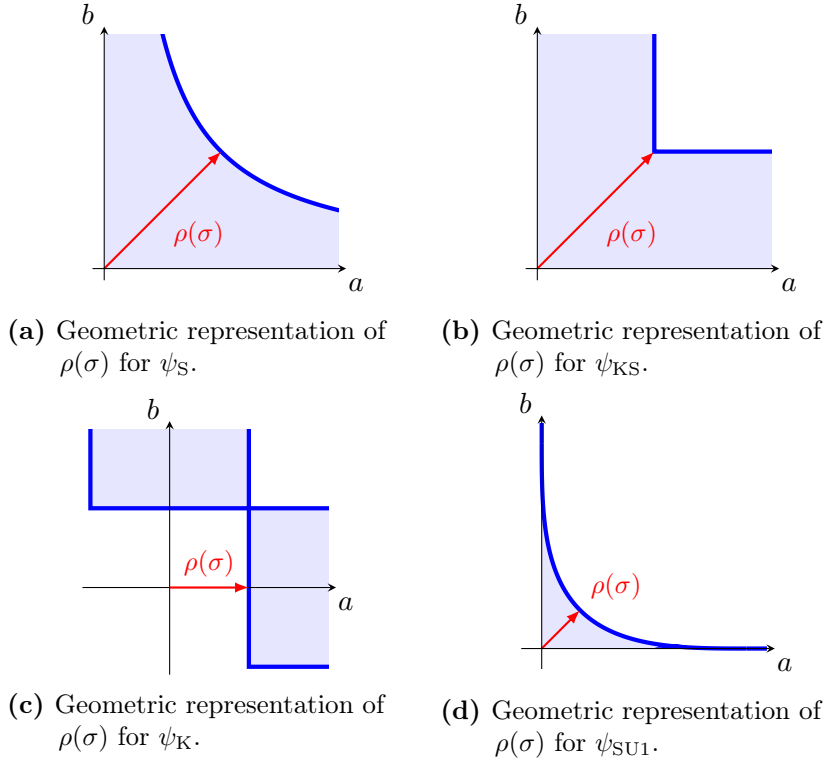
$$\Psi_1(G_i(w), H_i(w), c) = G_i(w) - s(c) = 0, \quad (26a)$$

$$\Psi_2(G_i(w), H_i(w), c) = H_i(w) - s(-c) = 0. \quad (26b)$$

This approach has moderately worse theoretical properties than the relaxation methods described above and can only be shown to converge to C-stationary points under MPCC-LICQ. Further, as we will discuss in Section 5.9, it is quite sensitive to initial infeasibility as Stein himself comments on in his paper. This makes it a difficult approach to use for FESD problems as providing a feasible initial guess that is somewhat close to a local minimum is difficult due to the often complex dynamics of the systems we treat.

### 2.2.5 Parameter Scaling

In this section, we discuss one potential way of equalizing a particular property of the various relaxations in order to better compare them based on their intrinsic properties rather than based on a particular choice of  $\sigma_0$  and  $\kappa$ . We introduce a metric for each relaxation function  $\psi(a, b, \sigma)$  which measures the level of relaxation from the L-shaped set of an MPCC as a function of  $\sigma$ :  $\rho(\sigma)$ . One can theoretically choose any number of values derived from  $\sigma$  to represent this, but in this case we choose the simplest one: the distance to the level set  $\psi(a, b, \sigma) = 0$  to the origin. A graphical representation of this is shown in Figure 9. As one can see this function depends



**Figure 9:** Geometric representations of  $\rho(\sigma)$ .

heavily on the exact structure of the relaxation, but is in all cases trivial to solve for. We list the  $\rho(\sigma)$  functions for all of the relaxation functions in Table 2.

Relaxation	$\rho(\sigma)$
Scholtes	$\sqrt{2\sigma}$
Fischer-Burmeister	$\sigma$
Natural Residual	$\frac{\sigma}{2}$
Chen-Chen-Kanzow	$\frac{\sigma}{2} + \frac{\sqrt{2\sigma}}{2}$
Steffensen-Ulbrich (trig)	$\left( \frac{\frac{2\sqrt{2}}{\pi} \sin(\frac{3\pi}{2}) + 1}{2} \right) \sigma$
Steffensen-Ulbrich (poly)	$\left( \frac{3\sqrt{2}}{16} \right) \sigma$
Kadrani	$\sigma$
Kanzow-Schwartz	$\sqrt{2\sigma}$
Lin-Fukushima	$\sigma$

**Table 2:** Table of  $\rho(\sigma)$  functions.

It is proposed to scale the parameter  $\sigma$  in each relaxation function to equilibrate the rate at which  $\rho(\sigma)$  approaches zero. We note the structure of all of the  $\rho(\sigma)$  other than Scholtes is some constant  $r$  times  $\sigma$ . Arbitrarily we choose to scale all of the methods to have the same  $\rho(\sigma)$  as the Scholtes relaxation. This can be trivially done by scaling the input by taking the square root of the relaxation parameter  $\sigma$  and scaling it by the appropriate constant  $\frac{\sqrt{2}}{r}$ . An important thing to note is doing this for the kinked relaxations increases the worst case complementarity violation for any given  $\sigma$  more significantly due to the limits of these relaxations do not approach the true L-shaped set as  $a$  or  $b$  tend to infinity. In the case of the Chen-Chen-Kanzow function we only rescale the  $\sigma$  component of the part that contributes the  $\frac{\sigma}{2}$  portion of  $\rho(\sigma)$ . This approach is discussed in Section 5.7.1 where we evaluate the relative performance of the scaled and unscaled approaches.

## 3 Nonsmooth Systems

This chapter first covers some rudimentary background on nonsmooth systems, their behaviors, and the numerical difficulties they pose. We then follow this with a brief overview of the newly introduced Finite Elements with Switch Detection (FESD) discretization method and Time-Freezing reformulation, which can be used to accurately discretize the aforementioned nonsmooth systems. Finally, this chapter ends with a description of NOSNOC [9], the open-source tool that implements these methods, and gives a brief overview of its structure and functionality.

### 3.1 Types of Nonsmooth Systems

All of the problems discussed in this thesis come from discretization of nonsmooth dynamic systems, i.e, systems of Ordinary Differential Equations (ODEs) whose right hand side (r.h.s) is defined by a vector field that is not necessarily differentiable or even continuous. This leads to these types of systems to be difficult to treat numerically without the development of additional theory to handle the difficulties posed. These types of systems can broadly be separated into three classes based on the characteristics of the right hand side (r.h.s) of the ODE, and the behavior of it's solutions:

(NSD1) Systems with continuous solutions but whose derivatives are nonsmooth.

One of the simplest examples of these kinds of system are ones that

contain absolute values, e.g,  $\dot{x} = |x|$  which, due to the nonsmoothness of the derivatives already renders the accuracy guarantees for standard integrators no longer valid.

(NSD2) This class contains ODEs of the form  $\dot{x} = f(x)$  where the r.h.s  $f(x)$  is discontinuous, but the solutions to the ODE is continuous. These types of systems make up the core of what NOSNOC is designed to handle. They exhibit a variety of interesting behaviors such as sliding modes and spontaneous switches which will be discussed in the section on Filippov systems.

(NSD3) These ODEs contain state jump laws, i.e, solutions to them may not be continuous in time. These are the most pathological of the three classes. Examples of this are systems that model impacts such as the classic elastic bouncing ball. As discussed in [48] these systems can be used when modeling systems with discrete states.

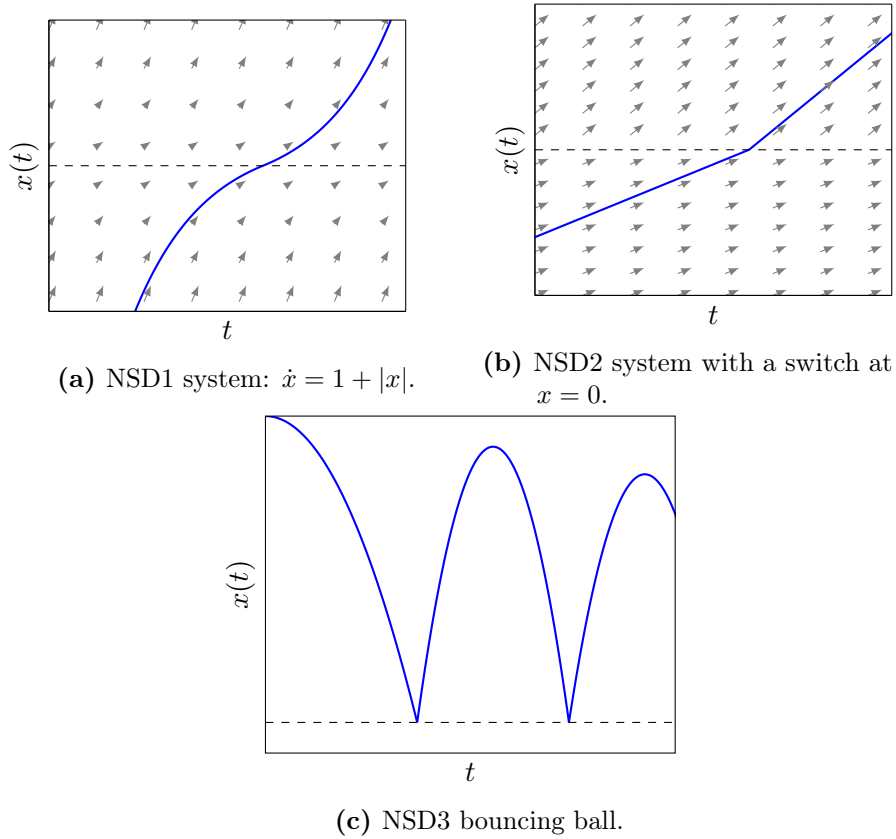
An example of each of these is given in Figure 10 Each of these system types are difficult or impossible to treat accurately with standard fixed step integration methods, and as such require the special handling that is provided by the FESD discretization.

#### 3.1.1 Piecewise Smooth Systems

The first formalism that we will use for NSD1 and NSD2 nonsmooth systems is the concept of a Piecewise Smooth System (PSS). As we are interested in controlled systems we will define everything in the section with a known control signal  $u(t)$ . Such a system with an index set  $\mathcal{I} = \{1, \dots, n_f\}$  is defined generally as:

$$\dot{x} = f_i(x(t), u(t)), \text{ for } x(t) \in R_i \subset \mathbb{R}^{n_x}, \quad i \in \mathcal{I}, \quad (27)$$





**Figure 10:** Types of nonsmooth systems.

over a set of disjoint open sets  $R_i$ . The boundaries of these sets  $\partial R_i$  is assumed to be piecewise smooth as well, and the closure of the sets is the whole set  $\mathbb{R}^{n_x}$ ,

$$\mathbb{R}^{n_x} = \overline{\bigcup_{i \in \mathcal{I}} R_i}, \quad R_i \cap R_j = \emptyset,$$

where  $i \neq j$ , which can intuitively be interpreted as the sets fully tiling the set  $\mathbb{R}^{n_x}$ . We take the functions  $f_i \in \mathcal{C}^2$  to be Lipschitz continuous on the corresponding set  $R_i$ .

This generic formulation is quite powerful and has been applied as a model to a variety of real world systems. A good overview of the theory of PSS and some examples can be found in [49]. However, due to the nonsmoothness of the ODE, standard

initial value problem theory does not generally apply. This along with several other motivating factors leads to the tools in the next section.

### 3.1.2 Filippov Systems

The systems described in Equation (27), can produce a very rich behavior, however that particular formalism does not define well the behavior of the system on the boundary between sets. For example we can take the example piecewise smooth system:

$$\dot{x} = -\text{sign}(x) + 0.5 \sin(t). \quad (28)$$

The interesting behavior we see occurs at  $x = 0$ , where  $\text{sign}(0) = 0$  leading to the dynamics  $\dot{x} = 0.5 \sin(t)$ . Intuitively one would expect this system to be stable at the point  $x = 0$  as the vector fields at  $x \pm \epsilon$  point towards 0. However with the simple PSS formulation this behavior is not recoverable, and we obtain that  $0.5 \sin(t) = 0$  which is clearly an untenable state. Applying numerics to this problem will yield unwanted results and artifacts such as numerical chattering.

As such we need a different formalism to consistently define the analytical behavior of such systems. One such formalism is embedding the ODE into a differential inclusion (DI) as is proposed by Filippov. A Filippov DI can generically be written as:

$$\dot{x} \in F(x(t), u(t)) = \bigcap_{\delta > 0} \bigcap_{\mu(N)=0} \overline{\text{conv}} f(x + \delta \mathcal{B}(x) \setminus N, u(t)), \quad (29)$$

where  $\mathcal{B}(x)$  is the unit ball,  $\mu(N)$  the Lebesgue measure, and  $\overline{\text{conv}}$  is the closed convex hull of a set. Applying Equation (29) to the well behaved sets from Equation (27), yields:

$$\dot{x} \in \left\{ \sum_{i \in \mathcal{I}} \theta_i f_i(x, u) \mid \sum_{i \in \mathcal{I}} \theta_i = 1, \theta_i \geq 0, \theta_i = 0 \text{ for } x \notin \bar{R}_i \right\}, \quad (30)$$

where  $\mathcal{I}(x)$  is the “active set” at a given point  $x \in \mathbb{R}^{n_x}$ . The active set in this case

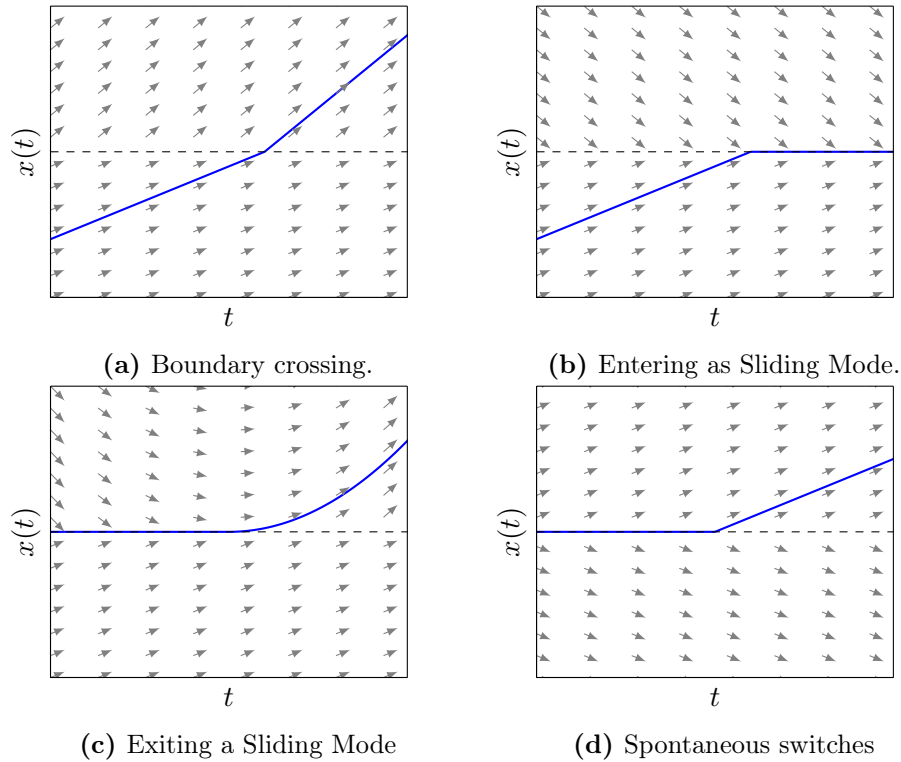
indicates which of the regions of the piecewise smooth system  $R_i$  have an effect on the state dynamics. On the interior of each region  $R_i$  the derivative  $\dot{x}$  is trivially the set of cardinality one defined as  $\{f_i(x, u)\}$ . Due to this structure we can write the Filippov DI via convex multipliers  $\theta_i(x, u)$  as in Equation (30).

From the definition in Equation (30) we can see the variety of rich behavior Filippov systems are capable of encoding. Depending on the behavior of the vector fields at a boundary between two regions we can see one of four different outcomes in the evolution of the system:

- Boundary crossing occurs when the vector fields on one side of the boundary points towards the boundary and the other points away as in Figure 11a.
- Entering (or staying in) a sliding mode, shown in Figure 11b, occurs when the vector field on both sides of the boundary points towards it.
- Exiting a sliding mode due to a change in the solution map for  $\theta$ . This occurs as shown in Figure 11c when the vector field conditions change from the one required for a sliding mode to one that is required for a boundary crossing.
- Spontaneously exiting a sliding mode shown in Figure 11d. This case only occurs when the vector field on both sides of the boundary points away. We call this behavior “spontaneous” as there are infinitely many solutions to an initial value problem that begins on such a boundary.

## 3.2 Continuous Time Optimal Control with Dynamic Complementarity Systems

The normal process of direct optimal control is to first discretize the OCP and obtain a finite dimensional version of the OCP as an NLP.



**Figure 11:** Behaviors of Filippov Systems.

### 3.3 FESD and Time-Freezing

The core of the recent development in the optimal control of nonsmooth systems has been the introduction of FESD, which is itself based on the work of Baumrucker and Biegler [50]. This approach is necessary due to the limitations of standard discretization methods when applied to nonsmooth systems. In particular, these standard approaches are limited to  $O(h)$  accuracy where  $h$  is the fixed step size of the discretization. They further fail to produce correct numerical sensitivities [51].

The FESD discretization is developed to overcome these challenges. In particular we apply it to systems whose dynamics can be represented as Filippov systems, however as we discuss in later subsections, similar concepts can be extended to more generic Differential Inclusions (DIs), as well as directly to some classes of DCS. An optimal

control problem of this type can be written generically as:

$$\min_{x(\cdot), u(\cdot)} \int_0^T L(x, u) dt + E(x(T)) \quad (31a)$$

$$\text{s.t.} \quad x(0) = x_0, \quad (31b)$$

$$\dot{x}(t) \in F_F(x(t), u(t)), \quad t \in [0, T], \quad (31c)$$

$$0 \geq h(x(t), u(t)), \quad t \in [0, T], \quad (31d)$$

$$0 \geq r(x(T)), \quad (31e)$$

where  $x \in \mathbb{R}^{n_x}$  is the state of the system,  $x_0$  is its initial conditions, and  $u \in \mathbb{R}^{n_u}$  is the control signal. The system is subject to the Filippov DI  $F_F(x, u)$ , and  $L(x, u)$  and  $E(x(T))$  are the Lagrange and Mayer cost terms. We also include path constraints in  $h(x, u)$  and terminal constraints  $r(x)$ .

### 3.3.1 FESD

We now introduce the two primary types of reformulation for Equation (31c) that have been described that transform the set inclusion into a more computationally friendly form. This is followed by a description of the concept of “cross-complementarity” and “step-equilibration” both of which are crucial to the FESD algorithm.

#### Stewart’s Reformulation

The first way we will discuss of reformulating the Filippov system was first introduced by Stewart in [52]. It involves embedding the calculation of the Filippov multipliers  $\theta$  as a Linear Program (LP). This on first look seems to only transfer the difficulty caused by the set definition to the necessity of solving an optimization problem. However, we can simply replace the LP with it’s KKT conditions as they are both necessary and sufficient conditions for optimality.

Stewart's reformulation assumes that the regions  $R_i$  of the Filippov system are given by:

$$R_i = \left\{ x \in \mathbb{R}^{n_x} \left| g_i(x) < \min_{j \neq i} g_j(x) \right. \right\}, \quad (32)$$

which can be interpreted as all points  $x$  for which some “indicator function”  $g_i(x)$  is lower than all the other indicator functions. This may at first seem like an inconvenient restriction but actually turns out to be sufficiently powerful to describe most systems. One can calculate these indicator functions from a much more intuitive format: a set of switching functions  $\phi(x) : \mathbb{R}^{n_x} \rightarrow \mathbb{R}^{n_f}$  and a so called “sign” matrix  $S \in \mathbb{R}^{n_f \times n_f}$ , which at least in the case of the Stewart's reformulation must be a dense matrix containing only the elements 1 or  $-1$ . At this point it can be shown that Stewart's indicator function can simply be calculated as  $g(x) = -S\phi(x)$  [53].

Given this assumption one can reformulate Equation (31c) by the single valued dynamics  $\dot{x} = \sum_{i=1}^{n_f} \theta_i f_i(x, u)$  and  $\theta$  which is a member of the solution map of the LP:

$$\theta \in \arg \min_{\tilde{\theta} \in \mathbb{R}^{n_f}} \sum_{i=1}^{n_f} g_i(x) \tilde{\theta}_i \quad (33a)$$

$$\text{s.t.} \quad 0 \leq \tilde{\theta}, \quad (33b)$$

$$1 = \sum_{i=1}^{n_f} \tilde{\theta}_i, \quad (33c)$$

at  $x$ .

We introduce a simplification of notation and define the matrix

$$F(x, u) = [f_1(x, u), \dots, f_{n_f}(x, u)] \in \mathbb{R}^{n_x \times n_f}.$$

The KKT conditions of Equation (33) can then be written along with the dynamics

to form a Dynamic Complementarity system as discussed in Section 3.2:

$$\dot{x} = F(x, u)\theta, \quad (34a)$$

$$0 = g(x) - \lambda - \mu e, \quad (34b)$$

$$0 \leq \theta \perp \lambda \geq 0, \quad (34c)$$

$$1 = \sum_{i=1}^{n_f} \theta, \quad (34d)$$

where  $\lambda \in \mathbb{R}^{n_f}$  and  $\mu \in \mathbb{R}$  are the Lagrange multipliers and are introduced as algebraic variables into the DCS.

We note that the system of algebraics is square in terms of  $\theta$ ,  $\lambda$ , and  $\mu$  and as such admit a unique solution under reasonable assumptions.

It is important to the development of FESD to note the properties of the algebraic variables  $\theta$  and  $\lambda$  that enter the complementarity in Equation (34c). As described in more detail in [53] we can show that  $\lambda$  is a continuous function of time whereas  $\theta$  may be discontinuous when the trajectory interacts with the boundaries between the sets  $R_i$ . It can also be shown that the DAE formed for a given “active set” of the DCS, i.e.  $\mathcal{I}(x(t)) = \{i | \theta_i(t) > 0\}$ , has a unique solution for given initial conditions  $x(0) = x_0$ .

### Heaviside Step Function Reformulation

An alternate way to reformulate the Filippov system comes in the form described by Nurkanović et al. in [54]. This uses the step valued Heaviside step function:

$$\gamma(x) = \begin{cases} \{1\}, & x > 0, \\ [0, 1], & x = 0, \\ \{0\}, & x < 0, \end{cases} \quad (35)$$

which can, alternatively, be represented as an LP:

$$\gamma(x) = \arg \min_{\alpha \in \mathbb{R}} -\alpha x \quad (36a)$$

$$\text{s.t.} \quad 0 \leq \alpha \leq 0. \quad (36b)$$

In this reformulation we again use the switching function  $\phi(x)$  and sign matrix  $S$ , as it was described in the previous section, to describe the regions:

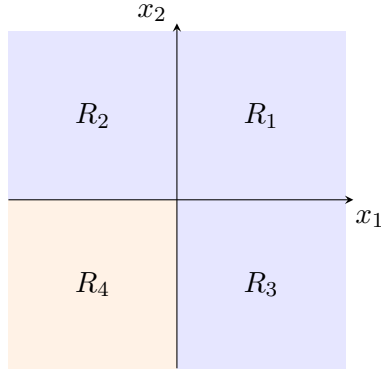
$$R_i = \{ x \in \mathbb{R}^{n_x} | S_{i,\bullet} \phi(x) > 0 \}, \quad (37)$$

where However, in this case we relax the constraints on the elements of  $S$  to allow the inclusion of zeros, making the  $S$  matrix sparse. The motivation for this reformulation is contingent on this sparsity as it allows a more compact representation of certain classes of regions. It is important to note that we still require the constraint on the  $R_i$  in Equation (30). A particular motivating example here can be seen in Figure 12 where using Stewart's reformulation we require 4 separate regions with their own algebraic variables and duplicated dynamics. On the other hand, with the Heaviside Step reformulation we can combine  $R_1$ ,  $R_2$  and  $R_3$ , using only 2 regions, with  $\phi(x) = [x_1, x_2]^\top$  and a sparse

$$S = \begin{bmatrix} 1 & 0 \\ -1 & 1 \\ -1 & -1 \end{bmatrix}. \quad (38)$$

We provide a comparison of the number of variables and constraints in the Table 4 produced by the two approaches as a function of  $n_\phi$ . From this analysis one can clearly see that under the assumptions that for systems with a large number of switching functions the Heaviside Step reformulation produces considerably fewer complementarity conditions.





**Figure 12:** Unions of sets is simplified by the Step reformulation, similar colored regions contain the same vector field.

The question remains of how to calculate  $\theta_i$  from the step values  $\alpha$ . The process for this is described in [54] and comes from the equivalent set arithmetic operations shown in Table 3:

$$\dot{x} \in F_F(x) := \left\{ \sum_{i=1}^{n_f} \theta_i f_i(x) \mid \theta_i = \prod_{j=1}^{n_\phi} \left( \frac{1 - S_{i,j}}{2} + S_{i,j} \alpha_j \right), \ i = 1, \dots, 2^{n_\phi}, \ \alpha_j \in \gamma(\phi_j(x)) \right\}. \quad (39)$$

Using these definitions we define an aggregated LP for calculating the  $\alpha_i$  associated with each region:

$$\alpha = \arg \min_{\alpha \in \mathbb{R}^{n_\phi}} -\phi(x)^\top \alpha \quad (40a)$$

$$\text{s.t.} \quad 0 \leq \alpha_i \leq 1, \quad i = 1, \dots, n_\phi, \quad (40b)$$

and by embedding its KKT conditions and using the same  $F(x)$  as in Stewart's

reformulation we get the DCS system:

$$\dot{x} = F(x, u)\theta, \quad (41a)$$

$$\theta_i = \prod_{j=1}^{n_\phi} \left( \frac{1 - S_{i,j}}{2} + S_{i,j}\alpha_j \right), \quad i = 1, \dots, 2^{n_\phi}, \quad (41b)$$

$$\phi = \lambda^p - \lambda^n, \quad (41c)$$

$$0 \leq \lambda^n \perp \alpha \geq 0, \quad (41d)$$

$$0 \leq \lambda^p \perp e - \alpha \geq 0. \quad (41e)$$

As with Stewart's reformulation we can show (and it is proven in [54]) that the Lagrange multipliers of the LP  $\lambda^n$  and  $\lambda^p$  are continuous functions of time while  $\alpha$  may be discontinuous, for example in the case of a boundary crossing.

We can now treat our OCP in a different form as an OCP subject to a DCS, and in particular we introduce  $G_{LP}$  to collect the algebraic equations from the LP's KKT conditions. We also combine the the algebraic variables  $y = (\theta, \lambda, mu)$  in the case of the Stewart reformulation, and  $y = (\alpha, \lambda^p, \lambda^n)$  for the Heaviside Step reformulation.

Definition $R_i$	Expression $\theta_i$
$R_i = A$	$\theta_i = \alpha_1$
$R_i = A \cup B$	$\theta_i = \alpha_1 + \alpha_2$
$R_i = A \cap B$	$\theta_i = \alpha_1 \alpha_2$
$R_i = \text{int}(\mathbb{R}^{n_x} \setminus A) = \{x   c_1(x) < 0\}$	$\theta_i = 1 - \alpha_1$
$R_i = A \setminus B$	$\theta_i = \alpha_1 - \alpha_2$

**Table 3:** Expressions of  $\theta_i$  for different definitions of  $R_i$ .

**Table 4:** Comparisons of the problem sizes in Stewart's and the Heaviside step reformulation for a fixed  $n_\phi$ , with a dense S.

Method	Number of systems	$n_{\text{alg}}$	$n_{\text{comp}}$	$n_{\text{eq}}$
Stewart	$2^{n_\phi}$	$2 \cdot 2^{n_\phi} + 1$	$2^{n_\phi}$	$2^{n_\phi} + 1$
Heaviside step	$2^{n_\phi}$	$2^{n_\phi} + 3n_\phi$	$2n_\phi$	$n_\phi + n_f$

Using these we define:

$$\begin{aligned}
& \min_{x(\cdot), u(\cdot), y(\cdot)} \quad \int_0^T L(x, u) dt + E(x(T)) \\
& \text{s.t.} \quad x(0) = x_0, \\
& \quad \dot{x}(t) = F(x(t), u(t))\theta(t), \quad t \in [0, T], \\
& \quad 0 = G_{\text{LP}}(x(t), y(t)), \quad t \in [0, T], \\
& \quad 0 \geq h(x(t), u(t)), \quad t \in [0, T], \\
& \quad 0 \geq r(x(T)).
\end{aligned}$$

As a brief aside we mention that the Heaviside step reformulation can be used for systems that expand beyond those which can be represented by a Filippov system. These are called Aizerman–Pyatnitskii systems, and while we provide expert functionality within NOSNOC for user to implement these, we do not treat them further in this thesis [2, 55].

### Switch Detection

We now have a process for converting Filippov PSS to two kinds of DCS, but as described in Section 3.2, we still suffer from the numerical and theoretical difficulties that come with discretizing such a system. As such we aim to discretize this DCS in such a way that the solution to the resulting NLP exhibits switch detecting behavior, i.e. the integrator can identify switches and place nodes of the integration method at those times. This is crucial as one can clearly see that if the discrete integration nodes coincide with switches, each integration step, or as we call them “Finite Element”, in the discretization is not subject to the problems normally caused by the r.h.s. of the ODE being discretized being discontinuous. In this section we will stick to treating version of the DCS produced by Stewart’s reformulation, but the same

general principles apply to the Heaviside step reformulation and more details for that particular approach can be found in [54].

In order describe the process of FESD we will treat the optimal control formulation of as single control interval and focus on a discretization of the DCS provided in Equation (34a) over the time interval  $[0, T]$ . The three primary components of the FESD scheme:

1. Allowing integration interval length  $h_i$  of each finite element to be an free variable in the optimization,
2. Enforcing switch occurrence only on finite element boundaries,
3. Step equilibration to get rid of the spurious degrees of freedom when there are no switches to detect.

We begin by treating  $N_{\text{fe}}$  steps of variable length  $h_i$  which yields integration intervals  $[t_k, t_k + h_k]$  with  $t_0 = 0$ ,  $t_{k+1} = t_k + h_k$ , and  $t_{N_{\text{fe}}} = T$ . Each finite element is treated as an integration of a smooth system with an implicit Runge-Kutta integrator with  $n_s$  stage points. If we take a fixed grid of step with  $h = \frac{T}{N_{\text{fe}}}$  and  $a_{i,j}$ ,  $b_i$ , and  $c_i$  ( $i, j = 1, \dots, n_c$ ) being Butcher tableaux entries as described in [53] we get a simple

time-stepping discretization of the form:

$$x_{0,0} = x_0, \quad (42a)$$

$$h_n = \frac{T}{N_{\text{fe}}}, \quad (42b)$$

$$x_{n+1,0} = x_{n,0} + h_n \sum_{i=1}^{n_s} b_i v_{n,i}, \quad (42c)$$

$$x_{n,i} = x_{n,0} + h_n \sum_{j=1}^{n_s} a_{i,j} v_{n,j}, \quad (42d)$$

$$v_{n,i} = F(x_{n,i}, u_{n,i}) \theta_{n,i}, \quad (42e)$$

$$0 = g(x_{n,i}) - \lambda_{n,i} - e\mu_{n,i}, \quad (42f)$$

$$0 \leq \theta_{n,i} \perp \lambda_{n,i} \geq 0, \quad (42g)$$

$$1 = \sum_{i=1}^{n_f} \theta_{n,i}, \quad (42h)$$

with  $i = 1, \dots, n_s$  and  $n = 0, \dots, N_{\text{fe}} - 1$ . This discrete-time system will clearly exhibit the poor behavior discussed prior including introducing artificial local minima and low order accuracy. FESD instead treats  $h_n$  as a free variable and simply imposes that the sum of step sizes is equivalent to the length of the interval. However we also need to somehow enforce that all switches occur at the boundaries between finite elements. In order to do this, we introduce the concept of “cross-complementarity”. The intuition for these additional constraints is that in order to enforce the above condition an equivalent condition is that for any given finite element the active set as defined previously must stay constant. In order to do this we can take advantage of the continuity properties of  $\lambda$  as mentioned previously. To the standard complementarity constraints in Equation (42g) we add the “cross-complementarity” constraints:

$$0 = \text{diag}(\theta_{n,m}) \lambda_{n,m'}, \text{ for all } m = 1, \dots, n_s, \ m = 0, \dots, n_s, \ m \neq m', \quad (43)$$

and can derive the following lemma, the proof of which is in [53]:

**Lemma 3.3.1.** *Regard a fixed  $n = 0, \dots, N_{\text{fe}} - 1$  and a fixed  $i1, \dots, n_f$ . If any  $\theta_{n,m,i}$*

with  $m = 1, \dots, n_s$  is positive, then all  $\lambda_{n,m',i}$  with  $m' = 0, \dots, n_s$  must be zero. Conversely, if any  $\lambda_{n,m',i}$  is positive, then all  $\theta_{n,m,i}$  are zero.

These constraints and the above lemma can be used to show that any active set change outside of the boundary between finite elements is made infeasible. Recall also that in the KKT conditions of the Stewart LP we have that  $\mu_{n,n_s} = \min_j g_j(x_{n+1})$  and if the active set changes at this point we get that  $\phi_{i,j}(x_{n+1}) = g_i(x_{n+1}) - g_j(x_{n+1}) = 0$ , where  $\phi_{i,j}$  is the boundary between  $R_i$  and  $R_j$ , i.e its zero level set. Therefore we get that  $x_{n+1}$  must be exactly on that boundary and thus  $h_n$  is enforced to move this finite element to start exactly when a switch occurs. Using the above constraints we get an FESD discretized DCS of the form:

$$x_{0,0} = x_0, \tag{44a}$$

$$T = \sum_{n=0}^{N_{\text{fe}}-1} h_n, \tag{44b}$$

$$x_{n+1,0} = x_{n,0} + h_n \sum_{i=1}^{n_s} b_i v_{n,i}, \tag{44c}$$

$$x_{n,i} = x_{n,0} + h_n \sum_{j=1}^{n_s} a_{i,j} v_{n,j}, \tag{44d}$$

$$v_{n,i} = F(x_{n,i}, u_{n,i}) \theta_{n,i}, \tag{44e}$$

$$0 = g(x_{n,i'}) - \lambda_{n,i'} - e\mu_{n,i'}, \tag{44f}$$

$$0 \leq \theta_{n,i} \perp \lambda_{n,i'} \geq 0, \tag{44g}$$

$$1 = \sum_{i=1}^{n_f} \theta_{n,i}, \tag{44h}$$

with  $i = 1, \dots, n_s$ ,  $i' = 0, \dots, n_s$  and  $n = 0, \dots, N_{\text{fe}}-1$ . We also enforce the continuity of  $\lambda(t)$  we further introduce the constraint that  $\lambda_{n,0} = \lambda_{n-1,n_s}$ . In order to simplify

some of our notation we introduce the collected vectors:

$$\begin{aligned}\theta_n &= (\theta_{n,1}, \dots, \theta_{n,n_s}), \\ \lambda_n &= (\theta_{n,0}, \dots, \theta_{n,n_s}), \\ h &= (h_1, \dots, h_n).\end{aligned}$$

It is important to note that in adding  $h_n$  as degrees of freedom we feasibly allow the NLP solver to possibly use discretization error to reduce the objective of the OCP. As such the final part of the FESD algorithm is introduced  $\nu(\theta_n, \theta_{n+1}, \lambda_n, \lambda_{n+1})$ , the Nurkanović function, who's structure is described in [53] and is zero if a switch occurs between finite elements  $n$  and  $n + 1$  and is non-zero otherwise. As such we add the constraint:

$$0 = \nu(\theta_{n'}, \theta_{n'+1}, \lambda_{n'}, \lambda_{n'+1})(h_{n'+1} - h_{n'}), \quad n' = 0, \dots, N_{\text{fe}} - 2, \quad (45)$$

to Equation (44). With this the definition of the FESD discretization is primarily complete. There are further complications that occur due to Runge-Kutta methods for which  $c_{n_s} \neq 1$  which require additional continuity constraints but we direct the reader to [53] for a more detailed explanation.

### Cross-Complementarity Aggregation

The complementarity constraints described in Equation (44g) can be equivalently formulated in several ways by aggregating the  $\theta_{n,i}$  or  $\lambda_{n,j}$  values via summation. This is due to the non-negativity constraints on both  $\theta$  and  $\lambda$ . In [56] multiple extensions to this sparse formulation are proposed including the use of inner products, summation of the algebraics over each finite element or over all the finite elements. In this thesis we primarily focus on the idea of summing the algebraics within each finite element. This yields three additional aggregation modes along with the sparse version previously described. In order to generate these modes we introduce the following

values:

$$\sigma_n^\theta = \sum_{i=1}^{n_s} \theta_{n,i}, \quad (46a)$$

$$\sigma_n^\lambda = \sum_{i=0}^{n_s} \lambda_{n,i}. \quad (46b)$$

We then can redefine Equation (44g) in the following ways:

$$0 \leq \sigma_n^\lambda \perp \theta_{n,i} \geq 0, \quad n = 0, \dots, N_{\text{fe}} - 1, \quad i = 1, \dots, n_s, \quad (47a)$$

$$0 \leq \lambda_{n,i} \perp \sigma_n^\theta \geq 0, \quad n = 0, \dots, N_{\text{fe}} - 1, \quad i = 0, \dots, n_s, \quad (47b)$$

$$0 \leq \sigma_n^\lambda \perp \sigma_n^\theta \geq 0, \quad n = 0, \dots, N_{\text{fe}} - 1. \quad (47c)$$

The above three modes are labeled 3, 4, and 7 respectively, while the original sparse mode is called mode 1. We introduce some theory in a later section for these four modes in a generic setting.

### Step-Equilibration Modes and Heuristics

As described in [53, 54, 56], the function  $\nu(\theta_n, \theta_{n+1}, \lambda_n, \lambda_{n+1})$  is generally unpleasant as it is both highly nonlinear and exhibits large variation in its magnitude which can cause numerical difficulties. As such there are several alternative reformulations of these so called step equilibration constraints that can achieve the same goal of preventing the optimizer from taking advantage of step size being a degree of freedom.

The first two of these approaches attempt to tackle both of the above issues at once. If we simply want to discourage the optimizer from deviating from an equidistant grid except when necessary due to the switch detection constraints, we can introduce a heuristic penalty from diverging from the mean  $\bar{h} = \frac{T}{N_{\text{fe}}}$  for this situation and



augmenting the objective of the discrete-time OCP with:

$$F_{\bar{h}}(h) = \rho_h \sum_{n=0}^{N_{\text{fe}}-1} (h_n - \bar{h})^2. \quad (48)$$

Alternatively we can more closely approximate the behavior of the original step equilibration constraint, which keeps the step sizes of adjacent finite elements constant when there are no switches. To accomplish this we can augment the objective with:

$$F_{\Delta h}(h) = \rho_h \sum_{n=0}^{N_{\text{fe}}-2} (h_n - h_{n+1})^2. \quad (49)$$

Both of these heuristic approaches replace  $\nu(\cdot)$  with more well behaved formulations, but come with a major caveat. In particular, one must be careful to choose an appropriate  $\rho_h$  in order to both prevent the optimizer from driving the value of  $h_n$  to zero but also scaled in such a way as to limit biasing the solver towards controls that lead to switches at the equidistant points.

An alternative approach to dealing with this is to introduce an  $\ell_2$ -penalty formulation to replace the constraint in Equation (45). This is a common approach for dealing with pathologically nonlinear constraints when solving general NLPs and often improves performance of methods on degenerate problems. This can be written as

$$F_{\ell_2}(h) = \rho_h \sum_{n=0}^{N_{\text{fe}}-2} \nu(\theta_n, \theta_{n+1}, \lambda_n, \lambda_{n+1}) (h_n - h_{n+1})^2, \quad (50)$$

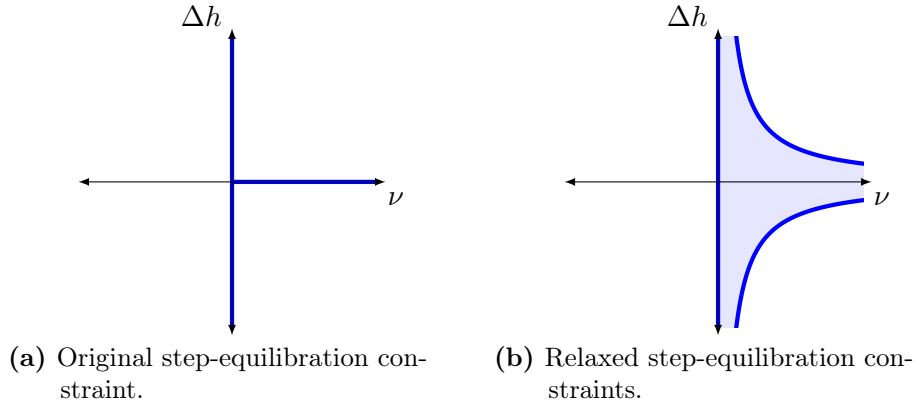
and simplifies greatly the feasibility issues that step-equilibration can cause. It however does not solve the pathological nature of the  $\nu$ -function and this can cause scaling issues which can wreak havoc on the linear algebra required by NLP solvers. We can however re-scale the indicator function  $\nu$  as we are only interested in whether this function is zero to allow  $h$  to vary. A simple way to do this is simply to take the hyperbolic tangent of  $\nu$  which rescales the value to between 0 and 1. This leads to an

“scaled- $\ell_2$ -relaxation” approach which is written as

$$F_{\tanh(\ell_2)}(h) = \rho_h \sum_{n=0}^{N_{\text{fe}}-2} \tanh(\nu(\theta_n, \theta_{n+1}, \lambda_n, \lambda_{n+1}))(h_n - h_{n+1})^2, \quad (51)$$

and trades extra non-linearity in the objective for better scaling.

There is also a final reformulation that takes advantage of the geometric structure of the step-equilibration constraint and its similarity to complementarity. In particular we show in Figure 13 that we can relax the constraint via the generation of 2 complementarity constraints. This unfortunately violates several of the constraint qualifications we will discuss in the next chapter, however in practice this can sometimes still improve performance in combination with the relaxation homotopy we also describe in Chapter 2.



**Figure 13:** A plot of a single step-equilibration constraint and its homotopy relaxation.

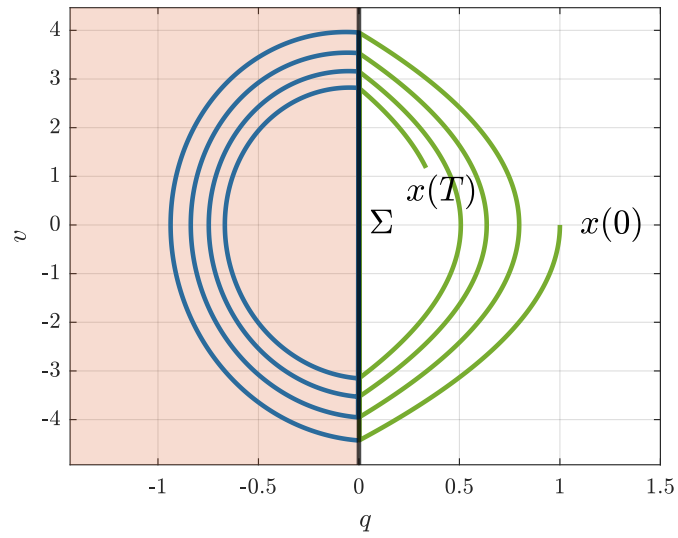
### 3.3.2 The Time-Freezing Reformulation

The prior section has described a discretization that allows us to accurately handle NSD1 and NSD2 systems, however we have heretofore not described any methods for dealing with NSD3 systems, i.e., those that contain state jumps. However, it would be a shame to not be able to re-use the machinery of the previous section to

handle all kinds of non-smoothness. To this end Nurkanović et al. introduce in [57] a reformulation that allows for the transformation of NSD3 systems into NSD2 systems, which can then be treated exactly as described above. In this section we, for brevity, do not describe the entirety of the approach but will go over an example for a simple one dimensional particle that makes an elastic impact with the ground, and omit the reformulation machinery required to handle, for example, friction, and inelastic impacts.

The Time-Freezing reformulation comes from two primary observations of dynamic systems with state jumps. The first is the fact that one can introduce time as a state variable (which we call “physical time”) which is often done when formulation time optimal control problems. We then treat numerical time (which in the time-freezing reformulation is called  $\tau$ ) and physical time as independent quantities, and ensure fixed time steps by introducing a speed-of-time variable  $s$  and a terminal constraint  $t(T) = T$  where  $T$  is the terminal time of a step in the discretization. The second is that in the cases of dynamic systems with impacts, there are regions of the state space that are not used as they are not valid configurations of the system. The Time-Freezing reformulation introduces new dynamics in that unused state space which do not advance physical time, but implement the state jump law for that contact. We show a schematic representation of this in phase plot form in Figure 14.

We use a one dimensional particle with state  $(q, v) \in \mathbb{R}^2$  and the dynamics  $(\dot{q}, \dot{v}) = (v, -g)$  with  $g \in \mathbb{R}$ , and it makes contact with the ground at  $x = 0$ . For the contact we define the contact function  $f_c(x) = x$ . The first step of the Time-Freezing reformulation is then to augment the state with physical time  $t$  so the state space becomes  $x(\tau) = (q(\tau), v(\tau), t(\tau))$ . The forbidden region in this case is the region  $q < 0$  which means we define two regions  $R_1 = \{x | q > 0\}$  and  $R_2 = \{x | q < 0\}$  with



**Figure 14:** Phase plot of an elastic time freezing system.

dynamics

$$f_1 = s \begin{bmatrix} v \\ -g \\ 1 \end{bmatrix},$$

$$f_2 = s \begin{bmatrix} v \\ -kq \\ 1 \end{bmatrix},$$

for an arbitrary  $k > 0$ . This becomes an NSD2 system that we can now discretize with FESD.

For brevity we point to a more in-depth explanations in [57, 58]. We also note that in [56] an alternative direct reformulation for such systems is introduced which can handle different elasticities and friction characteristics on a per-impact basis, which is called FESD-J.

### 3.4 Theoretical Properties of FESD

In this section we will briefly treat the theoretical properties of the MPCCs produced by the FESD discretization. In particular we treat a generic MPCC with cross complementarity conditions that shares its structure with a single FESD finite element:

$$\underset{x, \lambda, \theta}{\text{minimize}} \quad F(x, \lambda, \theta) \quad (52a)$$

$$\text{subject to} \quad 0 = g(x, \lambda, \theta), \quad (52b)$$

$$0 \leq h(x, \lambda, \theta), \quad (52c)$$

$$0 \leq G(\lambda) \perp H(\theta) \geq 0, \quad (52d)$$

where  $g, h$  are arbitrary equality and inequality constraints and  $G$  and  $H$  encode the cross complementarity constraints. For ease of notation we treat:

$$\lambda = \lambda_{i,j}, \quad i = 0, \dots, n_s, \quad j = 1, \dots, n_f, \quad (53a)$$

$$\theta = \theta_{i,j}, \quad i = 1, \dots, n_s, \quad j = 1, \dots, n_f, \quad (53b)$$

$$\lambda_i = \lambda_{i,j}, \quad j = 1, \dots, n_f, \quad (53c)$$

$$\theta_i = \theta_{i,j}, \quad j = 1, \dots, n_f. \quad (53d)$$

We also introduce the notation

$${}^k v = \left. \begin{bmatrix} v \\ v \\ \vdots \\ v \end{bmatrix} \right\} k \text{ times}$$

where  $v$  is a vector and  $k$  is a positive integer.

### 3.4.1 Sparse Cross-Complementarity

We then define the cross-complementarity functions:

$$G(\lambda) = \begin{bmatrix} n_s \lambda_0 \\ n_s \lambda_1 \\ \vdots \\ n_s \lambda_{n_s} \end{bmatrix}, \quad (54a)$$

$$H(\theta) = {}^{(n_s+1)}\theta, \quad (54b)$$

which match the cross complementarities given for a single finite element in their sparsest form in Section 3.3.1.

We now analyze the constraint qualification properties of this form under the assumptions that  $n_f \geq 2$  and  $n_c \geq 2$  and propose that:

**Theorem 3.4.1.** *For any problem of the form shown in Equation (52) with complementarity functions  $G(\lambda)$  and  $H(\theta)$ , described in Equation (54), that arises from an FESD discretization, any feasible point  $\bar{x}$ ,  $\bar{\lambda}$ ,  $\bar{\theta}$  violates MPCC-LICQ and MPCC-MFCQ.*

*Proof.* In order to prove this we must recall that there are three sets of indices that are critical to evaluating the constraint qualifications of a given point:  $\mathcal{I}_{00}$ ,  $\mathcal{I}_{0+}$ ,  $\mathcal{I}_{+0}$ . We first treat the case where we have no bi-active constraints, i.e.,  $\mathcal{I}_{00} = \emptyset$ . In this case,  $\mathcal{I}_{0+}$  and  $\mathcal{I}_{+0}$  partition the set  $\mathcal{I} := \{1, \dots, n_s n_f (n_s + 1)\}$ . As such in order to prove this theorem for this case we need to show that there is indeed no way to partition the corresponding columns of  $\nabla G$  and  $\nabla H$  that leads to a linearly independent set of vectors. We note that these sets of vectors  $\mathbf{G} := \{ \nabla G_i(\lambda) | i \in \mathcal{I}_{0+} \}$  and  $\mathbf{H} := \{ \nabla H_i(\theta) | i \in \mathcal{I}_{+0} \}$  are mutually independent and as such our partition is only concerned with the linear independence of the vectors within each set.

In order to show this we use a trick to reformulate this into a set theoretic problem. One can clearly see that we can map each  $i \in \mathcal{I}$  to a set of 3-tuples

$$S := \{ (a, b, c) | a \in \{0, \dots, n_s\}, b \in \{1, \dots, n_s\}, c \in \{1, \dots, n_f\} \}.$$

The problem of partitioning  $\mathcal{I}$  into  $\mathcal{I}_{0+}$  and  $\mathcal{I}_{+0}$  that produces a set of independent vectors  $\mathbf{G}$  and  $\mathbf{H}$ , can thus be recast into the problem of partitioning the set  $S$  into  $S_G$  and  $S_H$  such that for both of those sets the following condition holds:

$$\forall (a, b, c), (a', b', c') \in S_{\bullet}. ((a = a') \wedge (c = c') \Rightarrow (b = b')) \wedge ((b = b') \wedge (c = c') \Rightarrow (a = a')). \quad (55)$$

The intuitive way to interpret this condition is that each set cannot contain two distinct elements  $(a, b, c)$  and  $(a', b', c')$  that share their first and third, or second and third entries. To prove that this is indeed not possible we apply the well known “pigeon-hole” principle for  $n_s > 1$ . This is done by seeing that for any  $n_s > 1$  there are  $n_s + 1$  members of the set  $S$  of the form  $(a, 1, 1)$  with  $a \in \{0, \dots, n_s\}$ , which must be partitioned into only two sets. By the “pigeon-hole” principle there must be at least 2 tuples of this form in at least one of the sets as  $n_s + 1 > 2$ .

This means that for any partition of  $\mathcal{I}$  into  $\mathcal{I}_{0+}$  and  $\mathcal{I}_{+0}$  the independence conditions on the complementarity functions required for MPCC-LICQ and MPCC-MFCQ are violated at all feasible points with strict complementarity. To extend this to include points without strict complementarity we simply observe that the addition of a 3rd set  $i \in \mathcal{I}_{00}$  for which both vectors  $\nabla G_i$  and  $\nabla H_i$  must be checked for independence strictly adds more vectors to the set that must be linearly independent. This relaxes the partitioning constraint for  $S_G$  and  $S_H$ , to one where  $S = S_G \cup S_H$ . However in this situation we can apply the same argument to prove that a pair of sets  $(S_G, S_H)$  that satisfies Equation (55) does not exist. As such we have proven that in its sparsest form, the FESD discretization produces MPCCs that violate MPCC-LICQ and MPCC-MFCQ.  $\square$

### 3.4.2 Dense Cross-Complementarity

The dense version of the cross-complementarity constraints is both the most intuitive and also possesses some interesting theoretical properties. It can be shown that in the cases of problems with only “crossing” style switches (such as those seen in Figure 11c), that any solution  $w^*, \lambda^*, \theta^*$ , will exhibit strict complementarity, and therefore that point is guaranteed to be S-stationary. We also can prove that the dense cross-complementarity mode does not have the same LICQ violation problems. This can be done by directly calculating the gradients of  $G_j(\lambda) = \sum_{i=0}^{n_s} \lambda_{i,j}$  and  $H_j(\theta) = \sum_{i=1}^{n_s} \theta_{i,j}$ . It can clearly be seen that these gradients are made up of  $n_f$  stacked identity matrices which yields  $n_f$  linearly independent vectors for both  $\nabla G(\lambda)$  and  $\nabla H(\theta)$ .

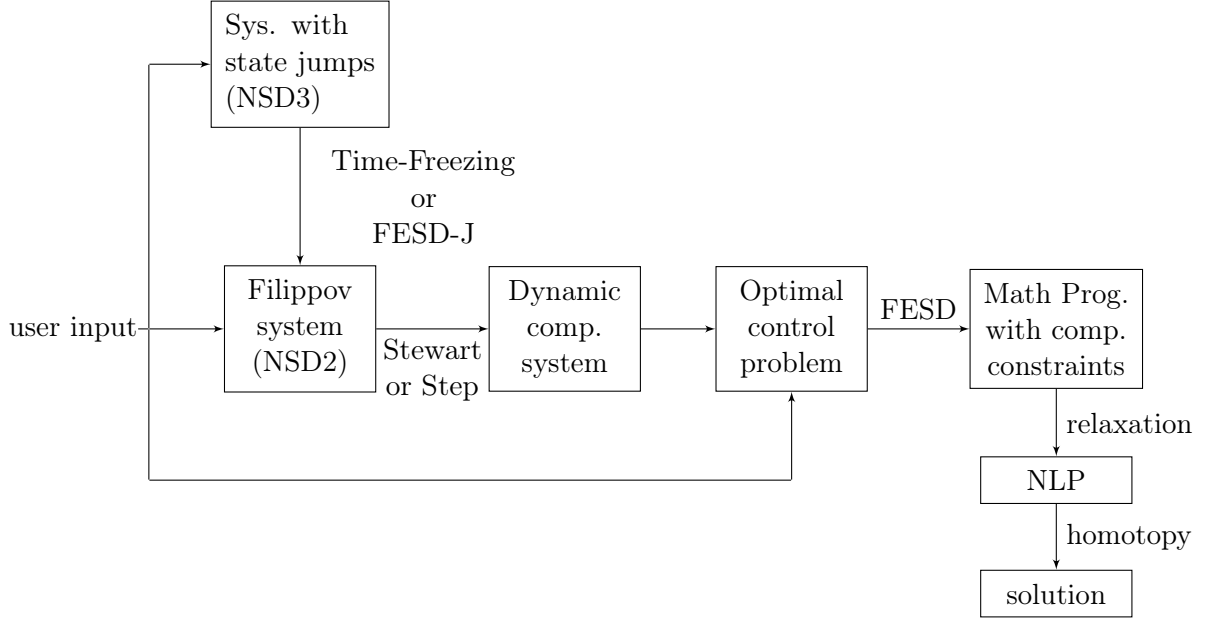
## 3.5 NOSNOC

The previously described approaches are implemented in the open-source tool NOSNOC [9], which is available both as a MATLAB and python package. This package uses CasADi [59], a modeling and Automatic Differentiation tool, to provide a user friendly method of defining the nonsmooth continuous time system. The continuous time OCP or simulation problem is then reformulated into MPCCs and these are solved via a variety of methods and the solutions are provided back to the user. We also include some functionality to automatically reformulate ODEs with nonsmooth operators, such as “min”, “max”, and “sign”.

### 3.5.1 Tool-chain

An overview of the NOSNOC tool-chain is shown in Figure 15 and describes the transformations applied at each step of the process. We will in this section briefly





**Figure 15:** The NOSNOC Tool-chain

describe the available features in the NOSNOC package and the interfaces it provides.

### NSD2 System Tool-chain

When using NOSNOC in conjunction with a system of type NSD2, the problem formulation that the tool supports are a simulation problem or an OCP with dynamics described by a Filippov system. As such we either simulate a system:

$$\dot{x} \in \left\{ \sum_{i \in \mathcal{I}} \theta_i f_i(x, u) \mid \sum_{i \in \mathcal{I}} \theta_i = 1, \theta_i \geq 0, \theta_i = 0 \text{ for } x \notin \bar{R}_i \right\}, \quad (56)$$

or, in the case of an OCP, we solve a problem of the form:

$$\min_{x(\cdot), u(\cdot), z(\cdot), v} \int_0^T L(x, u, z, v) dt + E(x(T), z(T), v) \quad (57a)$$

s.t.

$$x(0) = x_0, \quad (57b)$$

$$\dot{x}(t) \in F_F(x(t), u(t), z(t), v), \quad t \in [0, T], \quad (57c)$$

$$\underline{x} \leq x(t) \leq \bar{x}, \quad t \in [0, T], \quad (57d)$$

$$\underline{z} \leq z(t) \leq \bar{z}, \quad t \in [0, T], \quad (57e)$$

$$0 = g_z(x(t), u(t), z(t), v), \quad t \in [0, T], \quad (57f)$$

$$\underline{g} \leq g(x(t), u(t), z(t), v) \leq \bar{g}, \quad t \in [0, T], \quad (57g)$$

$$0 \leq G_{\text{user}}(x(t), u(t), z(t), v) \perp H_{\text{user}}(x(t), u(t), z(t), v) \geq 0, \quad t \in [0, T], \quad (57h)$$

$$\underline{g}_T \leq g_T(x(T), z(T), v) \leq \bar{g}_T, \quad (57i)$$

$$\underline{x}_T \leq x(T) \leq \bar{x}_T, \quad (57j)$$

which is a generic OCP and allows for encoding a large number of different behaviors and constraints. While most of the above components are self-explanatory, the remaining difficulty is how to allow the user to pass in the Filippov system  $F_F$ . To this end we use the previously introduced approaches that are taken advantage of in Section 3.3.1. As such the interface provides the options to provide the definition of the Filippov system via providing  $F(x, u, z, v) = [f_1(\cdot), f_2(\cdot), \dots]$  which are the dynamics in  $R_i$  and one of two ways to define that region. This is done by either providing  $g_i^{\text{Stewart}}(x)$ , the Stewart indicator functions for region  $R_i$ , or by providing  $\phi(x)$  and the sign matrix  $S$  as described in Section 3.3.1.

In order to discretize the system described in Equation (57c), NOSNOC first uses one of the two reformulations described previously to convert the Filippov system into a DCS which can be discretized via FESD. If this  $S$  matrix is dense then the user may choose to use either the Stewart or Heaviside Step reformulation, but only the latter

is available if the  $S$  matrix is sparse. This DCS now includes the algebraic variables  $\theta, \lambda, \mu$  for  $\alpha, \lambda^n, \lambda^p$ , in the case of the Stewart and Heaviside Step reformulation respectively, but is still essentially in continuous time.

At this point we discretize the DCS and OCP with the discretization parameters  $N_s$ ,  $N_{fe}$ , and  $n_s$ , and the user selected Runge-Kutta method. These represent the number of control stages with piecewise constant controls, the number of finite elements, and the number of stages used by the IRK scheme, respectively. We further provide the functionality to the user to choose at what frequency to enforce the constraints defined by Equation (57g), with the three options: at each IRK stage point, at the boundary of each Finite Element, and only at the boundaries of each control interval. The same is provided as an option for the user defined complementarity conditions  $0 \leq G_{\text{user}} \perp H_{\text{user}} \geq 0$ . Further FESD options are provided including one of the four cross-complementarity aggregation modes described in the earlier section on FESD, along with all of the described heuristic and relaxation approaches to implementing step-equilibration.

After this step we now have an MPCC which represents the discretization of the either the OCP or simulation problem at hand. This MPCC is then currently solved via one of many relaxation methods which we describe in Section 2.2. The default NLP solver used to solve the relaxed NLPs is IPOPT [60], however we also provide plugin functionality for SNOPT [61], WORHP [62], and UNO [63].

### NSD3 Systems

NOSNOC further provides functionality to automatically reformulate NSD3 systems and solve problems where Equation (57c) is replaced with a Complementarity Lagrangian system modeling rigid bodies with frictional impacts. When defining such a system the user provides the state space in terms of generalized positions  $q$  and generalized velocities  $v$ , yielding  $x = (q, v)$ . In particular it allows the user to

define the signed distance functions  $f_{\phi}^i(x) : \mathbb{R}^{n_x} \rightarrow \mathbb{R}^{n_{\text{contacts}}}$  which define the contact dynamics of the system. Each contact is also permitted to have distinct coefficients of restitution  $e_i$  and coefficients of friction  $\mu_i$ , but only in the case of the FESD-J reformulation. Otherwise, those coefficients must be the same for all impacts. This system's free-flight (i.e. when there are no active contacts) dynamics are defined by the time derivative of the generalized velocities  $\dot{v} = f_v(x, u)$ , and an inertia matrix  $M(q)$ .

This system then is handled in one of two ways: via the Time-Freezing reformulation previously described or via the FESD-J reformulation [56] that directly creates a complementarity-Lagrangian system from the above data. We note a limitation of the Time-Freezing reformulation here: in particular that it requires that it is impossible to mix inelastic and elastic contacts in the same problem. As such for such systems the FESD-J reformulation is the only approach that is viable.

#### 3.5.2 Software Structure

We briefly summarize the current software architecture of the package in this section. The user input for the OCP or simulation problem to be solved is split into two classes: `NosnocModel`, which contains the continuous time forms of the Filippov system or Lagrangian dynamics with contacts that are described in the previous section, and `NosnocProblemOptions` which contains the options used by NOSNOC to discretize the system. The latter also contains all of the options involved in the FESD discretization, and the simulation discretization parameters when NOSNOC is being used as an integrator. A third input structure, `NosnocSolverOptions`, is used to house the settings that relate to the regularization homotopy solver that is used to solve the MPCCs that are the output of the FESD pipeline.

If using NOSNOC purely for integration one can simply use the three input structure and the `NosnocIntegrator` class in order repeatedly solve the MPCC and produced

a simulated trajectory of the system based on initial conditions. On the other hand if one wants to solve an OCP the program flow is slightly different. In that case the user must create the MPCC (an `NosnocMPCC` object) from the `NosnocModel` and `NosnocProblemOptions`. This MPCC can then be passed to the `NosnocSolver` object along with the `NosnocSolverOptions`, and then one can call `NosnocSolver.solve()` to acquire the solution to the OCP. Internally `NosnocSolver` does the conversion from MPCC to relaxed NLP that we describe in the Chapter 2, and uses the `NosnocNLP` class to encode this NLP and maintain the index sets for all relevant variables.



## 4 NOSBENCH - An MPCC Benchmark

The primary contribution of this thesis is the introduction of a benchmark suite of MPCCs that come from the discretizing OCPs and Simulation problems with the Finite Elements with Switch Detection method discussed before. We first briefly discuss the formatting of the benchmark as well as various options for extracting the problems from the provided files. We then describe the problem set in detail.

### 4.1 Problem Format

Chapter 1 introduced the classical form of an MPCC in Equation (1). This however is not the form that we provide the problems in due to the influence of the CasADi interface as well as the interface of other NLP solvers. We provide parametric MPCCs in the following form:

$$\min_{w \in \mathbb{R}^{n_w}} F(w, p) \tag{58a}$$

$$\text{s.t.} \quad \ell_w \leq w \leq u_w, \tag{58b}$$

$$\ell_g \leq g(w, p) \leq u_g, \tag{58c}$$

$$0 \leq G(w, p) \perp H(w, p) \geq 0, \tag{58d}$$

where  $p \in \mathbb{R}^{n_p}$  is a parameter vector. This form permits complementarity constraints between arbitrary functions of the optimization variables  $G(w, p)$  and  $H(w, p)$ . Equality constraints are provided by setting  $\ell_{gi} = u_{gi}$ . We provide each problem in one of

two forms, the first of which is a NOSNOC specific pair of a model and the problem options, as described in Section 3.5, which can be used to generate the MPCC on the fly with the NOSNOC package. The alternative more generic format that is likely to be more useful to solver developers, is the MPCC in Equation (58) using a JSON object. In this format, the problem contains the variables  $w$  and parameters  $p$  as CasADi variables, the bounds  $\ell_w, u_w, \ell_g, u_g$  as numerical vectors, and the functions  $G, F, H, g$ . All CasADi functions and variables are converted to strings using CasADi's serialization functionality. Also provided is the unagumented objective function which can be used to evaluate the OCP performance without the heuristic step equilibration methods previously described. From this, a user can simply reconstruct the problem by loading in all of the components and using the provided CasADi deserialization functionality to interface their solver with NOSBENCH problems. In future, we expect to expand the availability of these problems to include a library of problems in the AMPL format.

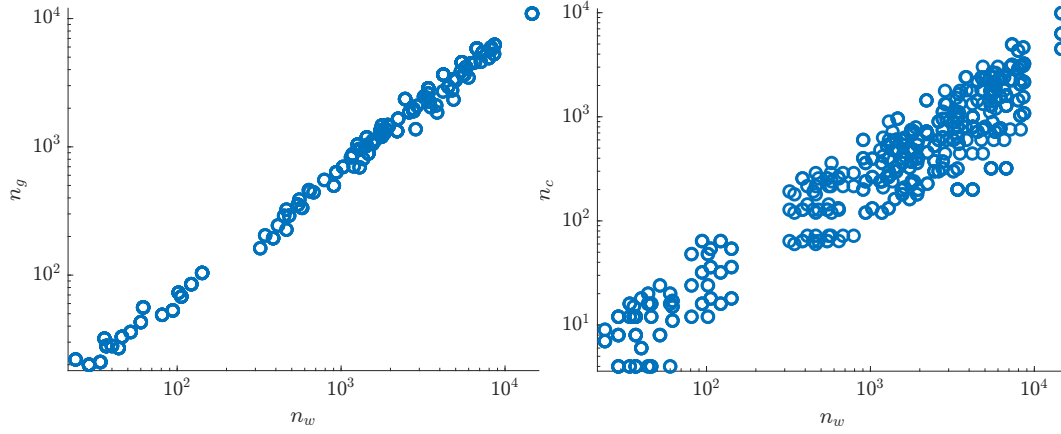
## 4.2 Problem Set

NOSBENCH contains a total of 603 distinct MPCCs within the full problem set. We provide in Figure 16 a scatter plot of the number of primal variables versus the dimension of  $g(w)$  and versus the number of complementarity pairs. We do not go into a detailed description of each problem as many are alternate discretizations of each other. In this section, we first discuss the continuous time systems that these problems are derived from, then the discretization options.

### 4.2.1 Original ODEs and OCPs

NOSBENCH problems are generated through FESD discretization of systems governed by nonsmooth ODEs. Table 5 presents the origins of each of these problems in





(a) Number of inequality constraints versus number of variables. (b) Number of complementarity constraints versus number of variables.

**Figure 16:** Size characteristics of the NOSBENCH Test-set.

continuous time along with references to them in the literature. This set includes both simulation and optimal control problems as both can and do produce interesting and difficult MPCCs and are an important use case of FESD. We briefly describe each of the systems along with some statistics about each of them in a later section.

**Table 5:** Continuous time problems used to generate the NOSBENCH test set

Nr.	Problem Slug	Short Description	Type	Class	Citation
1	3CPCLS	Two dimensional representation of three carts with only one actuated.	OCP	CLS direct	[21]
2	3CPTF	One dimensional representation of three carts with only one actuated.	OCP	CLS time-freezing	[21]
3	CARHYS	Turbo car example with hysteretic behavior.	OCP	HA time-freezing	[48]
4	CARTIM	Turbo car example with velocity dependent switch.	OCP	Filippov System	[53]
5	CPWF	Inverted pendulum on a cart with Coulomb friction	OCP	Filippov System	[64]
6	DAOBCLS	Disc control example from FESD-J paper.	OCP	CLS direct	[56]
7	DISCM	Disc control example from FESD-J paper.	OCP	CLS time-freezing	[56]

**Table 5:** Continuous time problems used to generate the NOSBENCH test set

Nr.	Problem Slug	Short Description	Type	Class	Citation
8	DRNLND	Two dimensional control of a landing drone.	OCP	CLS time-freezing	[65]
9	DSCOB	Disc control example from FESD-J paper with a central obstacle, modeled as a nonlinear constraint.	OCP	CLS time-freezing	[56]
10	DSCSP	Disc control example from FESD-J paper with discs switching positions.	OCP	CLS time-freezing	[56]
11	HOPOCP	Hopping robot actuated with a linear leg and reaction wheel in two Dimensions.	OCP	CLS time-freezing	[64]
12	MFTOPT	Time optimal control of a linear voice-coil motor.	OCP	Filippov System	[66]
13	MNPED	Two dimensional control of a monoped robot.	OCP	CLS time-freezing	[67]
14	MWFOCP	Optimal control of a linear voice-coil motor.	OCP	Filippov System	[66]
15	SCHUMI	Simple two dimensional car model with Cartesian track constraints. Both time optimal and non-time optimal forms.	OCP	Filippov System	[51]
16	SMOCP	Simple two dimensional optimal control with sliding modes.	OCP	Filippov System	[53]
17	TFBIB	Two dimensional control of a particle in a box with a rotating reference.	OCP	CLS time-freezing	
18	TNKCSC	Cascade of tanks with state dependent switches in flow rate.	OCP	Filippov System	[50]
19	TIMF1D	One dimensional particle with contacts.	Simulation	CLS time-freezing	[56]
20	2BCLS	Two balls connected with a stiff spring, that undergo contact dynamics.	Simulation	CLS direct	[68]
21	986EQ	Simplified model of structural pounding used in the study of the effects of earthquakes.	Simulation	Filippov System	[69]

**Table 5:** Continuous time problems used to generate the NOSBENCH test set

Nr.	Problem Slug	Short Description	Type	Class	Citation
22	986FO	Two masses linked by a spring and moving on a surface with Coulomb friction.	Simulation	Filippov System	[69]
23	986FV	Oscillator with varying friction.	Simulation	Filippov System	[69]
24	986OM	Oscillating mass with contact surfaces.	Simulation	Filippov System	[69]
25	CLS1D	One dimensional particle with contacts.	Simulation	CLS direct	[56]
26	FBS1S	Simulation of connected blocks with Coulomb friction.	Simulation	Filippov System	[70]
27	OSCIL	Simulation of an unstable oscillator with state dependent switch.	Simulation	Filippov System	[53]
28	RFB1S	Relay feedback system simulation.	Simulation	Filippov System	[71]
29	SMCRS	Simple Filippov system with a boundary crossing.	Simulation	Filippov System	[53]
30	SMLSM	Simple Filippov system with state entering a sliding mode.	Simulation	Filippov System	[53]
31	SMSLM	Simple Filippov system with state leaving a sliding mode.	Simulation	Filippov System	[53]
32	SMSPS	Simple Filippov system with state spontaneously leaving an unstable sliding mode.	Simulation	Filippov System	[53]
33	TFPOB	Two dimensional simulation of a pile of balls with contacts.	Simulation	CLS time-freezing	[53, 56]

### 4.2.2 Discretization Options

In order to generate problems of varying complexity and internal structure, we vary several discretization and MPCC parameters. The simplest of these are the actual discretization parameters of the FESD scheme:

- $N_s$ : The number of control intervals in the case of optimal control problems or the number of time steps solved as a single MPCC in the case of simulation problems. This is always one in the case of simulation problems.

- $N_{\text{fe}}$ : The number of finite elements per control stage.
- **irk\_scheme**: One of several available discretization schemes (defined by their Butcher tableau). In order to limit the number of MPCCs generated, and due to their accuracy, we primarily use the Radau-IIA or Gauss-Legendre schemes.
- $n_s$ : The number of stage points used by the selected implicit Runge-Kutta scheme within each finite element in the FESD discretization.

We also vary the so called “cross-complementarity mode” for various problems. This is primarily done to facilitate the experiments regarding the performance of these various modes in the experiments described in Chapter 5. These modes as described in Section 3.3 provide different sparsity to the complementarity constraints that enforce switch identification. And finally we vary the level of lifting done to the complementarity functions  $G(x)$ ,  $H(x)$  in the problem.

The set full set of problems is described in Table 6, where each column contains the values used for the corresponding parameter. Any values in curly braces are taken as a set and we provide a Cartesian product of all the sets for a given problem. Every problem is also provided in a lifted form where each  $G_i(x)$  and  $H_i(x)$  are a single variable.

**Table 6:** Problem discretizations for problems which make up NOSBENCH

Slug	irk_scheme	$N_s$	$N_{\text{fe}}$	$n_s$	Cross-Comp. Mode	Param. Var.	Total
2BCLS	Gauss-Legendre	1	2	3	{3, 4, 7}	3	9
3CPCLS	Radau-IIA	{15, 30}	3	2	{3, 4, 7}	3	18
3CPTF	Radau-IIA	{20, 30}	3	1	{3, 4, 7}	3	18
986EQ	Gauss-Legendre	1	3	2	{3, 4, 7}	2	12
986FO	Radau-IIA	1	2	3	{3, 4, 7}	3	18
986FV	Gauss-Legendre	1	2	2	{3, 4, 7}	3	18
986OM	Radau-IIA	1	2	3	{3, 4, 7}	2	12
CARHYS	Radau-IIA	{20, 31}	3	2	{3, 4, 7}	2	12
CARTIM	Radau-IIA	{10, 12, 35, 40}	3	2	{3, 4, 7}	1	24
CLS1D	Gauss-Legendre	1	{2, 3}	1	{3, 4, 7}	2	12

**Table 6:** Problem discretizations for problems which make up NOSBENCH

Slug	irk_scheme	$N_s$	$N_{fe}$	$n_s$	Cross-Comp. Mode	Param. Var.	Total
CPWF	Radau-IIA	{30, 33, 50, 57}	2	2	{3, 4, 7}	2	48
DAOBCLS	Radau-IIA	{23, 25, 31}	3	2	{3, 4, 7}	3	9
DISCM	Radau-IIA	{11, 25, 33}	3	1	{3, 4, 7}	1	9
DRNLND	Radau-IIA	{23, 30, 37}	3	2	{3, 4, 7}	2	18
DSCOB	Radau-IIA	{22, 25, 31}	3	1	{3, 4, 7}	3	27
DSCSP	Radau-IIA	{20, 23, 31}	3	1	{3, 4, 7}	1	9
FBS1S	Radau-IIA	1	3	2	{3, 4, 7}	3	18
HOPOCP	Radau-IIA	{20, 23, 37, 40}	3	2	{3, 4, 7}	1	12
MFTOPT	Radau-IIA	{27, 30, 50, 53}	3	2	{3, 4, 7}	1	24
MNPED	Radau-IIA	{40, 43, 57, 60}	3	2	{3, 4, 7}	3	36
MWFOCP	Radau-IIA	{27, 30, 50, 53}	3	2	{3, 4, 7}	1	24
OSCIL	Radau-IIA	1	2	4	{3, 4, 7}	2	12
RFB1S	Radau-IIA	1	2	2	{3, 4, 7}	3	18
SCHUMI	Radau-IIA	{50, 80}	2	2	{3, 4, 7}	6	72
SMCRS	Radau-IIA	1	32	2	{3, 4, 7}	1	6
SMLSM	Radau-IIA	1	32	2	{3, 4, 7}	1	6
SMOCP	Radau-IIA	{30, 37, 50, 63}	3	3	{3, 4, 7}	2	48
SMSLM	Radau-IIA	1	32	2	{3, 4, 7}	1	6
SMSPS	Radau-IIA	1	32	2	{3, 4, 7}	1	6
TFBIB	Radau-IIA	{40, 43}	4	2	{3, 4, 7}	2	12
TFPOB	Radau-IIA	1	5	2	{3, 4, 7}	4	12
TIMF1D	Gauss-Legendre	1	{3, 4}	1	{3, 4, 7}	2	12
TNKCSC	Gauss-Legendre	{96, 100}	2	1	{3, 4, 7}	1	6

Table 9 lists the particular variations of initial parameters or problem formulations that we use for the problems. This is done to provide some variety in the benchmark problems and to mitigate the effects of pre-tuning which has been done on some of these examples in the corresponding references. It also allows for simulation problems to be tested both in cases where there are no switches and cases where switches must be detected.

### 4.2.3 Problem Naming Scheme

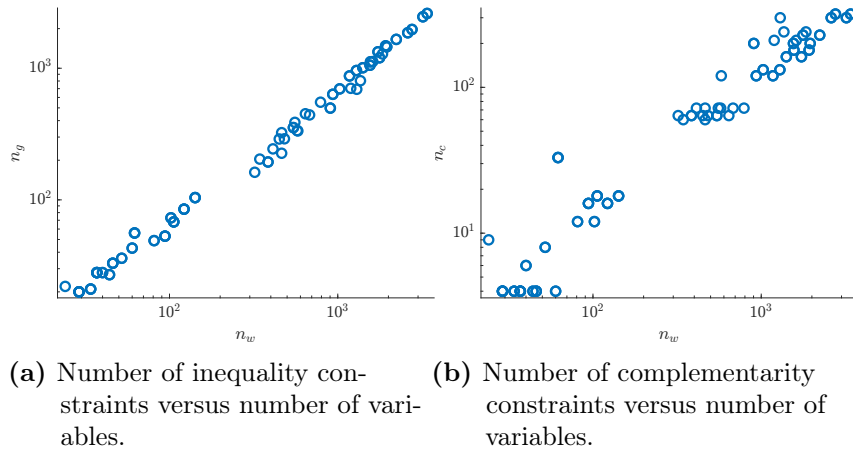
The names of the problem files encode some information about the problem. This is done so that the formation of sub-sets of the whole benchmark can be done systematically. The structure of these names is an underscore delimited string containing the following data in order: Problem slug, parameter index,  $N_s$ ,  $N_{fe}, n_s$ , IRK scheme, DCS mode (“Step”, “Stewart”, or “CLS”), cross complementarity mode, type, and whether the problem is lifted into vertical form. The type of the problem is split into “FIL”, Filippov systems, “IEC”, problems with only inelastic collisions, “ELC”, problems with some elastic (and possibly also inelastic) collisions, and “HYS” for problems containing hysteresis. For example the problem titled: 986EQ\_001\_001\_003\_2\_GL\_STEP\_7\_FIL\_1 would be the earthquake example from [69] with the first parameter set and  $N_s = 1$ ,  $N_{fe} = 3$ ,  $n_s = 2$ , using a Gauss-Legendre integrator. The problem is generated using the Step reformulation and cross complementarity mode 7, and is lifted into the vertical form.

### 4.2.4 Problem Subsets

As it is both often infeasible to run the full NOSBENCH suite, and doing so is not necessary to gain insight into the comparative performance of different solution methods. Therefore, we provide several smaller subsets of problems which can be used to benchmark any future solvers and are used in Chapter 5 to evaluate existing solver options.

#### Simple Problem Benchmark - NOSBENCH-S

The first subset of NOSBENCH is a benchmark that only uses 100 MPCCs which come exclusively from Filippov systems, and only contain the simplest time-freezing and CLS examples. These tend to produce relatively easier to solve MPCCs and as



**Figure 17:** Size characteristics of the NOSBENCH Simple Problem Benchmark.

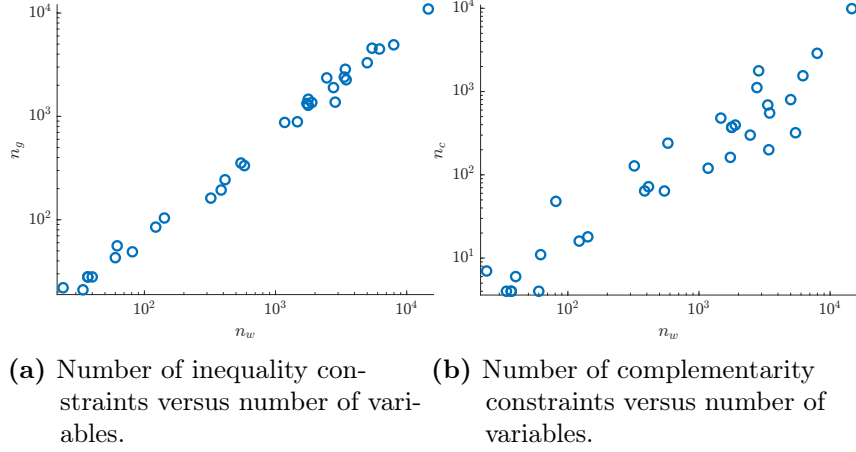
such this set is an effective way to identify particularly poor performing algorithms. It contains approximately equivalent numbers of simulation and optimal control problems and the vast majority of the problems can be solved with the existing state of the art in less than an hour. The size characteristics of the problems in this set can be seen in Figure 17.

#### Small Representative Benchmark - NOSBENCH-RS

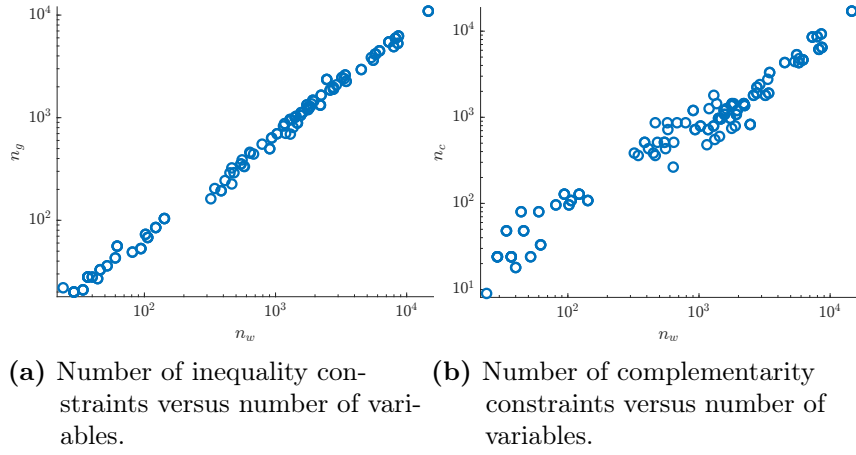
This subset of NOSBENCH is an even smaller but more representative benchmark. It contains 32 MPCCs that include ones from FESD-J and time-freezing reformulations. This subset is primarily used to as a second preliminary screen for solvers as it provides more insight into the performance of solvers on problems ranging from the easiest to the most difficult within NOSBENCH. The size characteristics of the problems in this set can be seen in Figure 18.

#### Large Representative Benchmark - NOSBENCH-RL

This subset is a set of 167 problems that is made up of a representative sample of all problem difficulties. This benchmark is meant to be the full problem set that is



**Figure 18:** Size characteristics of the NOSBENCH Small Representative Benchmark.



**Figure 19:** Size characteristics of the NOSBENCH Large Representative Benchmark.

used to benchmark solvers, and will continue to be expanded as new and interesting problems are added to NOSBENCH. The size characteristics of the problems in this set can be seen in Figure 19.



## 5 Numerical Experiments

In this chapter we discuss several experiments done to evaluate different solution methods for MPCCs. We First briefly evaluate the effectiveness of the parameter scaling introduced in Section 2.2.5 via comparing several of the relaxations used with and without the scaling. This is followed by an evaluation of the different relaxation options introduced in Section 2.2.1 and evaluate their performance quantitatively. We then explore the space of the homotopy parameters which are used to drive the complementarity relaxations toward the exact complementarity set.

This is followed by a comparison of three major NLP solvers (IPOPT, SNOPT, and WORHP) used to solve the regularized NLPs. These experiments are designed to extract some rules of thumb for the optimal default solver in NOSNOC.

We follow this with several experiments comparing the discretization options offered within NOSNOC. The first of these is a comparison between the performance of the two reformulations: Stewart and Heaviside step functions. This is used to evaluate the relative performance of the two reformulations. We follow this with a comparison of the cross complementarity modes to evaluate the importance of the complementarity sparsity or lack thereof. We finally briefly mention a comparison between lifting and direct methods (ones where the complementarities are treated directly with inequality constraints) and the sensitivity of each to perturbations of initialization.

## 5.1 Performance Profiles

We first introduce some necessary background on the visualization and analysis techniques used in this chapter. The primary tool that we use is the “performance profile” approach introduced by Dolan and Moré to compare the relative performance of optimization solvers. In order to generate a performance profile for a set of solvers  $\mathcal{S}$  and a set of problems  $\mathcal{P}$ , first a performance metric  $t_{s,p}$  is chosen and recorded via benchmarking for each pair of solver and problem  $(s,p)$ . One could imagine any number of valid performance metrics, wall time, CPU time, objective value, number of function calls, etc, but the most common tend to be wall time, CPU time, and number of function calls. The next step in generating a performance profile is to extract the performance ratio for each pair:

$$r_{s,p} = \frac{t_{s,p}}{\min \{t_{s,p} : s \in \mathcal{S}\}}. \quad (59)$$

The final important addition of Dolan and Moré here is to then use this ratio to describe a cumulative distribution function (CDF),

$$\rho_s(\tau) = \frac{1}{|\mathcal{P}|} |\{p \in \mathcal{P} | r_{s,p} \leq \tau\}|, \quad (60)$$

which can obviously be viewed as the probability that for a given bound on the ratio, the ratio for a given solver is within that bound.

This piecewise constant probability distribution is very useful in the relative analysis of solvers. In particular, it is very simple to choose an optimal solver in the two primary categories practitioners often care about: raw speed and robustness. This is because to compare solvers on raw speed one can compare the percentage of “wins” a solver has which is represented by  $\rho_s(1)$ , and to compare the robustness of the solvers one needs to simply look at  $\rho_s(\tau_{\max})$  to discern which solver will solve the most problems given infinite time. We however also use a similar CDF formulation

with the raw performance metric on occasion as, given a good knowledge of the data and problem set that can elucidate differences in performance on specific subclasses of problem.

## 5.2 General Experiment Setup

In this section, we cover the common experimental setup that applies to all of the following benchmarks. The benchmarks are run using a Intel Xeon W-2225 4 core processor with a base clock of 4.1 GHz and a boost clock of 4.6 GHz. In all cases where we are not explicitly varying the NLP solver used, the default for NOSNOC is used which is IPOPT. IPOPT is used with default except the settings listed in Table 7, wherever we do not explicitly mention changes. It is particularly important to note that we set the bound-relaxation factor to zero, as such preventing IPOPT from internally relaxing the inequality constraints that we give it.

IPOPT can use a variety of linear algebra solvers, including those provided by HSL, which is a collection of FORTRAN libraries implementing a variety of scientific computing tasks [72]. This collection contains three solvers of interest, in particular, MA27 [73], MA57 [74], and MA97 [75]. We default to using MA27 [73] as the linear solver in IPOPT as it has, in our experience, been the most stable of the HSL solvers. This is due to our observation that both MA57, and MA97, the solvers recommended as state of the art, occasionally cause IPOPT crashes due to segmentation faults. The single threaded nature of MA27 also allows us to run multiple IPOPT instances in parallel in order to improve throughput of the benchmark.

In order to facilitate the parallelization we use the “parfor” functionality of matlab to run four separate MATLAB processes. We measure the performance of each solver primarily using a wall time timer which sums the real time taken to solve each NLP, and ignore any processing time in between, both as it is not relevant to solver performance, and as it is generally equivalent between different solution methods as it

Option	Value
<code>bound_relax_factor</code>	0
<code>mu_strategy</code>	adaptive
<code>mu_oracle</code>	quality-function
<code>acceptable_tol</code>	$10^{-6}$
<code>tol</code>	$10^{-12}$
<code>dual_inf_tol</code>	$10^{-12}$
<code>comp_inf_tol</code>	$10^{-12}$

**Table 7:** IPOPT options that deviate from the default in our experiments.

primarily depends on the size of the problem. While the author admits that in most case wall time is not the optimal metric of performance, the fact that we use a single threaded solver mitigates the overhead effects of the OS-scheduler. In much of the literature surrounding the benchmarking of NLP and MPCC solvers it is popular to use the number of function evaluations as a performance metric. For completeness we include this in the plots for some of the following benchmarks, but would like to note that this is not a particularly useful metric when comparing different reformulations of the same problem rather than just the same problem using different approaches. This is because the different reformulation fundamentally alter the linear algebra that is done by the solver and as such any comparison that uses just the number of function evaluations ignores the fact that one reformulation or another may lead to more numerically pathological matrices and turn the inversion of said matrices into the true performance bottleneck.

### 5.3 Stopping Criteria and Solution Quality

Beyond the NLP stopping criteria used in the underlying NLP solver, use a further stopping criteria on the homotopy iteration that is based on what we call the “complementarity residual”:

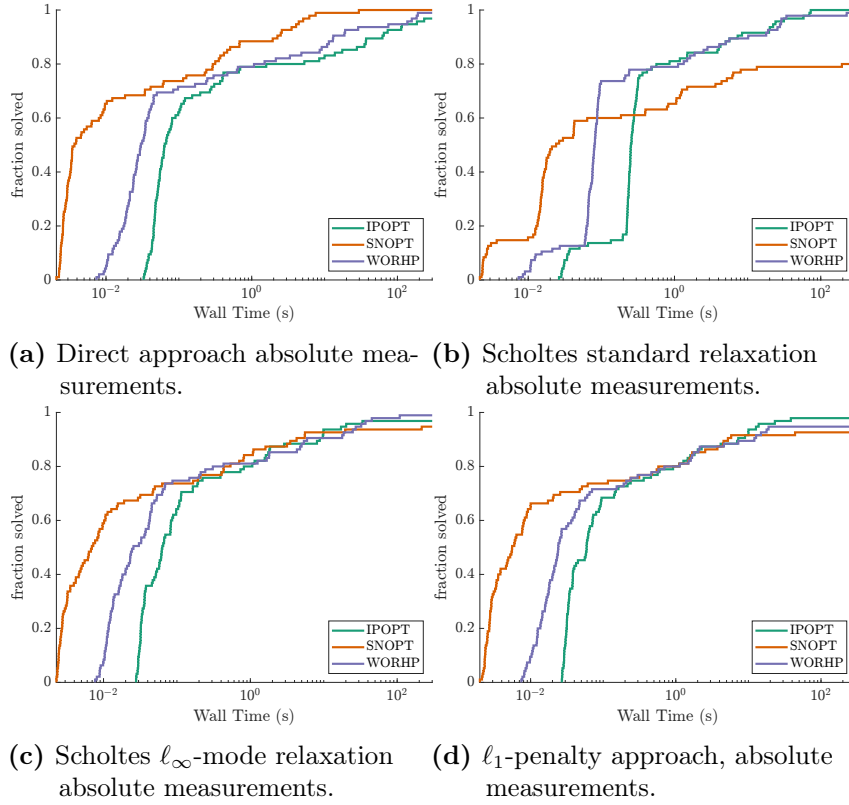
$$r_{\perp}(w) = \max(\{G_i(w)H_i(w) | i = 1, \dots, n_c\}).$$

The stopping condition used is the achievement of a successful NLP solve in the homotopy loop, whose result has a complementarity residual smaller than a given tolerance. For all the following experiments (except if otherwise noted) we use a complementarity residual tolerance of  $10^{-7}$ . In the case of IPOPT we treat any solution that is reported as optimal or “solved to an acceptable level” as a successful solve. We further accept solutions where the search direction becomes too small if they meet the complementarity tolerance as well as are primal feasible.

When analyzing the results in following sections we include in the analysis the quality of a given solution when the problem comes from a discretized OCP. We do not apply this check for simulation problems as it can be proven that for such problems the solutions are sufficiently isolated. This verification is done through checking the relative objective value of a given problem-solver pair against the best known found solution. For this check we only use the “true” objective of the OCP without any augmentation, i.e. any penalty terms or step equilibration heuristics. We then treat solutions which exceed the best known objective by at least a factor of two as failures. This approach is used in order to better evaluate the solution methods specifically in an optimal control context as in this context a significantly worse solution is often a sign of a failure of the solver to achieve the goals of the controller.

## 5.4 Verification of Methods Against MacMPEC

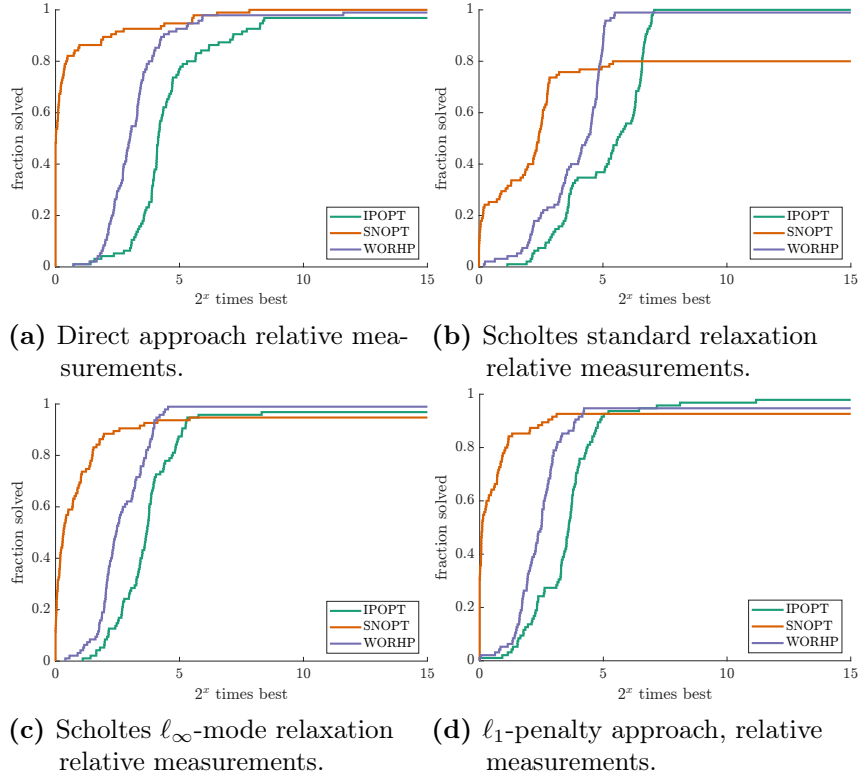
In order to verify the correctness of our methods we further include a test of our implementations against an existing benchmark that is commonly used in prior literature such as [35], [36], and [27]. The problem set for this benchmark is available in the form of a tarball of AMPL [76] format `.mod` and `.dat` files from <https://wiki.mcs.anl.gov/leyffer/index.php/MacMPEC>. We use a modified version of CasADi [59] to extract 95 of the MPEC problems from this benchmark. This is done by first generating `.nl` files for each problem, then reading these in and generating



**Figure 20:** Evaluating several approaches against a section of the MacMPEC benchmark suite.

CasADi MPCCs of the form in Equation (58). We drop any problems that contain complementarities with a “body” parameter of 3, as described in [77], which are slightly more generic than our implementation permits. This test suite is then run on four different approaches:  $\ell_1$ -penalty, standard Scholtes relaxation,  $\ell_\infty$ -mode Scholtes relaxation, and the direct method, using the three NLP solvers we evaluated in a prior section.

We note that several of these approaches solve each of the 95 problems in under ten minutes, and most solve more than 90% of the problem set in the same amount of time. These results are summarized in Figure 20 and Figure 47, and show an interesting trend in comparison to those that will be presented in Section 5.7.3. In particular we see much better performance of the direct method on the smaller problems in this



**Figure 21:** Evaluating several approaches against a section of the MacMPEC benchmark suite using relative speed.

test set, along with much better performance from SNOPT, again tied to the smaller size of the problems.

In general this supports our prior assertions on the relative difficulty of the NOS-BENCH test suite when compared to existing state-of-the-art benchmarks.

## 5.5 Cross Complementarity Modes

We now discuss the aggregation levels of the cross-complementarity modes. This experiment is run on a subset of NOSBENCH that contains problems that use all three cross-complementarity modes (see Table 6). This is augmented with all of these problems additional discretized in the sparsest mode.

As noted in Section 3.4 the different cross-complementarity modes have some different theoretical properties, which may impact the convergence of the relaxation method. In particular one would expect that the sparsest mode (`cc_mode` = 1) would have the poorest performance in terms of robustness due to the constraint qualification violations. We do in fact see this play out in Figure 22 particularly in the  $\ell_\infty$ -mode case. There we see a significant improvement in performance for the denser cross-complementarity modes, and even a minor improvement in robustness.

For the standard relaxation we see something that may seem unexpected though does match our numerical experience. We see much better robustness from one of the two semi-sparse modes `cc_mode` = 3. It is an open question as to whether there is a theoretical reason for this improvement in both speed and robustness. These results and the theoretical result shown in Section 3.4, we do not include problems in the sparsest cross-complementarity mode in the remaining experiments.

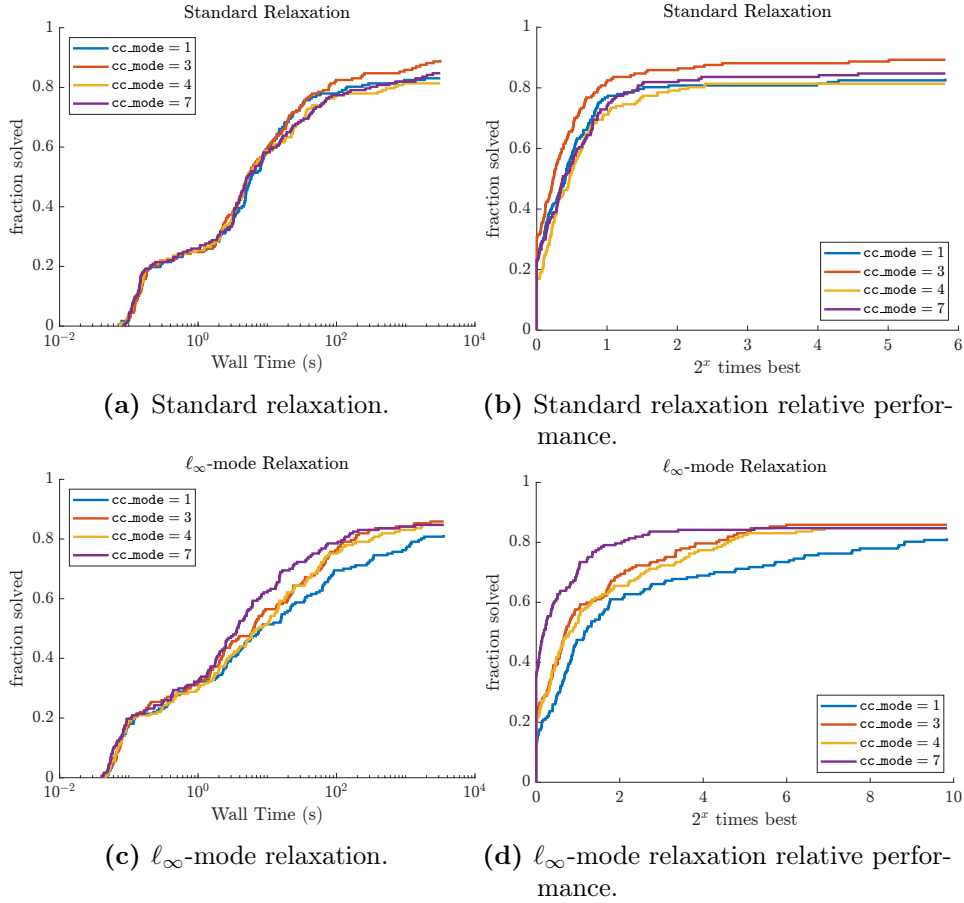
## 5.6 Comparing $\sigma$ Scaling vs No $\sigma$ Scaling

In this section we briefly evaluate the parameter scaling approach discussed in Section 2.2.5, first on the NCP functions that have the same level sets as the Scholtes relaxation. We then evaluate the same for the Steffensen-Ulbrich local relaxation [42], and then the two kinked relaxations of Kadrani et al. and Kanzow-Schwartz [16, 17]. These evaluations are done on the `NOSBENCH-S` test-set and use IPOPT as the NLP solver. The homotopy parameters are fixed for all of the relaxations with  $\sigma_0 = 1$  and  $\kappa = 0.1$ , and we use the “linear” update rule, i.e.,  $\sigma_{k+1} = \kappa\sigma_k$ . We experiment with this change for both the standard homotopy and the  $\ell_1$  and  $\ell_\infty$  modes.



### Smoothed Nonlinear Complementarity Functions

We observe some insightful results from the comparisons of the scaled and unscaled versions of the NCP function relaxations. While the standard relaxation method does not see a significant improvement in any of the three NCP function approaches, there is a significant improvement in the elastic modes both in terms of speed of convergence and robustness. In the case of the Natural Residual for example in Figure 24 we see that the scaled version of the relaxation performs both consistently faster winning outright on almost 80% of problems for both elastic modes, and solving significantly more problems. This is mirrored almost exactly in the performance



**Figure 22:** Evaluating cross complementarity modes.

of the Fischer-Burmeister function which sees the same improvements in its elastic modes. On the other hand the Chen-Chen-Kanzow relaxation only sees improvement from  $\sigma$  scaling when it comes to the  $\ell_1$ -mode relaxation. Recall that we treat the Chen-Chen Kanzow relaxation in a unique way, only scaling the  $\sigma$  in its quadratic form that corresponds to the Fischer-Burmeister component of the NCP.

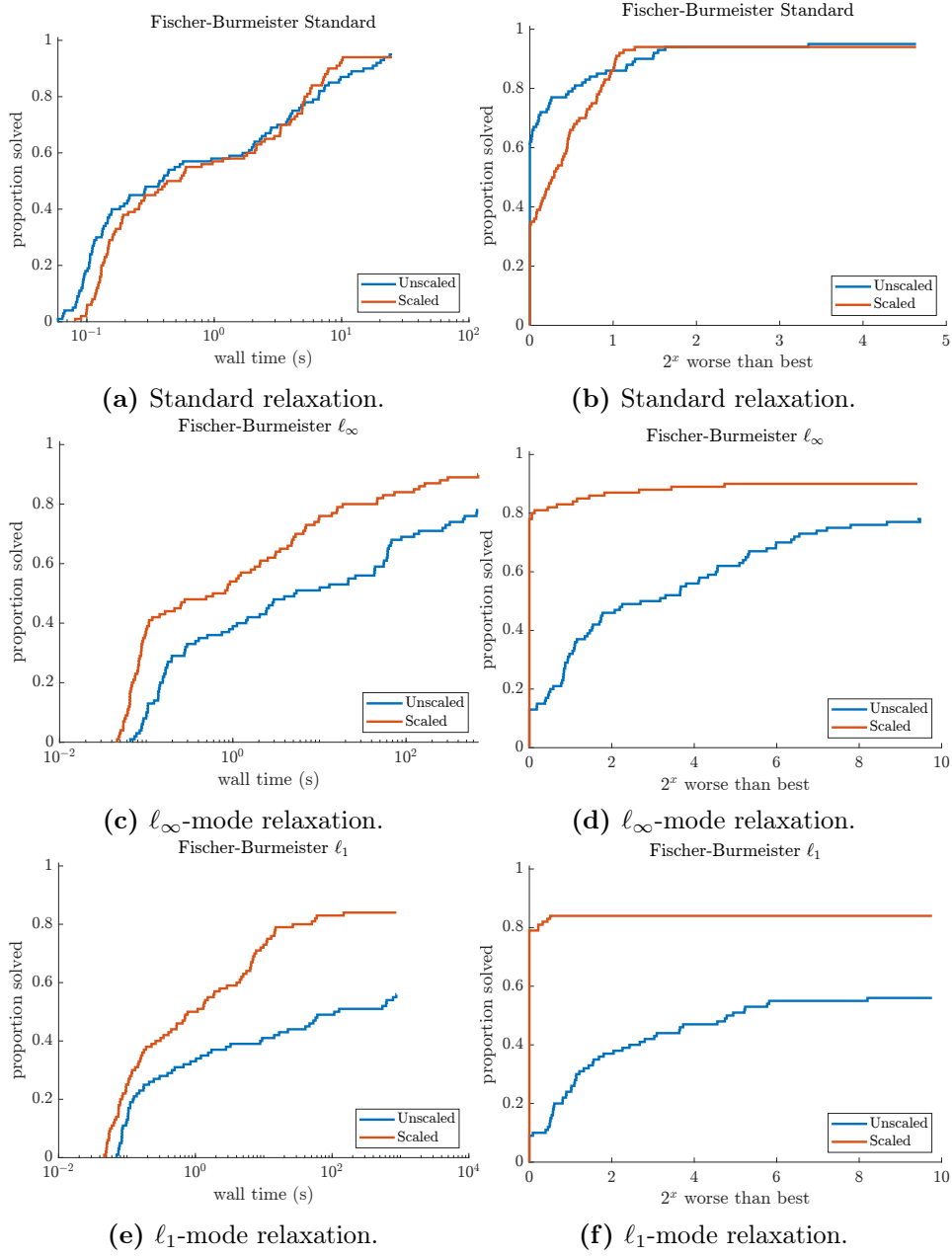
This performance improvement is unsurprising as it brings it closer to the Scholtes relaxation in performance, as seen in Figure 31 and Figure 32. This is because it can be shown that the each of these NCPs with the scaling apply has exactly the same zero-level sets as the Scholtes relaxation as a function of  $\sigma$ . As such the only difference between these and the Scholtes relaxation becomes the gradient of the function with respect to its inputs, the unscaled versions of which we list in Table 8. Due to these results we choose to use the rescaled relaxation for the Natural Residual, Fischer-Burmeister, and Chen-Chen-Kanzow, relaxation.

### The Lin-Fukushima Relaxation

For the Lin-Fukushima relaxation we see little improvement for the standard relaxation from the rescaling, as with the NCP functions. In addition we see significant improvement in performance for the  $\ell_\infty$ -mode relaxation but a decrease in performance for the  $\ell_1$ -mode relaxation. The improvement of the  $\ell_\infty$ -mode relaxation is somewhat expected as we see the same change in  $\rho(\sigma)$  as we do for the NCP functions, however the lack of commensurate improvement in the  $\ell_1$ -mode is not so

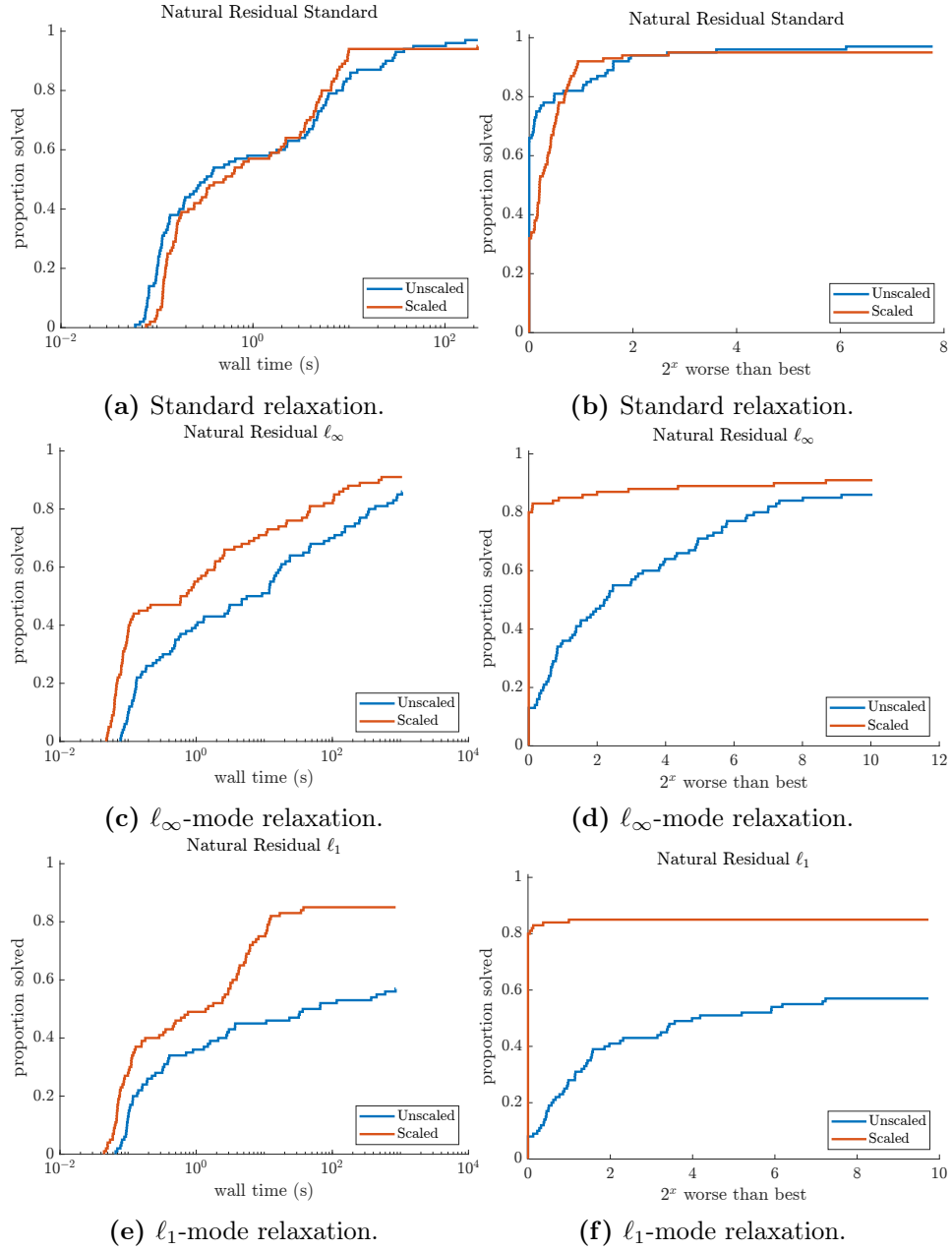
Name	$\psi(a, b, \sigma)$	$\nabla_a \psi(a, b, \sigma)$	$\nabla_b \psi(a, b, \sigma)$	$\nabla_\sigma \psi(a, b, \sigma)$
Scholtes	$ab - \sigma$	$b$	$a$	$-1$
NR	$\frac{a+b-\sqrt{(a-b)^2+\sigma^2}}{2}$	$\frac{1}{2} - \frac{a-b}{2\sqrt{(a-b)^2+\sigma^2}}$	$\frac{1}{2} + \frac{a-b}{2\sqrt{(a-b)^2+\sigma^2}}$	$\frac{-\sigma}{2\sqrt{(a-b)^2+\sigma^2}}$
FB	$a + b - \sqrt{a^2 + b^2 + \sigma^2}$	$1 - \frac{a}{\sqrt{a^2+b^2+\sigma^2}}$	$1 - \frac{b}{\sqrt{a^2+b^2+\sigma^2}}$	$\frac{-\sigma}{\sqrt{a^2+b^2+\sigma^2}}$

**Table 8:** Gradients of NCP functions.



**Figure 23:** Evaluating  $\sigma$  scaling for the Fischer-Burmeister relaxation.

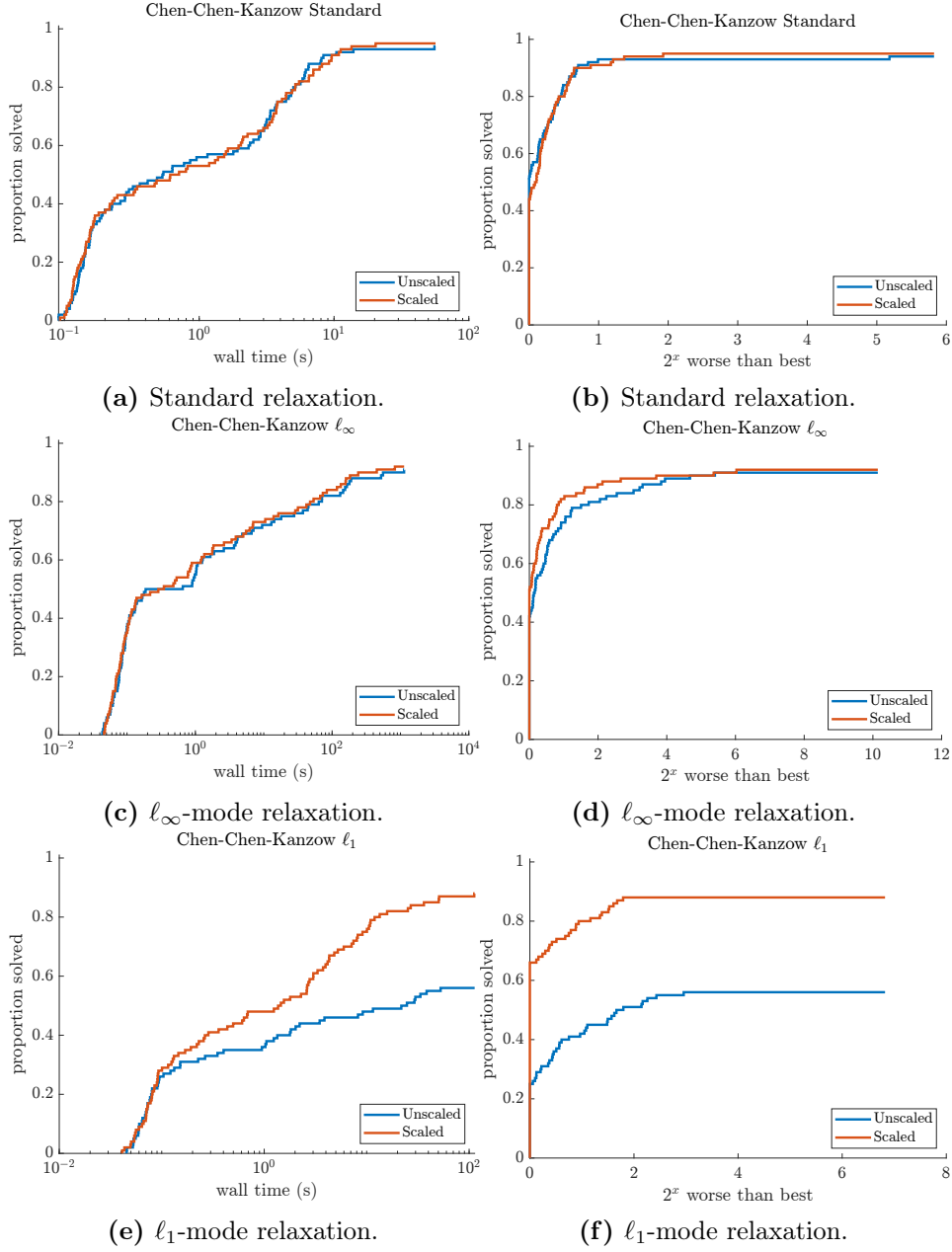
easily explained. Based on the results (seen in Figure 26) we choose to use  $\sigma$ -scaling for only the  $\ell_\infty$ -mode of the Lin-Fukushima relaxation.



**Figure 24:** Evaluating  $\sigma$  scaling for the Natural Residual relaxation.

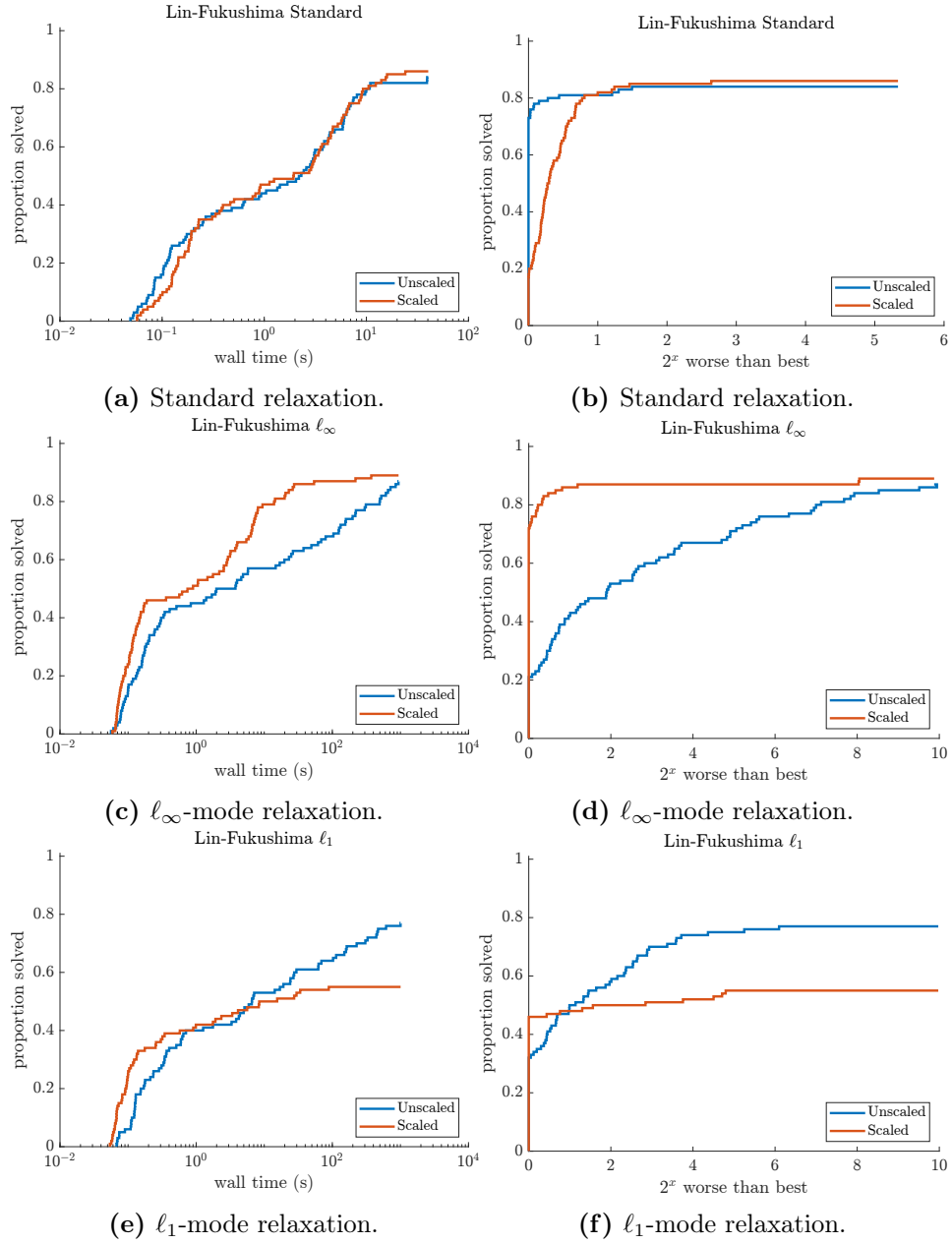
### The Steffensen-Ulbrich Local Relaxation

Here we also see a significant improvement due to the  $\sigma$  rescaling, but contrary to the NCP functions, this improvement is primarily seen in the standard relaxation method.



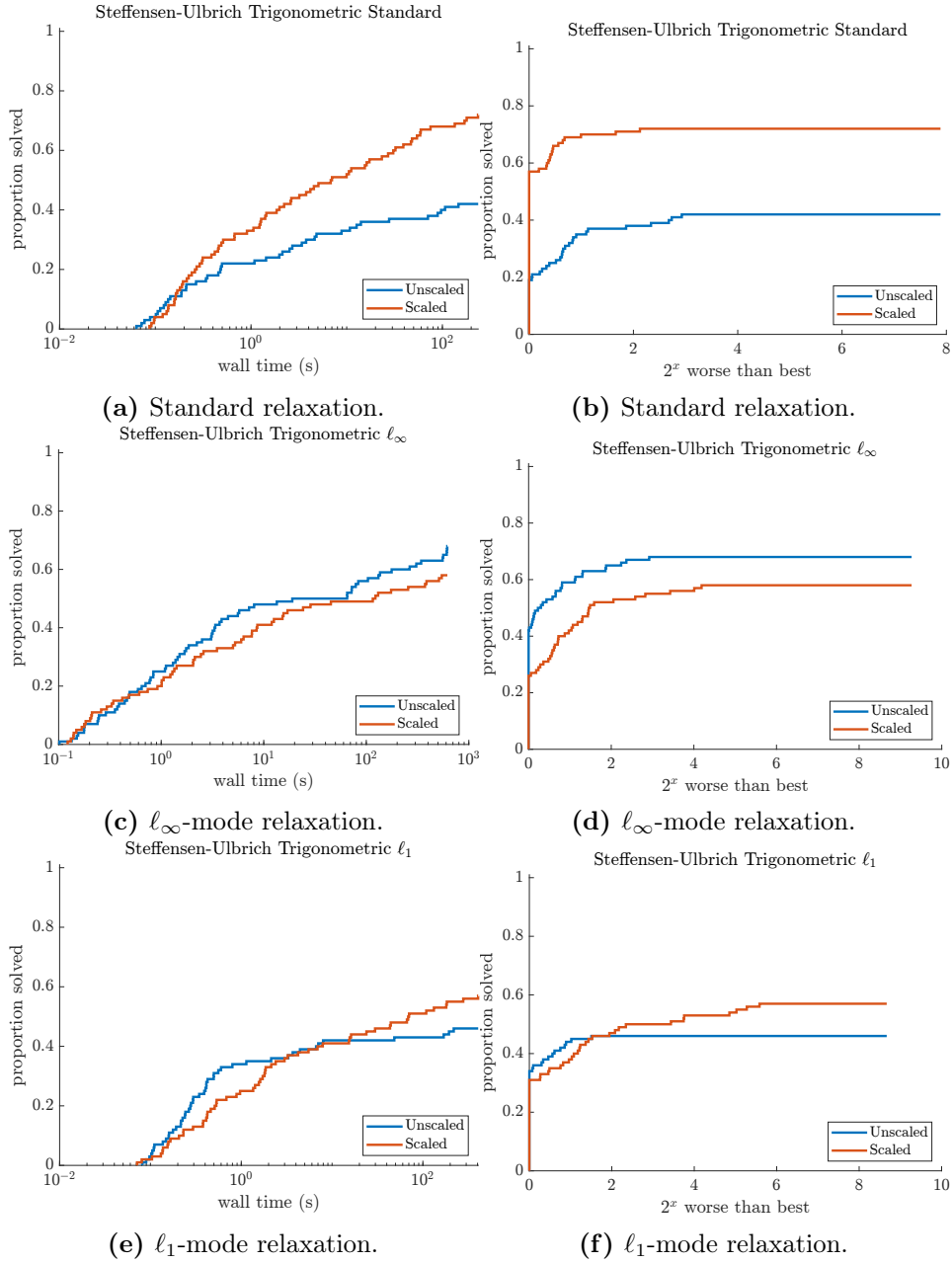
**Figure 25:** Evaluating  $\sigma$  scaling for the Chen-Chen-Kanzow relaxation.

We posit that this is simply due to the fact that without the scaling this relaxation only very slightly relaxes the problem and in many cases the NLP solver simply cannot find a feasible solution. As such we also proceed to use the  $\sigma$  re-scaling technique for the Steffensen-Ulbrich relaxation for the standard relaxation in the following

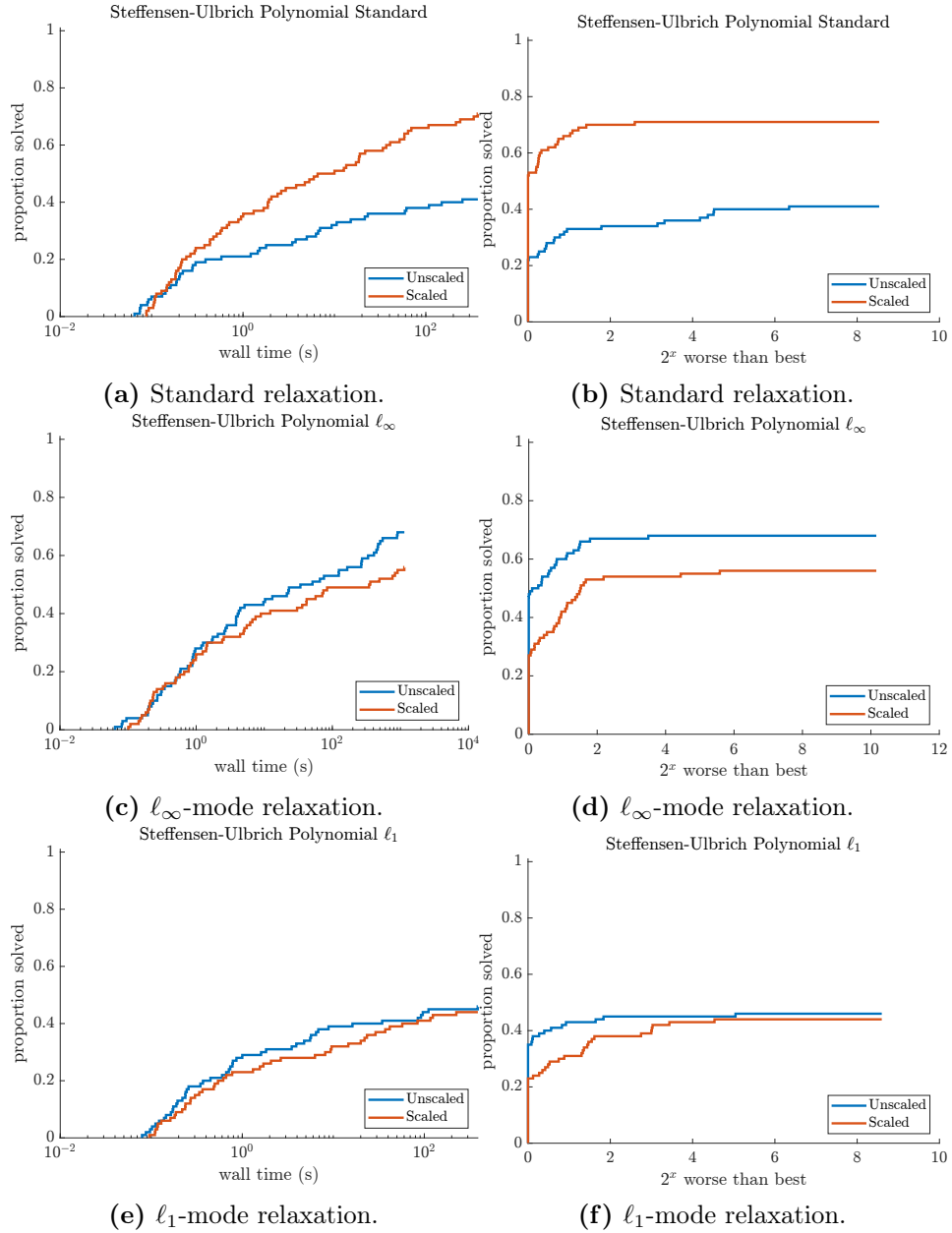


**Figure 26:** Evaluating  $\sigma$  scaling for the Lin-Fukushima relaxation.

experiments, and an unscaled version for the two elastic modes. The performance profiles can be seen in Figure 29 and in Figure 30.



**Figure 27:** Evaluating  $\sigma$  scaling for the Steffensen-Ulbrich trigonometric relaxation.

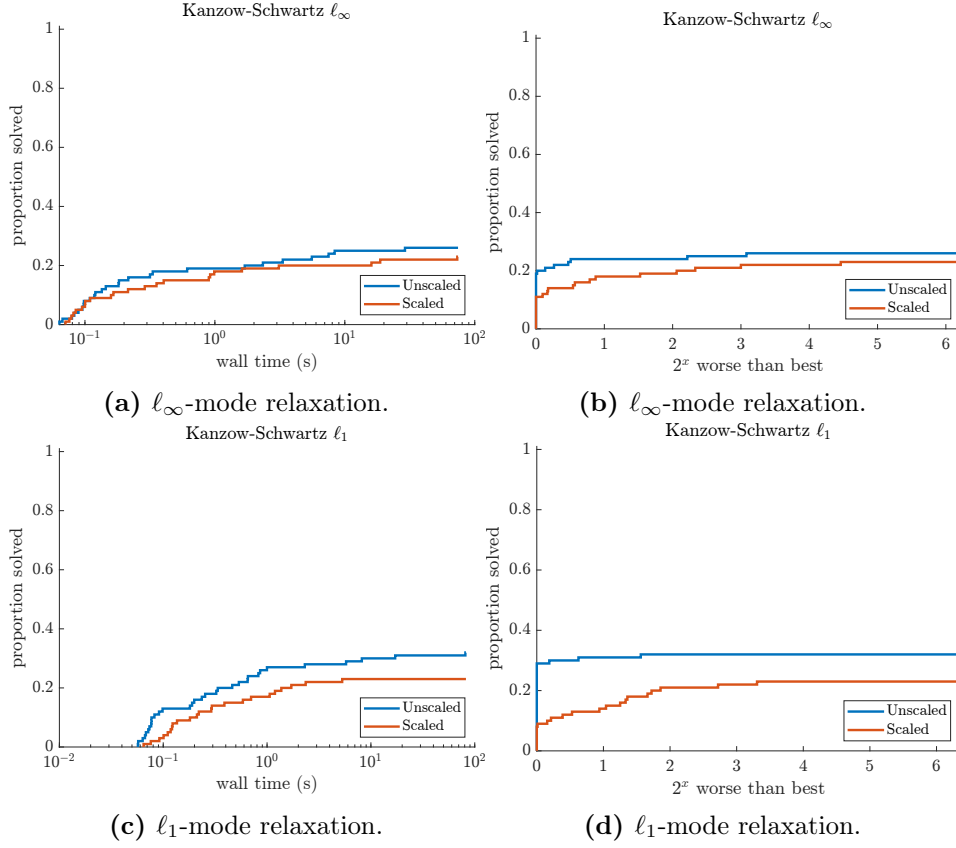


**Figure 28:** Evaluating  $\sigma$  scaling for the Steffensen-Ulbrich trigonometric relaxation.

### Kinked Relaxations

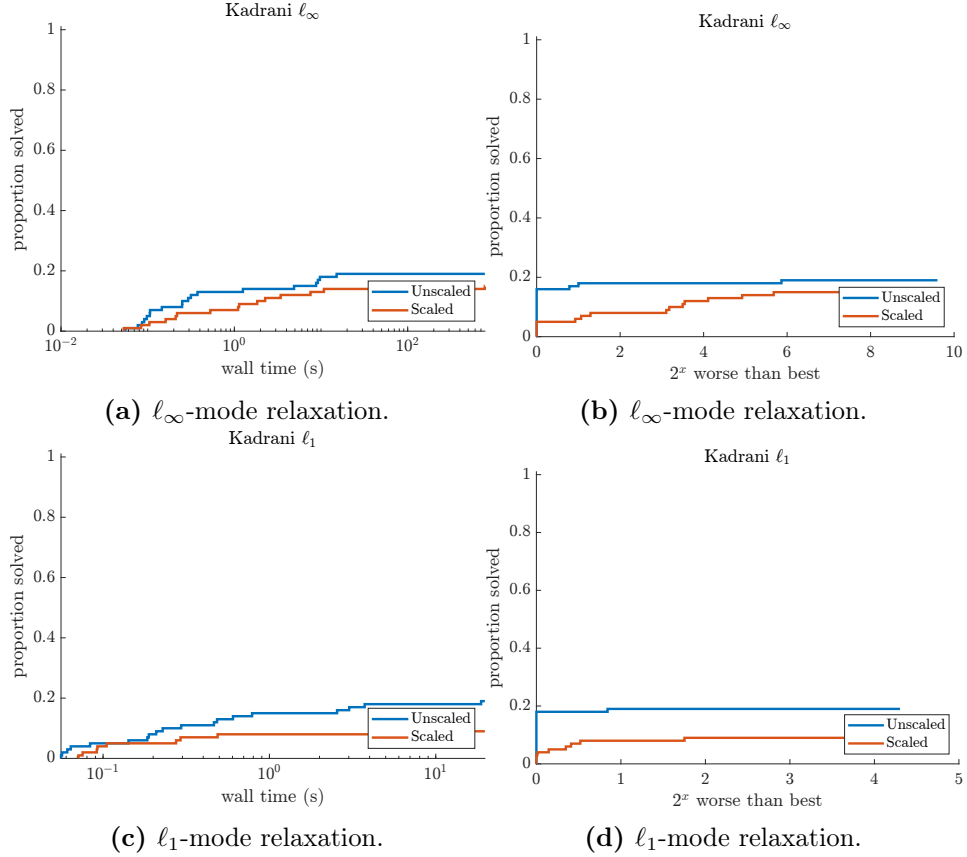
The kinked relaxations pose a different challenge when it comes to doing sigma re-scaling as their structure permits an unbounded complementarity residual. As





**Figure 29:** Evaluating  $\sigma$  scaling for the Kanzow-Schwartz nonsmooth relaxation.

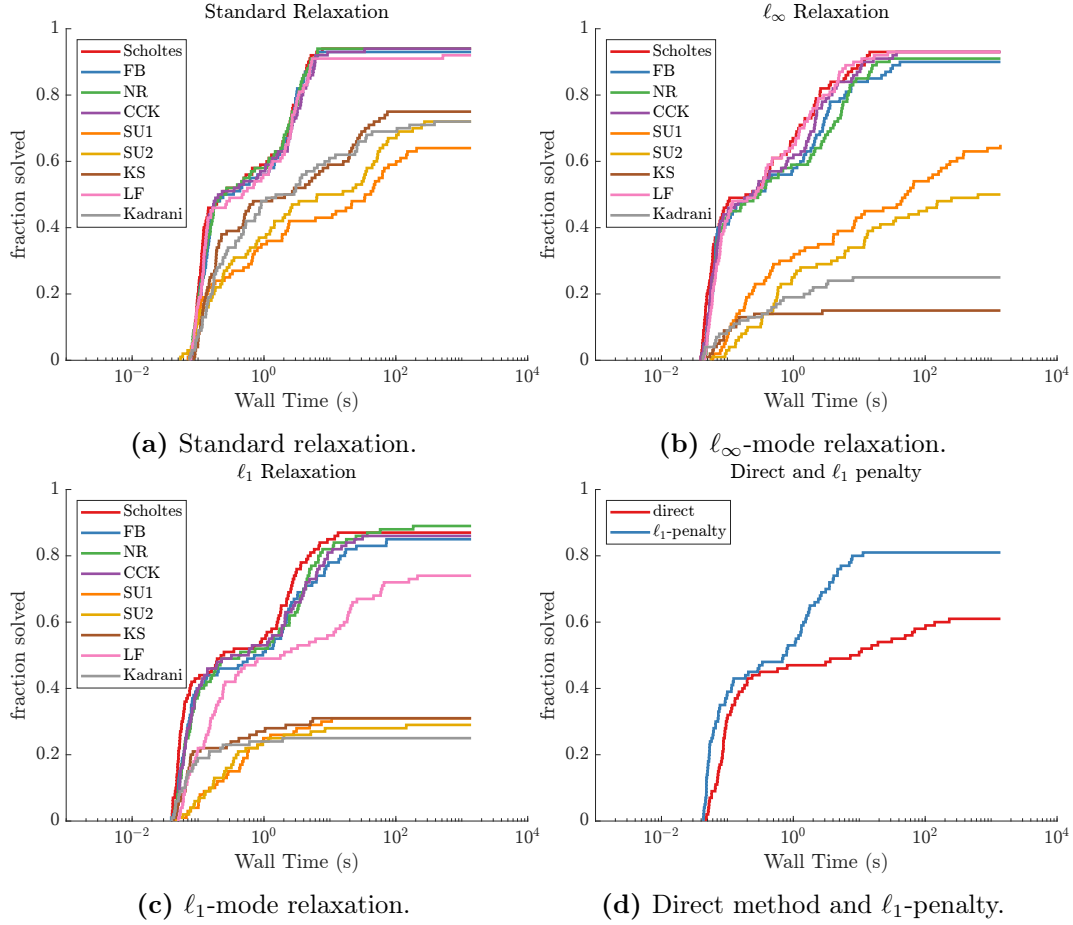
such, in the case of both the Kanzow-Schwartz and Kadrani relaxation we see that if we scale the sigma as suggested via normalizing  $\rho(\sigma)$ , we do not converge to a sufficiently small complementarity residual in the maximum number of outer loop iterations. Therefore, we provide only the plots for the  $\ell_\infty$  and  $\ell_1$ -mode relaxations for these relaxations. However even for these we see only detrimental effects from the rescaling and as such we do not use the rescaling for any of the approaches using the kinked relaxations.



**Figure 30:** Evaluating  $\sigma$  scaling for the Kadrani nonsmooth relaxation.

## 5.7 Improved Relaxation Solver Parameters

In this section we discuss four experiments which pertain to the relaxation method solver that is implemented in the software package NOSNOC. Namely we first explore and compare the types of relaxations described in Section 2.2.1, along with the methods for steering the complementarity relaxation described in Section 2.2.3. We further include in this comparison, as “control” solvers, the direct method, i.e. replacing  $a \perp b$  with the constraint  $ab \leq 0$ , and an exact- $\ell_1$ -penalty method without slacks introduced. As we discussed in Section 2.2.3, the algorithms implemented in NOSNOC for solving MPCCs have several free parameters, namely  $\sigma_0$ ,  $\kappa$ , and  $\zeta$ . In the second and third experiments we vary these parameters for both an  $\ell_\infty$  and a



**Figure 31:** Evaluating relaxation methods for MPCCs.

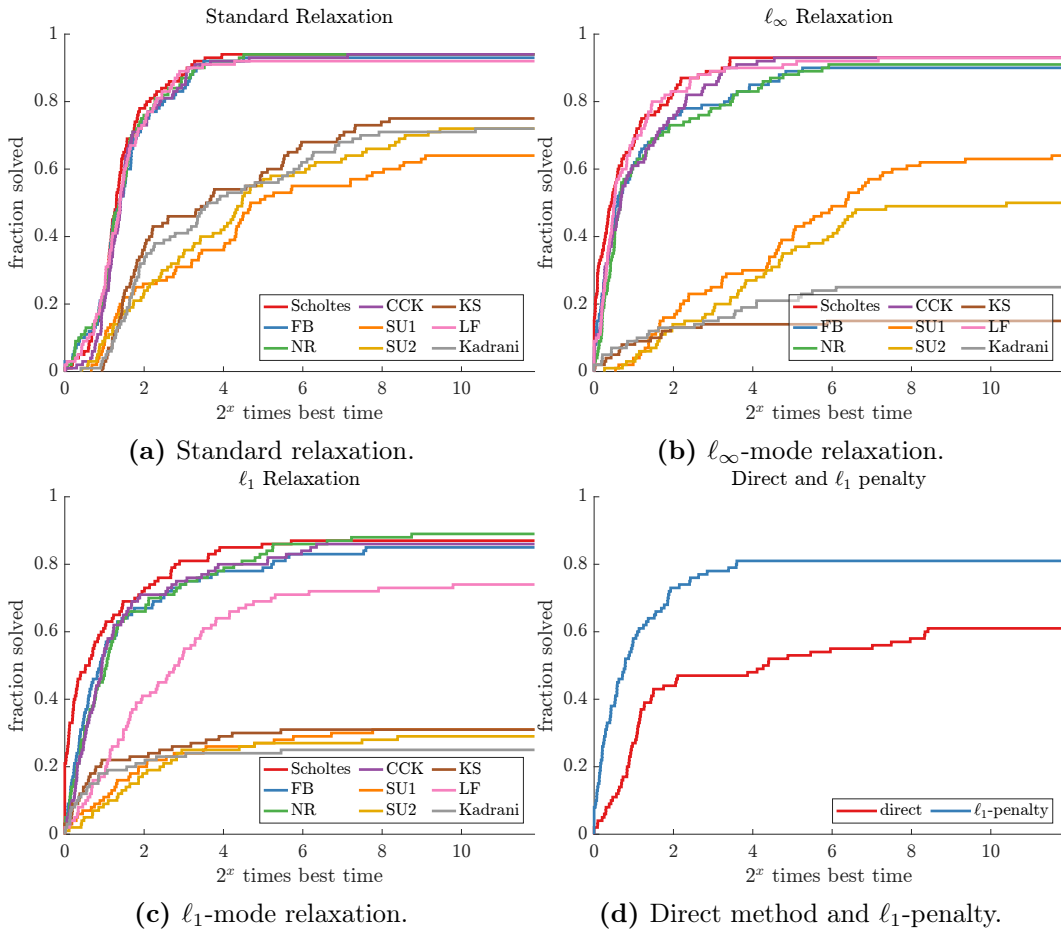
standard homotopy approach in order to evaluate a good set of default parameters for most NOSNOC users. We finally compare three NLP solvers' (IPOPT [60], SNOPT [61], and, WORHP [62]) performance on our particular problems.

### 5.7.1 Relaxation Benchmark

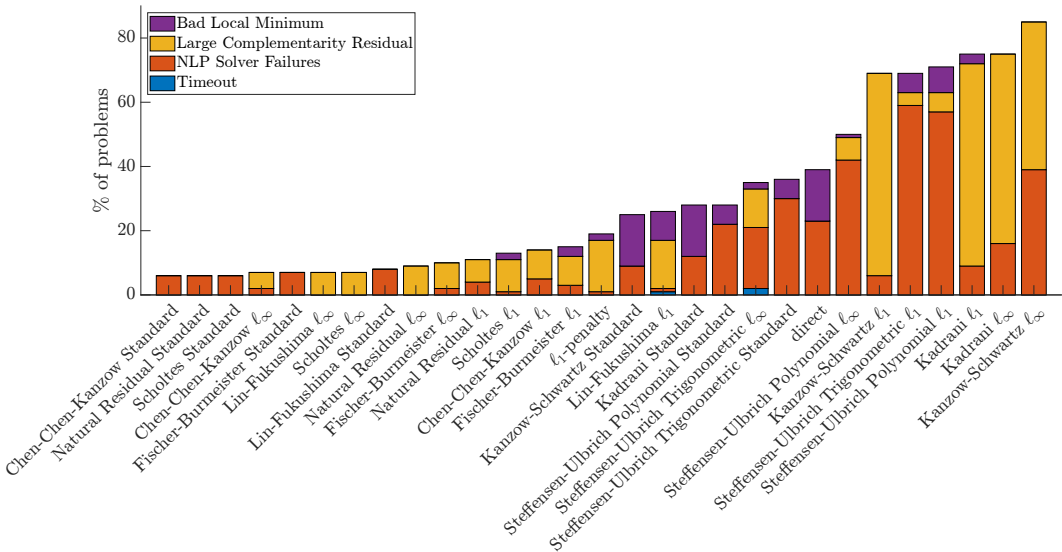
In order to evaluate the practical performance of the various relaxation functions  $\psi(a, b, \sigma)$  and approaches of driving the relaxation parameter to zero, we run the set of 29 methods on the NOSBENCH-S test set, solving 2900 MPCCs in the process. In this experiment we use the homotopy parameters  $\sigma_0 = 1$ ,  $\kappa = 0.1$ , and the linear

update rule. The NLP solver IPOPT is used to solve the regularized NLPs with default settings outside of those explicitly mentioned in Section 5.2 and a one hour (3600 second) cumulative wall time limit on each problem, implemented as described in that section. An overview of the absolute performance of each solver can be seen in Figure 31.

The first notable result is that while for about half the problems in this set the direct approach solves the problem in an acceptable time-frame, it quickly stalls at that point. It's failures as seen in Figure 33 are fairly evenly split between converging to unacceptable local minima, and failing to converge at all, primarily due to either



**Figure 32:** Evaluating relaxation methods for MPCCs, relative performance.



**Figure 33:** Failure reasons for different relaxation methods.

step calculation failures at unacceptable points, or due to claiming infeasibility. This already points to the usefulness of the homotopy methods as it is clear to see that those methods perform better on a large proportion of problems with little to no impact on absolute performance.

The general outcome of this benchmark is a victory for the Scholtes relaxation approach in all of its three versions. From the absolute performance plots we can clearly see that for all three methods of steering the relaxation parameter, the Scholtes relaxation successfully solves almost as many or more problems than the other relaxation methods.

We can then first analyze the relaxations using the standard approach to drive the relaxation parameter to zero. Here we see a performance lead for the Scholtes relaxation, albeit a very slim one. For this method relaxations can be split into three different groups based on performance (in order): the group containing the Scholtes relaxation as well as those that use NCP functions, Lin-Fukushima and Kanzow-Schwartz which are nearly as fast but plateau earlier, and the Steffensen-Ulbrich and Kadrani relaxations which gain only about 10% robustness over the direct method

but are somewhat slower. We note that Figure 32 suggests that we see very few outright victories for the standard method, however it is the most robust, achieving the highest fraction of problems solved, 95%.

We then move on to our analysis of the  $\ell_\infty$ -mode relaxation where we see that the Scholtes version of this relaxation wins on the largest fraction of solvers. It maintains its lead but only reaches a solution on about 92% of the problems. Once again we see that the NCP function relaxations approach the performance of the Scholtes relaxation and match it in robustness. We also see the Lin-Fukushima relaxation perform well again but not quite at the level of the Scholtes and NCP group. On the other hand, we see extremely poor performance from the Steffensen-Ulbrich relaxations and the kinky relaxations of Kadrani et al. and Kanzow and Schwartz. We posit that this is due to poor performance of the NLP solver when it comes to the sensitivity of the relaxation with respect to the relaxation parameter  $s$ . From Figure 33 we see that both the Kadrani and Kanzow-Schwartz relaxations fail frequently with a point which the NLP solver claims is optimal, but still has a large complementarity residual.

Finally, we analyze the performance of the  $\ell_1$ -mode relaxations and the  $\ell_1$ -penalty formulation. The  $\ell_1$ -penalty method nearly ties the  $\ell_\infty$  Scholtes relaxation in absolute wins, however it fails to break the 90% of problems solved. This performance is nearly mirrored by the  $\ell_1$ -mode Scholtes relaxation albeit without the outright victories of the penalty method. Here we also see the second major gap between the Scholtes relaxation and the NCP function based relaxations, though they remain the best performing group. The remaining relaxations perform worse than even the direct method and as such can be neglected.

In general it is clear to see that the best methods in terms of speed and robustness are the  $\ell_\infty$ -mode Scholtes relaxation and the standard Scholtes relaxation respectively. The latter is certainly the choice for robustness as we will show in future experiments as it's robustness can even be improved by adjusting the  $\sigma_0$  and  $\kappa$  parameters which govern the trajectory of the relaxation parameter.

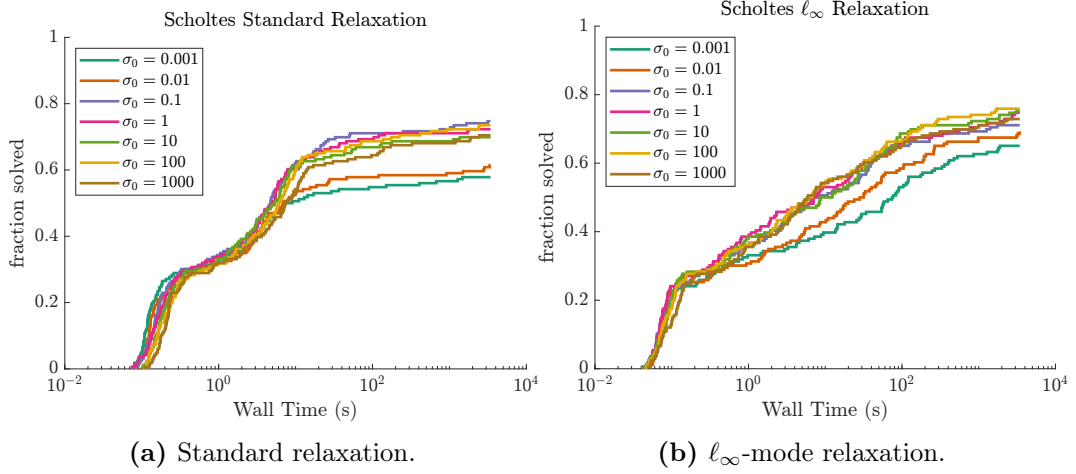
### 5.7.2 Homotopy Parameters

As described in the previous section we note that the Scholtes relaxation is the optimal choice in its standard and  $\ell_\infty$  forms. In order to elucidate a default set of parameters we run an experiment on the NOSBENCH-RL test set, first varying the initial relaxation parameter  $\sigma_0$  and then vary  $\kappa$ , the slope at which we drive the relaxation parameter to zero. These two parameters have a moderate influence on the stability and speed of the homotopy solver converging to an acceptable solution.

We first discuss the effects of  $\sigma_0$  the performance plots for which can be seen in Figure 35. The first major takeaway from the analysis of the performance plots is that this has a much less significant effect on the  $\ell_\infty$ -mode relaxation. Intuitively this does make sense as in this mode we simply use a penalty factor  $\frac{1}{\sigma}$  in order to drive the complementarity residual to zero. It is, as such, not a limiting factor on the complementarity residual in each step, depending of course on the scaling of the problem at hand. On the contrary, it is observed that for the standard relaxation the solvers converge to points with a complementarity residual exactly equal to  $\sigma$  especially in all Scholtes and NCP relaxation schemes. However, minimal effect is not no effect and we still see a penalty to convergence if we choose a  $\sigma_0$  that is sufficiently small. It is notable that this reduction in performance primarily comes from convergence to significantly worse local minima as seen in Figure 38.

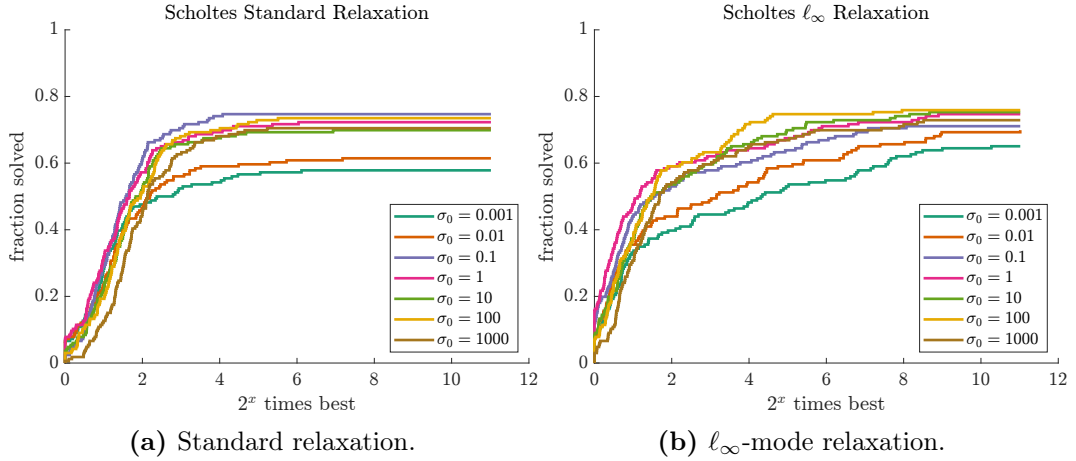
On the other hand we see much earlier and much more pronounced decay in performance for the standard Scholtes relaxation technique. We see almost a 30% reduction in number of problems solved and from Figure 38 we see that the primary reason for failure is a failure of the NLP solver to converge to a good local minimum. In general the homotopy procedure will tend to be stuck closer to the initialization if we take a smaller  $\sigma_0$  which matches what is observed in smoothing methods when small smoothing parameters are used [51].

The effect of  $\kappa$  on convergence is surprisingly minimal compared to the initial



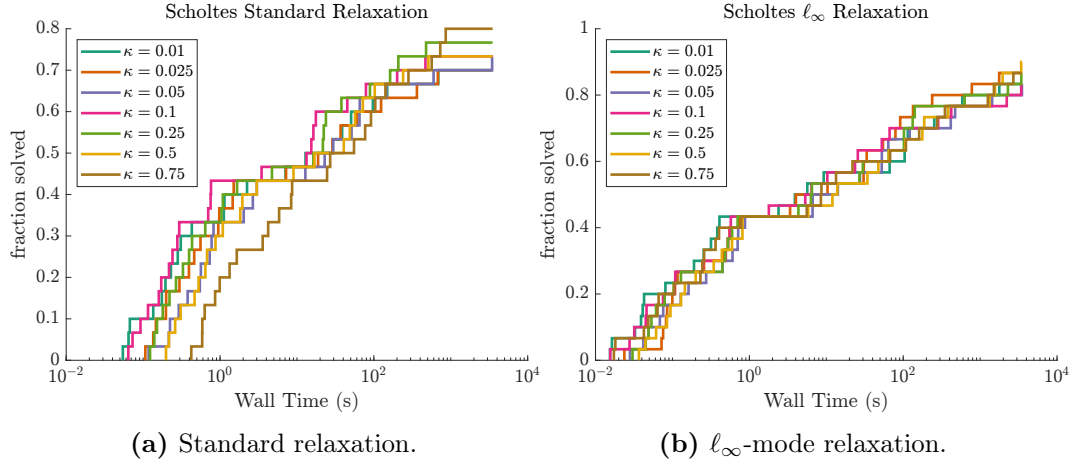
**Figure 34:** Evaluating  $\sigma_0$ , the initial homotopy parameter.

relaxation parameter. In particular we see essentially no difference in performance for the  $\ell_\infty$ -mode relaxation. This is very likely due to the fact that for a majority of problems we see only several (and occasionally only one) homotopy iterations before the solver converges to an acceptable complementarity residual, which can be observed in Figure 45 and Figure 46 in the appendix. The standard relaxation does clearly show both the weakness and the strength of a relatively slow homotopy. We note that for smaller  $\kappa$  the problem converges quicker due to having to take less



**Figure 35:** Evaluating  $\sigma_0$ , the initial homotopy parameter with relative performance.



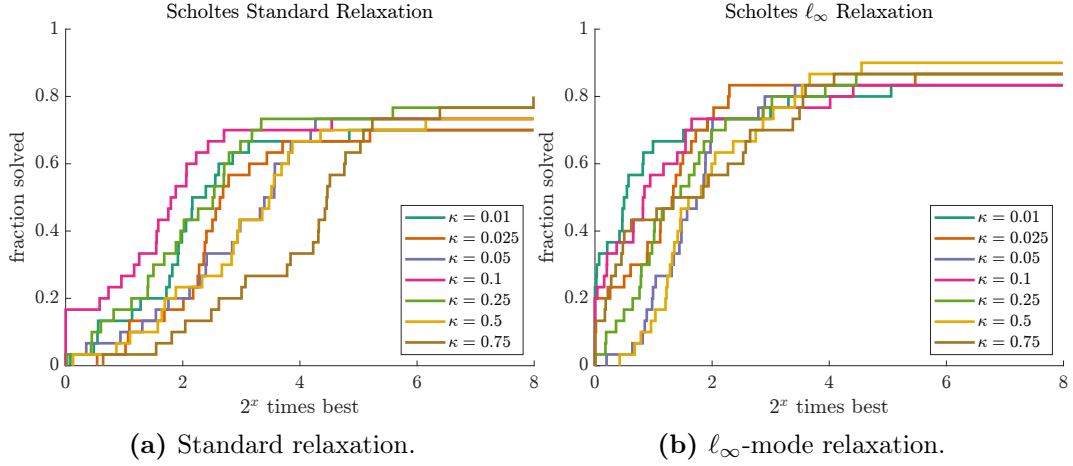


**Figure 36:** Evaluating  $\kappa$ , the initial homotopy slope parameter.

homotopy iterations to converge to a sufficient solution. We also see that this benefit disappears as you get to more difficult problems, i.e. OCPs. This is likely because of the observed behavior that many of the intermediate NLP solves for the larger  $\kappa$  are shorter due to more easily being able to follow the path to the solution. This also is likely caused by a smaller initial infeasibility caused by the updated relaxation parameter. We also see a mild improvement (around 10%) in the number of problems that are solved with the larger  $\kappa$ . This makes it likely that a larger homotopy update slope is particularly useful for ensuring convergence for more difficult problems while a smaller  $\kappa$  (or use of the  $\ell_\infty$ -mode) is the superior option for simulation problems.

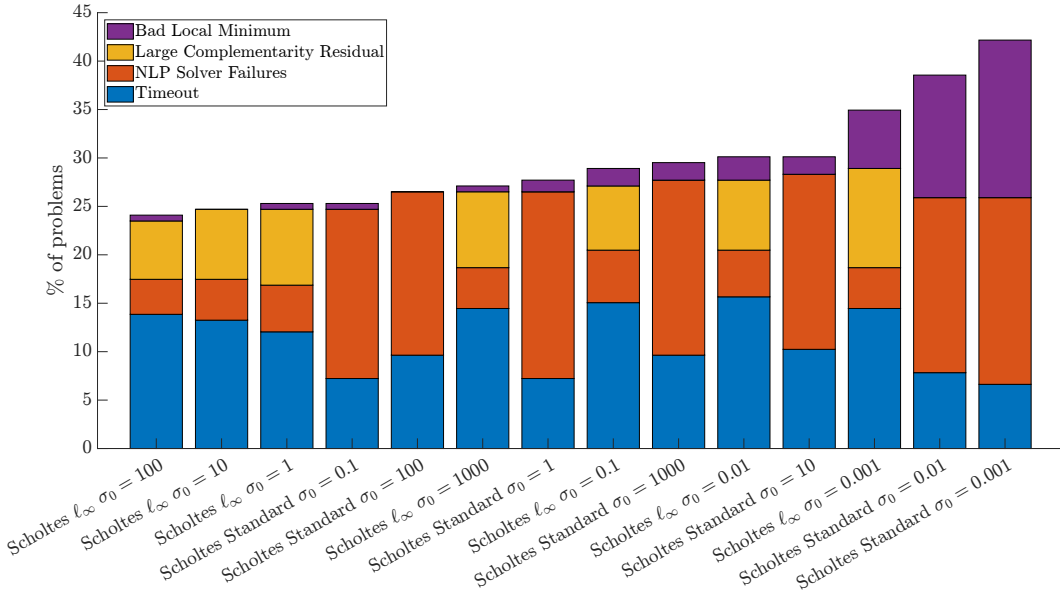
### 5.7.3 NLP Solvers

We also test several popular NLP solvers as engines for the homotopy solver on the full NOSBENCH test set. In this case we use the existing default homotopy parameters  $\sigma_0 = 1$  and  $\kappa = 0.1$  and use the standard default settings for both SNOPT and WORHP, except for those related to maximum iterations and timeouts which are set as for IPOPT. SNOPT is also used in its warm-start mode, however as WORHP labels its warm-starting mode as experimental we do not make use of it.

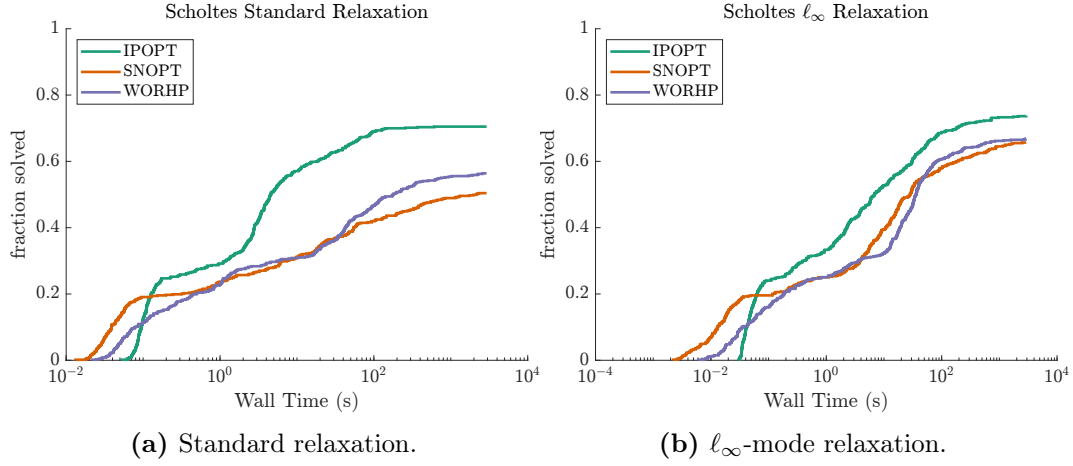


**Figure 37:** Evaluating  $\kappa$ , the initial homotopy slope parameter with relative performance.

We observe some significant wins for SNOPT, particularly on smaller problems, which is expected due to the specialization of active set methods on small to medium size problems. This speed comes at the cost of the worst robustness out of the three solvers for both relaxation versions. WORHP on the other hand is not particularly fast but is more robust than SNOPT, and this gap is more prevalent in the standard



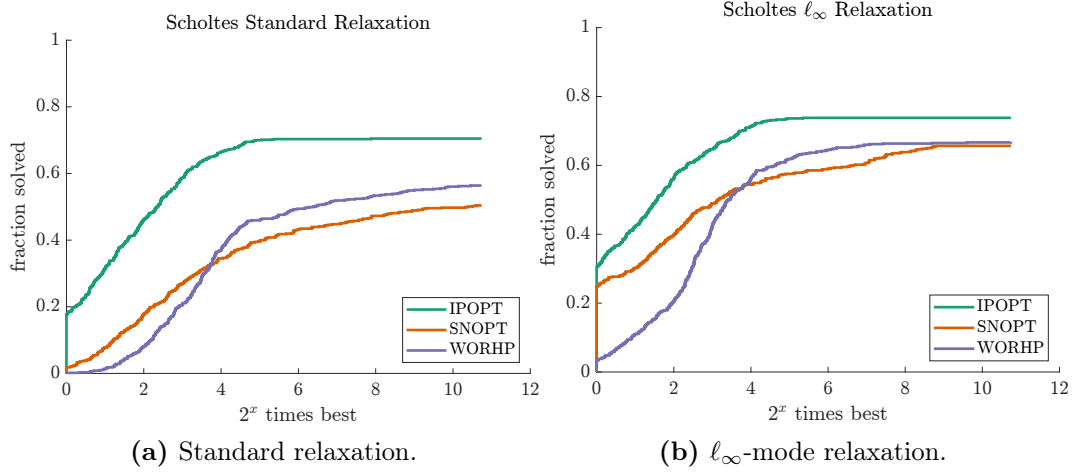
**Figure 38:** Failure reasons for different values of  $\sigma_0$ .



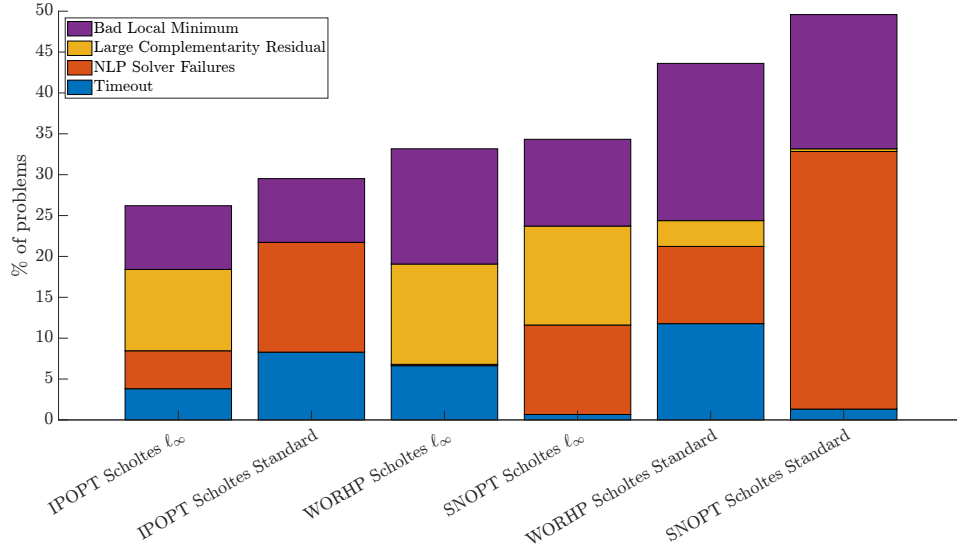
**Figure 39:** Evaluating different NLP solvers.

relaxation. It is however clear to see that IPOPT is by far the winner here with both the most overall wins in the  $\ell_\infty$ -mode with it solving 30.5% of problems the fastest, and in terms of robustness where it successfully solves 73.8% of the problem set.

It is possible that further tuning may improve the performance of each of these solvers, however in their current state IPOPT is a clear choice.



**Figure 40:** Evaluating different NLP solvers with relative performance.

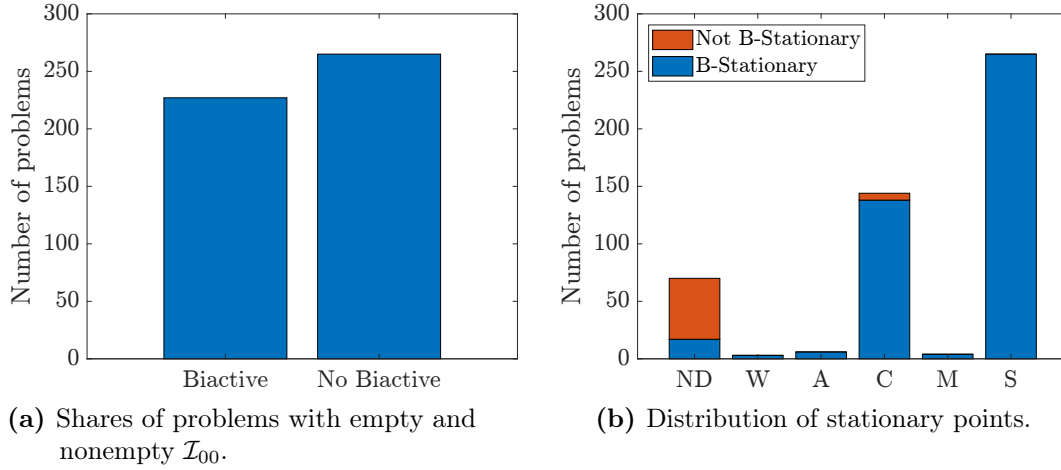


**Figure 41:** Failure reasons of different NLP solvers.

### Stationarity

We use this benchmark of the full test set to evaluate the stationarity properties of the points our methods converge to. In Figure 42a we show the number of problems with an empty and nonempty bi-active set  $\mathcal{I}_{00}$ . Recall that this implies the S-stationarity of the point as described in Section 2.1. We then solve a TNLP for problems with nonempty bi-active set  $\mathcal{I}_{00}$ , and extract the multipliers  $\nu$  and  $\xi$  to calculate the type of stationarity. This bi-active set is calculated via checking the values of  $G(w)$  and  $H(w)$  and comparing this with the square root of the complementarity tolerance. An iterative approach is then applied, wherein we try to solve the TNLP with a fixed active set, and if this fails we iteratively remove the complementarity pair which is furthest from the origin. In some cases however this approach still fails and we label these points as “ND” or “not determined”.

We then evaluate the B-stationarity of each of these points. In the case of points where the TNLP converged, we use the improved point from the output of the TNLP to check B-stationarity, and in cases where the TNLP fails we use the original point that the solver converged to. This B-stationarity check is done through solving a linear



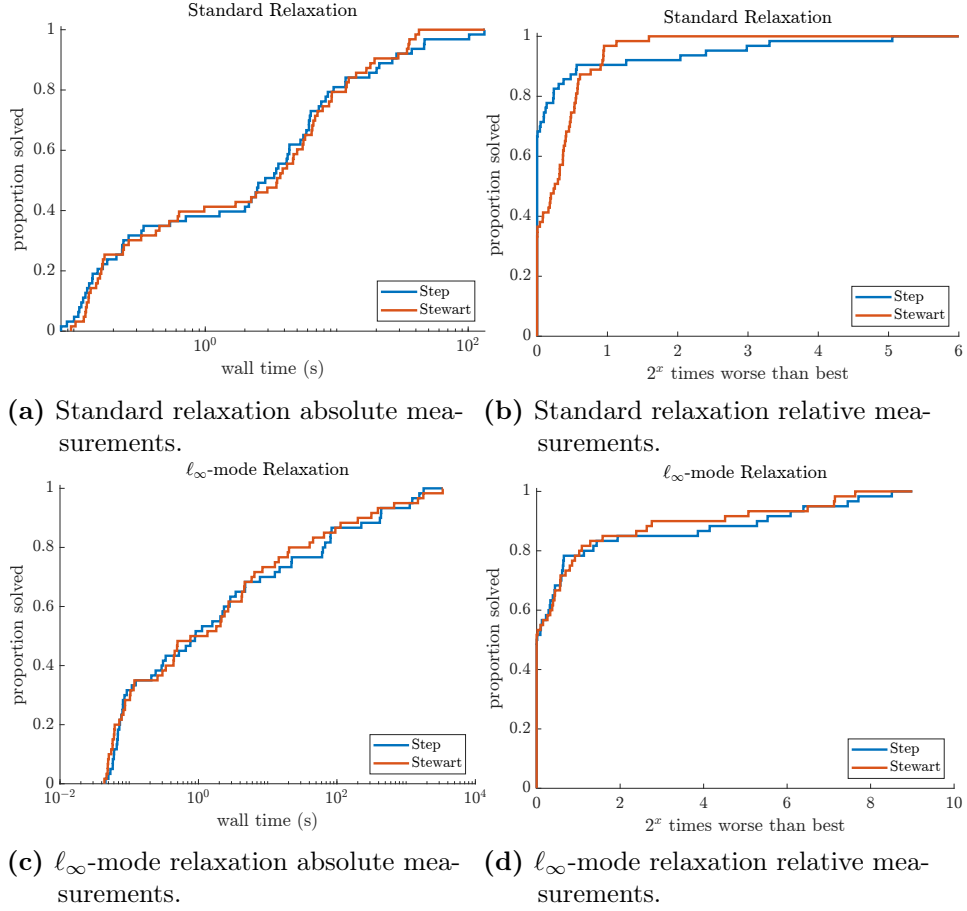
**Figure 42:** Evaluating type of stationarity points in the NOSBENCH-F test set.

program with complementarity constraints (LPCC) that arises from Equation (4):

$$\min_{d \in \mathbb{R}^n} \quad \nabla f(w^*)^\top d \quad \text{s.t.} \quad d \in \mathcal{F}_{\Omega^{\text{MPCC}}}^{\text{MPCC}}(w^*). \quad (61)$$

and verifying that  $d = 0$  is a local minimum of that problem, as described in [78, 4]. We do this by formulating the LPCC as an mixed-integer linear program within a trust region which we solve using Gurobi. The results of this B-stationarity verification is shown in Figure 42b.

We note that the vast majority of the converged to points satisfy B-stationarity. However we do also note that in many of the more interesting problems, those being time-freezing and CLS problems, we sometimes fail to verify B-stationarity even in cases where from analysis of the solution we clearly see the found solution is a optimal solution. In some cases solving those problems to a higher accuracy, i.e. a lower complementarity tolerance, allows us to verify the B-stationarity of the solution. This suggests that the sequence of solutions will indeed converge to a B-stationary point, but it requires more steps to do so.

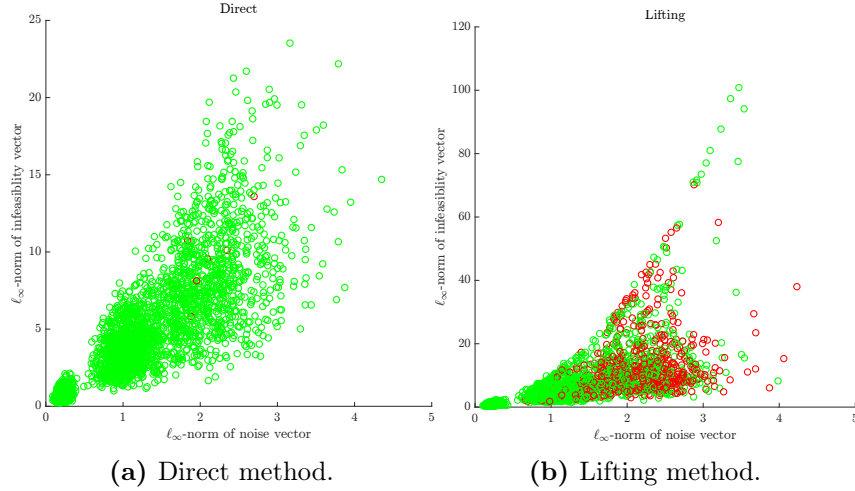


**Figure 43:** Evaluating Heaviside Step vs Stewart Reformulations.

## 5.8 Heaviside Step vs Stewart

This experiment is run on a subset of the NOSBENCH test set that contains all problems which have the same parameters differing only by the reformulation used for the Filippov system. This set tends to contain only easier problems as there are no problems that contain state jumps. In Figure 43 we see that there is very little difference between the Heaviside Step and Stewart reformulations when it comes to the performance of solver.

We draw the readers attention to the fact that while for the standard relaxation it seems that the Heaviside step wins more than 60% of the problems but the difference



**Figure 44:** Evaluating direct and lifting modes, red marks represent failures to converge.

is within a factor of 1.5 to the Stewart reformulation which we do not view as significant. This is backed up in our numerical experience, however this also speaks to the fact that the majority of the systems we treat in NOSBENCH have few regions and lack the structure that would be beneficial to take advantage of for the Heaviside step reformulation as they do not permit very sparse  $S$  matrices.

## 5.9 Lifting vs Direct

Finally we evaluate the lifting approach against direct approaches. In order to do this we took 3 simulation problems from the NOSBENCH test set with known good solutions  $w^*$  and attempted to solve the problem from an perturbed initial point  $\bar{w}$ .  $\bar{w}$  is generated via an additive Gaussian random perturbation with three different standard deviations: 0.1, 1 and 5. We attempt to solve 100 of these randomly perturbed problems with each standard deviation and each simulation problem and plot the results in Figure 44. We see pretty clearly that the lifting method performs more poorly with a smaller perturbation, which matches the behavior described in [18].





## 6 Conclusions & Outlook

In this chapter we briefly review the work done in the production of this thesis and the results contained in prior chapters. We conclude the thesis with some possible continuations of this work, as well as directions for further work on MPCC solvers.

### 6.1 Conclusions

In this thesis we introduce a new benchmark suite of MPCCs and use it to evaluate a popular and generally successful class of algorithms. This is done in order to further the goals of developing a fast and reliable solver for problems that arise from nonsmooth optimal control. We compared a variety of relaxation approaches to solving MPCCs and verified that of these the Scholtes relaxation is the best approach, and that both standard homotopies and penalty based approaches to driving the relaxation perform similarly.

We also introduce a theorem which suggests the degeneracy of the sparse formulation (cross complementarity mode 1) of the Finite Elements with Switch Detection (FESD) discretization. This is followed up with an empirical test of the performance of the cross complementarity modes of the FESD discretization which back up the theoretical assertions with worse practical performance for the sparse formulation. This has led to a change in the NOSNOC software package towards defaulting to the most dense cross complementarity mode 7, and, in future, a complete phase out of

cross complementarity mode 1 as an option. The other experiments that were run in the course of this thesis in order to evaluate the performance of various solver settings in the NOSNOC package revealed that the existing parameters are at least acceptable, or even optimal for the current set of problems that it has been used to solve. Finally, the parity of the two reformulations of Filippov systems, the Stewart and Heaviside step reformulations, was empirically confirmed.

As it stands we observe generally passable if not real-time performance on simulation problems, which is in contrast to the performance on anything but simple optimal control problems which cause even the most robust methods to perform unacceptably. This points to the fact that further work must be done to allow FESD and NOSNOC to be used for more interesting applications.

### 6.2 Future Work

The existing solution methods for MPCCs are clearly not sufficient for the purposes of applying FESD to real time problems. As such there is certainly open directions for research into better, possibly tailored solvers for FESD problems. In particular we note that solvers that take advantage of the combinatorial nature of MPCCs are an interesting avenue of research that is currently being pursued by various groups, and would be interesting to apply to the problems produced by FESD.

We also note that there are likely structures specific to FESD problems that are likely able to be exploited for further improved solvers. In particular the FESD active set can be used to map to the complementarity active set which may yield more combinatorially tractable problems.

Beyond the development of novel solvers there is some further improvements to be made to the NOSBENCH problem set, including the addition of an AMPL interface to ease it's use for more researchers in the field. We also would like to continue to

extend the problem set to include more applications of nonsmooth optimal control to better cover the whole range of possible problems. In particular an extension of the Time-Freezing reformulation that allows the use of the Stewart reformulation would allow us to evaluate that approach on the more difficult OCPs.



## 7 Acknowledgments

I would like to thank my Advisor, Armin Nurkanović, for all of his help and support throughout the time I spent working on this thesis. It has been invaluable. I will always appreciate the opportunity that he and Prof. Dr. Moritz Diehl have given me to do by far the most interesting work I have yet had the pleasure to do.

I would also like to thank Jonathan Frey, with whom I have closely collaborated with on several projects now and has been an excellent resource to help me navigate my way through the process, and provided excellent feedback on my thesis.

Furthermore I would like to thank the Gregory Kallaur, whose mentorship and support during the beginning of my academic journey put me on the path to working on this thesis. Thank you for all that time you spent nurturing a passion for learning and problem solving, not just in me but in so many of your students.

Finally, I'd like to thank my loving wife who has been incredibly supportive of my work and has helped significantly with polishing my writing.



# Bibliography

- [1] V. Acary, O. Bonnefon, and B. Brogliato, *Nonsmooth modeling and simulation for switched circuits*, vol. 69. Springer Science & Business Media, 2010.
- [2] V. Acary, H. De Jong, and B. Brogliato, “Numerical simulation of piecewise-linear models of gene regulatory networks using complementarity systems,” *Physica D: Nonlinear Phenomena*, vol. 269, pp. 103–119, 2014.
- [3] A. Nurkanović, J. Frey, A. Pozharskiy, and M. Diehl, “Finite elements with switch detection for direct optimal control of nonsmooth systems with set-valued step functions,” *IEEE Conference on Decision and Control*, 2023.
- [4] H. Scheel and S. Scholtes, “Mathematical programs with complementarity constraints: Stationarity, optimality, and sensitivity,” *Math. Oper. Res.*, vol. 25, pp. 1–22, 2000.
- [5] M. C. Ferris and J. S. Pang, “Engineering and economic applications of complementarity problems,” *SIAM Review*, vol. 39, no. 4, pp. 669–713, 1997.
- [6] J. Bard, *Practical Bilevel Optimization: Algorithms and Applications*. Boston MA: Kluwer Academic Publishers, 1999.
- [7] J. Outrata, M. Kocvara, and J. Zowe, *Nonsmooth approach to optimization problems with equilibrium constraints: theory, applications and numerical results*, vol. 28. Springer Science & Business Media, 2013.

- [8] F. Facchinei, H. Jiang, and L. Qi, “A smoothing method for mathematical programs with equilibrium constraints,” *Mathematical Programming*, vol. 85, pp. 107–134, 1999.
- [9] A. Nurkanović and M. Diehl, “NOSNOC: A software package for numerical optimal control of nonsmooth systems,” *IEEE Control Systems Letters*, 2022.
- [10] 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.
- [11] R. Fletcher and S. Leyffer, “Solving mathematical programs with complementarity constraints as nonlinear programs,” *Optimization Methods and Software*, vol. 19, no. 1, pp. 15–40, 2004.
- [12] M. Anitescu, “On using the elastic mode in nonlinear programming approaches to mathematical programs with complementarity constraints,” *SIAM Journal on Optimization*, vol. 15, no. 4, pp. 1203–1236, 2005.
- [13] 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.
- [14] G.-H. Lin and M. Fukushima, “New relaxation method for mathematical programs with complementarity constraints,” *Journal of Optimization Theory and Applications*, vol. 118, no. 1, pp. 81–116, 2003.
- [15] V. DeMiguel, M. P. Friedlander, F. J. Nogales, and S. Scholtes, “A two-sided relaxation scheme for mathematical programs with equilibrium constraints,” *SIAM Journal on Optimization*, vol. 16, no. 2, pp. 587–609, 2005.



- [16] A. Kadrani, J.-P. Dussault, and A. Benchakroun, “A new regularization scheme for mathematical programs with complementarity constraints,” *SIAM Journal on Optimization*, vol. 20, no. 1, pp. 78–103, 2009.
- [17] C. Kanzow and A. Schwartz, “A new regularization method for mathematical programs with complementarity constraints with strong convergence properties,” *SIAM Journal on Optimization*, vol. 23, no. 2, pp. 770–798, 2013.
- [18] O. Stein, “Lifting mathematical programs with complementarity constraints,” *Mathematical programming*, vol. 131, no. 1, pp. 71–94, 2012.
- [19] A. F. Izmailov, A. Pogosyan, and M. V. Solodov, “Semismooth newton method for the lifted reformulation of mathematical programs with complementarity constraints,” *Computational Optimization and Applications*, vol. 51, no. 1, pp. 199–221, 2012.
- [20] J. Hall, A. Nurkanović, F. Messerer, and M. Diehl, “A sequential convex programming approach to solving quadratic programs and optimal control problems with linear complementarity constraints,” *IEEE Control Systems Letters*, vol. 6, pp. 536–541, 2022.
- [21] K. Lin and T. Ohtsuka, “A non-interior-point continuation method for the optimal control problem with equilibrium constraints,” *arXiv preprint arXiv:2210.10336*, 2022.
- [22] K. Hotta and A. Yoshise, “Global convergence of a class of non-interior point algorithms using chen-harker-kanzow-smale functions for nonlinear complementarity problems,” *Mathematical Programming*, vol. 86, pp. 105–133, 1999.
- [23] B. Chen and T. Harker, “A non-interior-point continuation method for linear complementarity problems,” *SIAM Journal on Matrix Analysis*, pp. 1168–1190, 1993.

- [24] C. Kirches, J. Larson, S. Leyffer, and P. Manns, “Sequential linearization method for bound-constrained mathematical programs with complementarity constraints,” *SIAM Journal on Optimization*, vol. 32, no. 1, pp. 75–99, 2022.
- [25] S. Leyffer and T. S. Munson, “A globally convergent filter method for mpecs,” *Preprint ANL/MCS-P1457-0907, Argonne National Laboratory, Mathematics and Computer Science Division*, 2007.
- [26] M. Fukushima and P. Tseng, “An implementable active-set algorithm for computing a b-stationary point of a mathematical program with linear complementarity constraints,” *SIAM Journal on Optimization*, vol. 12, no. 3, pp. 724–739, 2002.
- [27] L. Guo and Z. Deng, “A new augmented lagrangian method for mpccs—theoretical and numerical comparison with existing augmented lagrangian methods,” *Mathematics of Operations Research*, vol. 47, no. 2, pp. 1229–1246, 2022.
- [28] S. Leyffer, “Macmpec: Ampl collection of mpecs,”. Argonne National Laboratory, 2000.
- [29] “Mpeclib.” <https://github.com/GAMS-dev/gamsworld/tree/master/MPECLib>.
- [30] F. Furini, E. Traversi, P. Belotti, A. Frangioni, A. Gleixner, N. Gould, L. Liberti, A. Lodi, R. Misener, H. Mittelmann, *et al.*, “Qplib: a library of quadratic programming instances,” *Mathematical Programming Computation*, vol. 11, pp. 237–265, 2019.
- [31] F. Leibfritz, “COMpleib: Constraint matrix optimization problem library - a collection of test examples for nonlinear semidefinite programs, control system design and related problems,” tech. rep., Dept. Math., Univ. Trier, Trier, Germany, 2004.

- [32] E. de Klerk and R. Sotirov, “A new library of structured semidefinite programming instances,” *Optimization Methods & Software*, vol. 24, no. 6, pp. 959–971, 2009.
- [33] N. I. Gould, D. Orban, and P. L. Toint, “CUTEr – a constrained and unconstrained testing environment, revisited.” <http://www.cuter.rl.ac.uk/>, 2001.
- [34] N. Gould, D. Orban, and P. L. Toint, “Cutest: a constrained and unconstrained testing environment with safe threads,” *Cahier du GERAD G*, vol. 2013, p. 27, 2013.
- [35] R. Fletcher and S. Leyffer, “Numerical experience with solving mpecs as nlps,” in *Department of Mathematics and Computer Science, University of Dundee, Dundee*, Citeseer, 2002.
- [36] A. F. Izmailov, M. V. Solodov, and E. Uskov, “Global convergence of augmented lagrangian methods applied to optimization problems with degenerate constraints, including problems with complementarity constraints,” *SIAM Journal on Optimization*, vol. 22, no. 4, pp. 1579–1606, 2012.
- [37] E. D. Dolan and J. J. Moré, “Benchmarking optimization software with performance profiles,” *Math. Program.*, vol. 91, pp. 201–213, 2002.
- [38] A. Schwartz, *Mathematical programs with complementarity constraints: Theory, methods and applications*. PhD thesis, Universität Würzburg, 2011.
- [39] H. Jiang and D. Ralph, “Smooth sqp methods for mathematical programs with nonlinear complementarity constraints,” *SIAM Journal on Optimization*, vol. 10, no. 3, pp. 779–808, 2000.
- [40] B. Chen, X. Chen, and C. Kanzow, “A penalized Fischer-Burmeister NCP-function,” *Mathematical Programming*, vol. 88, pp. 211–216, 2000.

- [41] G.-H. Lin and M. Fukushima, “A modified relaxation scheme for mathematical programs with complementarity constraints,” *Annals of Operations Research*, vol. 133, no. 1, pp. 63–84, 2005.
- [42] S. Steffensen and M. Ulbrich, “A new relaxation scheme for mathematical programs with equilibrium constraints,” *SIAM Journal on Optimization*, vol. 20, no. 5, pp. 2504–2539, 2010.
- [43] C. Kanzow and A. Schwartz, “The price of inexactness: convergence properties of relaxation methods for mathematical programs with complementarity constraints revisited,” *Mathematics of Operations Research*, vol. 40, no. 2, pp. 253–275, 2015.
- [44] C. Kanzow and A. Schwartz, “Convergence properties of the inexact lin-fukushima relaxation method for mathematical programs with complementarity constraints,” *Comput. Optim. Appl.*, vol. 59, no. 1-2, pp. 249–262, 2014.
- [45] H. Benson, A. Sen, D. Shanno, and R. Vanderbei, “Interior-Point Algorithms, Penalty Methods and Equilibrium Problems,” *Computational Optimization and Applications*, vol. 34, pp. 155–182, 2006.
- [46] M. Fukushima, Z.-Q. Luo, and J.-S. Pang, “A globally convergent sequential quadratic programming algorithm for mathematical programs with linear complementarity constraints,” *Computational optimization and applications*, vol. 10, no. 1, pp. 5–34, 1998.
- [47] S. Leyffer, G. López-Calva, and J. Nocedal, “Interior methods for mathematical programs with complementarity constraints,” *SIAM Journal on Optimization*, vol. 17, no. 1, pp. 52–77, 2006.
- [48] A. Nurkanović and M. Diehl, “Continuous optimization for control of hybrid systems with hysteresis via time-freezing,” *IEEE Control Systems Letters*, 2022.

- [49] M. Bernardo, C. Budd, A. R. Champneys, and P. Kowalczyk, *Piecewise-smooth dynamical systems: theory and applications*, vol. 163. Springer Science & Business Media, 2008.
- [50] 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.
- [51] D. E. Stewart and M. Anitescu, “Optimal control of systems with discontinuous differential equations,” *Numerische Mathematik*, vol. 114, no. 4, pp. 653–695, 2010.
- [52] R. Stewart and R. Chapman, “Fast stable Kalman filter algorithms utilising the square root,” in *Proc. of Intern. Conf. on Acoustics, Speech, and Signal Processing*, pp. 1815–1818, 1990.
- [53] A. Nurkanović, M. Sperl, S. Albrecht, and M. Diehl, “Finite Elements with Switch Detection for Direct Optimal Control of Nonsmooth Systems,” 2022.
- [54] A. Nurkanović, A. Pozharskiy, J. Frey, and M. Diehl, “Finite elements with switch detection for numerical optimal control of nonsmooth dynamical systems with set-valued step functions,” *arXiv preprint arXiv:2307.03482*, 2023.
- [55] A. Filippov, *Differential Equations with Discontinuous Righthand Sides: Control Systems*, vol. 18. Springer Science & Business Media, 1988.
- [56] A. Nurkanović, J. Frey, A. Pozharskiy, and M. Diehl, “Fesd-j: Finite elements with switch detection for numerical optimal control of rigid bodies with impacts and coulomb friction,” *arXiv preprint arXiv:2305.17003*, 2023.
- [57] A. Nurkanović, S. Albrecht, B. Brogliato, and M. Diehl, “The time-freezing reformulation for numerical optimal control of complementarity lagrangian systems with state jumps,” *Automatica*, vol. 158, p. 111295, 2023.

- [58] A. Nurkanović, S. Albrecht, B. Brogliato, and M. Diehl, “The Time-Freezing Reformulation for Numerical Optimal Control of Complementarity Lagrangian Systems with State Jumps -Extended Version,” *arXiv preprint arXiv:2111.06759*, 2023.
- [59] 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.
- [60] A. Wächter and L. Biegler, “IPOPT - an Interior Point OPTimizer.” <https://projects.coin-or.org/Ipopt>, 2009.
- [61] P. Gill, W. Murray, and M. Saunders, “SNOPT: An SQP algorithm for large-scale constrained optimization,” *SIAM Review*, vol. 47, no. 1, pp. 99–131, 2005.
- [62] C. Büskens and D. Wassel, “The ESA NLP solver WORHP,” in *Modeling and Optimization in Space Engineering* (G. Fasano and J. D. Pintér, eds.), vol. 73, pp. 85–110, Springer New York, 2013.
- [63] C. Vanaret and S. Leyffer, “Uno.” <https://github.com/cvanaret/Uno>.
- [64] T. A. Howell, S. Le Cleac’h, S. Singh, P. Florence, Z. Manchester, and V. Sindhwani, “Trajectory optimization with optimization-based dynamics,” *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 6750–6757, 2022.
- [65] N. J. Kong, G. Council, and A. M. Johnson, “iLQR for piecewise-smooth hybrid dynamical systems,” in *2021 60th IEEE Conference on Decision and Control (CDC)*, pp. 5374–5381, 2021.
- [66] B. Christiansen, H. Maurer, and O. Zirn, “Optimal control of a voice-coil-motor with coulombic friction,” in *2008 47th IEEE Conference on Decision and Control*, pp. 1557–1562, 2008.

- [67] J. Carius, F. Farshidian, and M. Hutter, “Mpc-net: A first principles guided policy search,” *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 2897–2904, 2020.
- [68] O. Bröls, V. Acary, and A. Cardona, “Simultaneous enforcement of constraints at position and velocity levels in the nonsmooth generalized- $\alpha$  scheme,” *Computer Methods in Applied Mechanics and Engineering*, vol. 281, pp. 131–161, 2014.
- [69] M. Calvo, J. I. Montijano, and L. Rández, “Algorithm 968: Disode45: a matlab runge-kutta solver for piecewise smooth ivps of filippov type,” *ACM Transactions on Mathematical Software (TOMS)*, vol. 43, no. 3, pp. 1–14, 2016.
- [70] D. E. Stewart, “A numerical method for friction problems with multiple contacts,” *The ANZIAM Journal*, vol. 37, no. 3, pp. 288–308, 1996.
- [71] P. T. Piiroinen and Y. A. Kuznetsov, “An event-driven method to simulate filippov systems with accurate computing of sliding motions,” *ACM Transactions on Mathematical Software (TOMS)*, vol. 34, no. 3, pp. 1–24, 2008.
- [72] HSL, “A collection of Fortran codes for large scale scientific computation.,” 2011.
- [73] I. S. Duff and J. K. Reid, “MA27 – a set of fortran subroutines for solving sparse symmetric sets of linear equations,” 1982.
- [74] I. Duff, “MA57 — a code for the solution of sparse symmetric definite and indefinite systems,” *ACM Transactions on Mathematical Software*, vol. 30, pp. 118–144, June 2004.
- [75] J. Hogg and J. A. Scott, “HSL\_MA97 : a bit-compatible multifrontal code for sparse symmetric systems,” *Rutherford Appleton Laboratory Technical Reports*, 2011.
- [76] R. Fourer, D. M. Gay, and B. W. Kernighan, *AMPL: A modeling language for mathematical programming*. Thomson, 2nd ed., 2003.

- [77] D. M. Gay, “Writing .nl files,” tech. rep., Optimization and Uncertainty Estimation, Sandia National Laboratories, 2005.
- [78] Z. Luo, J. Pang, and D. Ralph, *Mathematical Programs with Equilibrium Constraints*. Cambridge: Cambridge University Press, 1996.



# Appendices



# A Additional NOSBENCH Description and Results

**Table 9:** Parameters for for problems which make up NOSBENCH

Problem Slug	Parameter Description
2BCLS	Initial state: $x_0 = (1, 2, 0, 0)$ $x_0 = (0.2152, 1.2152, -3.9240, -3.9240)$ $x_0 = (0.2119, 1.1741, -2.8636, 1.8424)$
3CPCLS	Cart mass: $m = \{0.5, 1.0, 2.0\}$
3CPTF	Cart mass: $m = \{0.5, 1.0, 2.0\}$
986EQ	Initial state: $x_0 = (-10^{-12}, -10^{-12}, 0)$ $x_0 = (0.0045, 0.1673, 0.4020)$
986FO	Initial state: $x_0 = (3, 0)$ $x_0 = (-2.1740, -0.3669)$ $x_0 = (-0.2028, 0.0331)$

**Table 9:** Parameters for problems which make up NOSBENCH

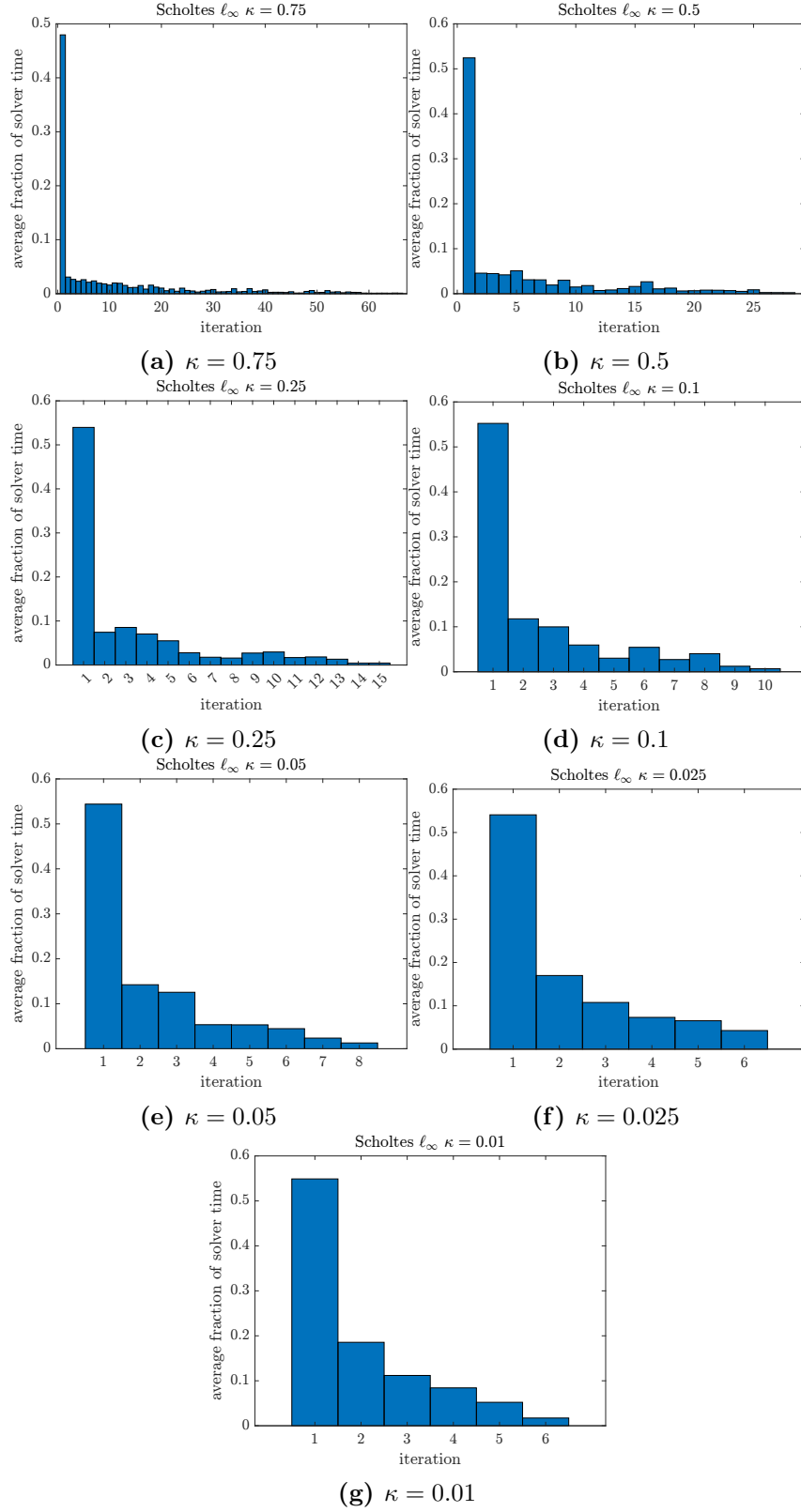
Problem Slug	Parameter Description
986FV	Initial state: $x_0 = (-2, 3, 0, 0)$ $x_0 = (-0.0075, 0.7870, 3.0307, -3.4507)$ $x_0 = (1.9337, -1.8933, 0.0055, -0.9974)$
986OM	Initial state: $x_0 = (3, 0, 0)$ $x_0 = (1.2992, 1.2595, 26.1000)$
CARHYS	Fuel cost enabled or not.
CARTIM	None
CLS1D	Initial particle height $h = \{0.03, 0.2\}$
CPWF	Friction force $F = \{2, 5\}$
DAOBCLS	Obstacle radius $r_{\text{obs}} = \{0.5, 1, 1.5\}$
DISCM	None
DRNLND	Linear or nonlinear controls.
DSCOB	Obstacle radius $r_{\text{obs}} = \{0.5, 1, 1.5\}$
DSCSP	None
FBS1S	Initial state: $x_0 = (-1.0991, 1.0902, -0.7588, -0.4115, 0.2943, 2.3575, 0.1411)$ $x_0 = (0.3491, -0.0470, -0.3207, 0.1599, -0.2828, 2.1859, 2.1176)$ $x_0 = (-1, 1, -1, -1, 1, 1, 0)$
HOPOCP	None
MFTOPT	None
MNPED	Distance of target: $d = \{2.0, 3.0, 4.0\}$
MWFOCP	None

---

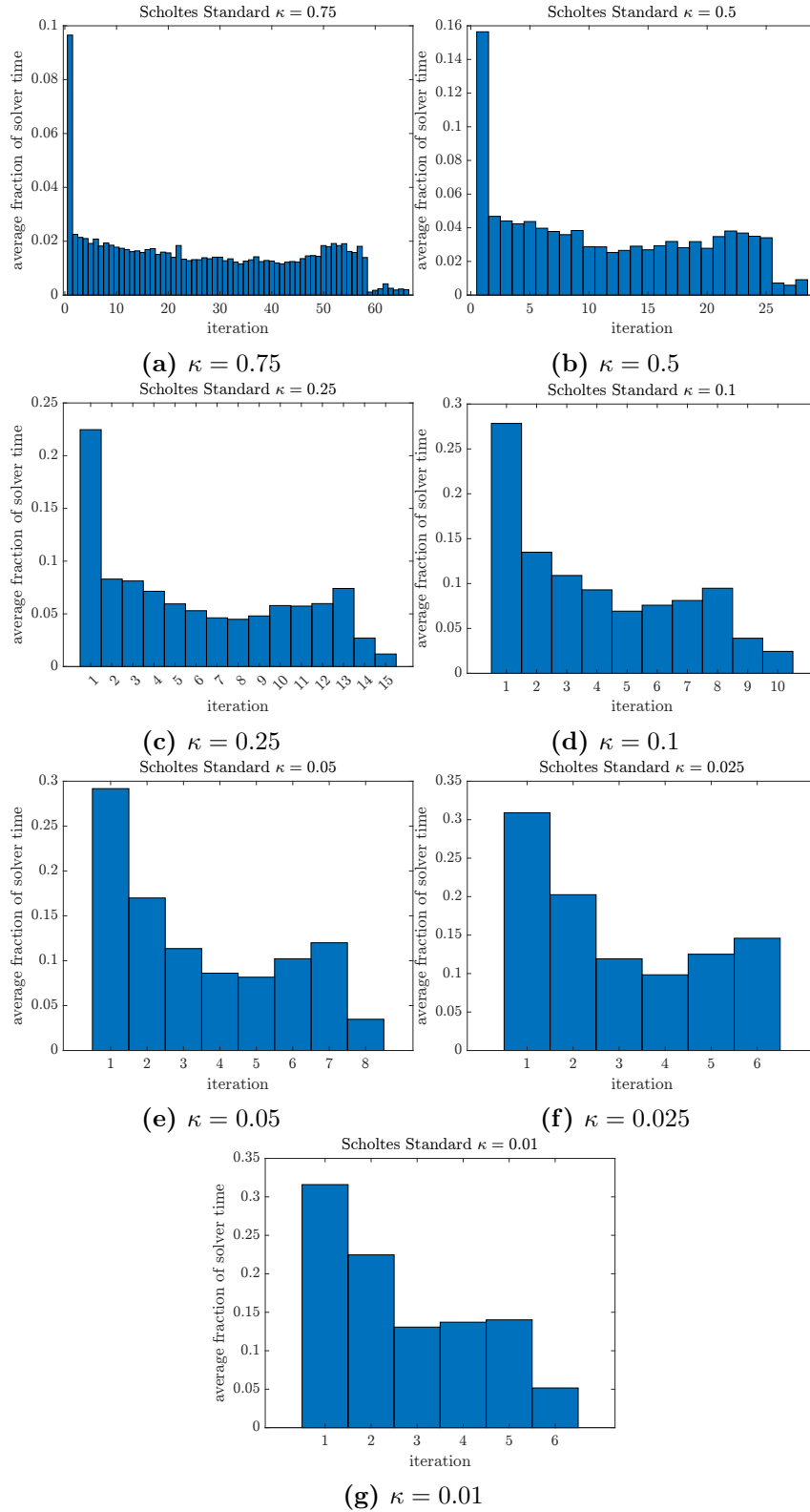
**Table 9:** Parameters for problems which make up NOSBENCH

Problem Slug	Parameter Description
OSCIL	Initial state: $x_0 = (e^{-1}, 0)$ $x_0 = (0.9633, -0.1527)$
RFB1S	Initial state: $x_0 = (0.0, -0.9388, -0.9388)$ $x_0 = (0.0003, 0.9820, -1.7315)$ $x_0 = (0, -0.001, -0.02)$
SCHUMI	3 track options and time optimal/not time optimal modes
SMCRS	None
SMLSM	None
SMOCP	Linear or nonlinear control
SMSLM	None
SMSPS	None
TFBIB	Reference rotational velocity: $\omega = \{-2\pi, -3\pi\}$
TFPOB	5 different initial states
TIMF1D	Initial particle height $h = \{0.04, 0.2\}$
TNKCSC	None



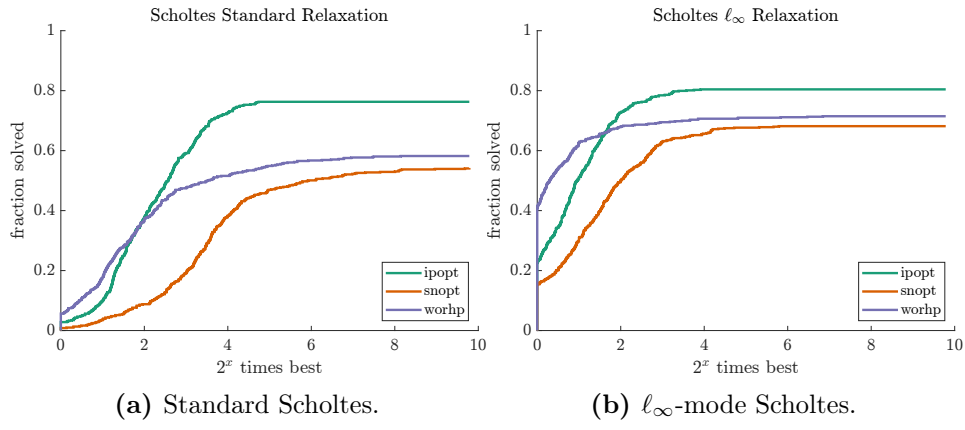


**Figure 45:**  $\ell_\infty$ -mode iteration time share.



**Figure 46:** Standard relaxation iteration time share.





**Figure 47:** Relative performance based on number of jacobian evaluations.