Nonlinear Dynamic Systems Parameterization Using Interval-Based Global Optimization: Computing Lipschitz Constants and Beyond

Sebastian A. Nugroho*, Ahmad F. Taha†, and Vu Hoang‡

Abstract—Numerous state-feedback and observer designs for nonlinear dynamic systems (NDS) have been developed in the past three decades. These designs assume that NDS nonlinearities satisfy one of the following *function set classifications*: bounded Jacobian, Lipschitz continuity, one-sided Lipschitz, quadratic inner-boundedness, and quadratic boundedness. These function sets are characterized by *constant* scalars or matrices bounding the NDS' nonlinearities. These constants *(i)* depend on the NDS' operating region, topology, and parameters, and *(ii)* are utilized to synthesize observer/controller gains. Unfortunately, there is a near-complete absence of algorithms to compute such bounding constants. In this paper, we develop analytical then computational methods to compute such constants. First, for every function set classification, we derive analytical expressions for these bounding constants through global maximization formulations. Second, we utilize a derivative-free, interval-based global maximization algorithm based on branch-and-bound framework to numerically obtain the bounding constants. Third, we showcase the effectiveness of our approaches to compute the corresponding parameters on some NDS such as highway traffic networks and synchronous generator models.

Keywords—Nonlinear systems, bounded Jacobian, Lipschitz continuous, one-sided Lipschitz, quadratic inner-boundedness, quadratic boundedness, interval-based global optimization.

I. Introduction

CONSIDER nonlinear dynamic systems (NDS) of the form

$$\dot{x}(t) = Ax(t) + Gf(x, u), \;\; y(t) = Cx(t), \qquad (1)$$

where in (1), x, u, y, and $f(\cdot)$ define the n states, m control inputs, p output measurements, and system's nonlinearities while A, G, and C are the corresponding system's matrices. The NDS model (1) represents a wide variety of systems such as transportation, water, and energy systems. In the past three decades, numerous of Lyapunov-based methods have been developed to perform state/output-feedback control and robust state estimation for NDS (1). These methods often consider or assume that nonlinear dynamics in (1) belong to one of the following function sets of nonlinear functions: *(a) bounded Jacobian*, *(b) Lipschitz continuous*, *(c) one-sided Lipschitz* (OSL), *(d) quadratic inner-boundedness* (QIB), and *(e) quadratically bounded* (QB). For example, $f(\cdot)$ is locally Lipschitz continuous in $\boldsymbol{\Omega}$ if the following holds

$$\|f(x, u) - f(\hat{x}, u)\|_2 \le \gamma_l \|x - \hat{x}\|_2,$$

for some $\gamma_l \ge 0$ and all $(x, u), (\hat{x}, u) \in \boldsymbol{\Omega}$. Tab. I lists the mathematical definitions of the other function sets.

In this paper, we investigate NDS *parameterization*. That is, computing *constants* that bound these nonlinearities—we refer to these constants as *NDS parameters* (or just simply *parameters*). For instance, the nonnegative constant γ_l defines the Lipschitz continuity property in the above example. This parameterization is crucial as hundreds of observer/controller design are in fact relying on accurate values of these constants—see Tab. I. In what follows, we perform a brief review on how these function set classifications have been utilized in some observer/controller design literature.

In [1]–[3], observer designs for Lipschitz NDS via solving linear matrix inequalities (LMIs) or semi-definite programs (SDPs) are proposed. As an alternative to tackle some limitations in Lipschitz-based observer, observers for OSL and QIB systems are developed in [4]–[6]. For bounded Jacobian systems, the corresponding observer designs are proposed in [7]–[9]. The authors in [10] use these Jacobian bounds to propose a new approach of observer design for Lipschitz nonlinear systems, which is then improved in [11] to increase scalability. As for controller and observer-based stabilization designs, the authors in [12], [13] develop a feedback stabilization framework given for QB NDS nonlinearity. Not to be confused with QIB, QB is a drastically different function set—see Tab. I For Lipschitz NDS, observer-based control framework are proposed in [14]–[16]. Various robust $\mathcal{H}_\infty$ control designs assuming OSL and QIB conditions are developed in [17]–[20].

Appendix B lists detailed formulations of several LMI- and SDP-based formulations for observer/controller designs for NDS classes provided in Tab. I. For example, considering that the nonlinear function $f(\cdot)$ in NDS (1) is Lipschitz continuous with constant $\gamma_l \ge 0$ and given that $G = I$, solving the following LMI

$$\begin{bmatrix} A^\top P + PA - YC - C^\top Y^\top + \kappa \gamma_l^2 I & P^\top \\ P & -\kappa I \end{bmatrix} \prec 0,$$

for matrix variables $P \succ 0, Y$ and $\kappa > 0$ yields an asymptotically stable estimation error through observer dynamics

$$\dot{\hat{x}}(t) = A\hat{x}(t) + f(\hat{x}, u) + L(y(t) - C\hat{x}(t)),$$

with gain $L = P^{-1}Y$. Notice that the solution to the above LMI depends on the Lipschitz constant γ_l.

From the aforementioned literature, we realize that most of these observer/controller designs are posed as SDPs that incorporate constants characterizing the nonlinearities in NDS.

*Department of Electrical Engineering and Computer Science, University of Michigan, 1301 Beal Ave., Ann Arbor, MI 48109 (snugroho@umich.edu).

†Department of Civil and Environmental Engineering, Vanderbilt University, 2201 West End Ave., Nashville, TN 37235 (ahmad.taha@vanderbilt.edu).

‡Department of Mathematics, The University of Texas at San Antonio, One UTSA Circle, San Antonio, TX 78249 (duynguyenvu.hoang@utsa.edu).

†Corresponding author.

As such, finding the tightest bounds and constants is crucial. For example, in the case for Lipschitz functions, a larger γ_l than the actual one can restrict the feasible set of the corresponding SDPs—this is commonly referred to as a *conservative* Lipschitz constant. On the contrary, if we approximate γ_l such that the value is below the actual one, there is a possibility that, due to the dynamical behavior of the system, the Lipschitz condition characterized by γ_l may be violated in certain points in $\mathbf{\Omega}$. In turn, this can lead to an unstable closed-loop system. Therefore, it is very important to obtain the tightest bounds that characterize $\boldsymbol{f}(\cdot)$ for Lipschitz function sets and beyond. This importance is not only needed in the context of state estimation/stabilization of NDS, but also in other domain—for example, to provide a robustness certificate in neural network classifier [21].

We note the following: *(i)* All of the aforementioned studies solely focus on developing methods for observer/controller designs while assuming that the NDS belongs to one of the aforementioned function sets, rather than attempting to find or design algorithms to obtain bounding constants. *(ii)* The literature mostly focuses on small-scale dynamic systems where bounding constants can be obtained analytically. In fact, we have found that although analytical computations of Lipschitz constants can be useful, such computations could return large Lipschitz constants that are way too conservative [22]. Therefore, in order to be able to use these observer designs for a high-dimensional large-scale NDS, there is a need for systematic methods that are dedicated for NDS parameterization.

To the best of our knowledge, there are no computational methods in the literature that parameterize the nonlinearities in complex, large-scale NDS—beyond Lipschitz nonlinear systems. With that in mind, some recent studies have proposed methods to parameterize some Lipschitz NDS only. The authors in [23] propose a stochastic method to approximate Lipschitz constant, albeit only applicable for functions containing single variable. In [24], a method of computing Lipschitz constant for continuously differentiable real-valued function is presented. Recently, authors in [25] propose fast numerical methods for computing Lipschitz constant using interval arithmetic and automatic differentiation. A new approach to find tight Lipschitz bounds based on SDPs for feed-forward neural networks is proposed in [26], while authors in [27] employ *kernelized density estimation* to estimate Lipschitz constant using data generated from unknown dynamical systems. The computation of Lipschitz constant for synchronous generator models is investigated in [22], where the authors utilize *low discrepancy sequence* (LDS) to under-approximate the Lipschitz constant and compare it with an analytical solution. Realize that the majority of these works only focus on parameterizing one class of nonlinearity, that is, Lipschitz continuous. For the remaining function sets in Tab. I, no methodology has been put forth.

The objective of this work is to provide a framework for NDS parameterization for large-scale NDS via systematic methods to compute the corresponding constants/parameters for each class of aforementioned nonlinearities, considering arbitrary nonlinear models having continuous partial derivatives. These constants are computed by optimally solving global maximization problems over a box-shaped convex compact set through

Table I
FUNCTION SETS OF NONLINEARITY CONSIDERED IN THIS PAPER.

Class	Mathematical Property
Bounded Jacobian	$-\infty < \underline{f}_{ij} \leq \dfrac{\partial f_i}{\partial x_j}(\boldsymbol{x}, \boldsymbol{u}) \leq \bar{f}_{ij} < +\infty,$ $(\boldsymbol{x}, \boldsymbol{u}) \in \mathbf{\Omega}, \underline{f}_{ij}, \bar{f}_{ij} \in \mathbb{R}$
Lipschitz Continuous	$\|\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) - \boldsymbol{f}(\hat{\boldsymbol{x}}, \boldsymbol{u})\|_2 \leq \gamma_l \|\boldsymbol{x} - \hat{\boldsymbol{x}}\|_2$ $(\boldsymbol{x}, \boldsymbol{u}), (\hat{\boldsymbol{x}}, \boldsymbol{u}) \in \mathbf{\Omega}, \gamma_l \in \mathbb{R}_+$
One-Sided Lipschitz	$\langle \boldsymbol{G}(\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) - \boldsymbol{f}(\hat{\boldsymbol{x}}, \boldsymbol{u})), \boldsymbol{x} - \hat{\boldsymbol{x}} \rangle \leq \gamma_s \|\boldsymbol{x} - \hat{\boldsymbol{x}}\|_2^2,$ $(\boldsymbol{x}, \boldsymbol{u}), (\hat{\boldsymbol{x}}, \boldsymbol{u}) \in \mathbf{\Omega}, \gamma_s \in \mathbb{R}$
Quadratically Inner-Bounded	$\langle \boldsymbol{G}(\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) - \boldsymbol{f}(\hat{\boldsymbol{x}}, \boldsymbol{u})), \boldsymbol{G}(\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) - \boldsymbol{f}(\hat{\boldsymbol{x}}, \boldsymbol{u})) \rangle \leq$ $\gamma_{q1} \|\boldsymbol{x} - \hat{\boldsymbol{x}}\|_2^2 + \gamma_{q2} \langle \boldsymbol{G}(\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) - \boldsymbol{f}(\hat{\boldsymbol{x}}, \boldsymbol{u})), \boldsymbol{x} - \hat{\boldsymbol{x}} \rangle,$ $(\boldsymbol{x}, \boldsymbol{u}), (\hat{\boldsymbol{x}}, \boldsymbol{u}) \in \mathbf{\Omega}, \gamma_{q1}, \gamma_{q2} \in \mathbb{R}$
Quadratically Bounded	$\langle \boldsymbol{f}(\boldsymbol{x}), \boldsymbol{f}(\boldsymbol{x}) \rangle \leq \boldsymbol{x}^\top \mathbf{\Gamma}^\top \mathbf{\Gamma} \boldsymbol{x},$ $\boldsymbol{x} \in \boldsymbol{\mathcal{X}}, \mathbf{\Gamma} \in \mathbb{R}^{h \times n}$

a global optimization algorithm based on *interval arithmetic* (IA). The paper contributions are summarized as follows:

• Proposing closed-form, analytical formulations presented as global maximization problems for computing constants that characterize bounded Jacobian, Lipschitz continuous, OSL, QIB, and QB function sets—see Tab. I. Our formulations allow NDS parameterization problems to be solved with global maximization algorithms. It is demonstrated later that some of our formulations can be scalable for high-dimensional large-scale NDS since *(a)* we only need to solve optimization problems for each unique form of nonlinearities, *(b)* it is possible to consider only a small portion of optimization variables, and *(c)* these problems can be solved via distributed computing technique.

• Utilizing a derivative-free interval-based global maximization algorithm based on branch-and-bound framework for performing NDS parameterization. The key advantage of this particular algorithm is that, since it does not necessitate the corresponding objective function to be differentiable, it can be used for performing parameterization for a wide class of NDS. Moreover, this algorithm provides an optimality guarantee in the sense that, when the algorithm terminates, it computes the *best* upper and lower bounds for the objective function. This algorithm is adopted from [28] and herein we *(a)* reformulate it for global maximization problems such that it can be directly used for NDS parameterization, *(b)* present it using rigorous mathematical notations for the ease of analysis and implementation, and *(c)* include an additional step that can aid the practical implementation of this algorithm.

• Showcasing the effectiveness and applicability of the proposed framework for NDS parameterization as well as observer designs for different kinds of NDS including traffic networks, synchronous generator, and dynamics of a moving object. Our numerical test results suggest that the proposed approach provides Lipschitz constants that are considerably less conservative than the ones obtained from analytical computations.

A preliminary version of this work appears in [29], where we *(i)* discover a novel relation between Lipschitz and QIB function sets and *(ii)* pinpoint mistakes made in the numerical section of [6], [30] and present the appropriate corrections. In addition, we also made all MATLAB codes used in our numerical experiments to be freely available online via GitHub [31]. The structure of this paper is described as follows. Section

II showcases the problem formulation. In Section III, closed-form formulations to determine the corresponding constants for all function sets through global maximization problems are all presented. Section IV presents the proposed algorithms for global optimization based on IA designed specifically for solving problems given in Section III. Thorough numerical examples are provided in Section V and the paper is concluded in Section VI.

Notation: Italicized, boldface upper and lower case characters represent matrices and column vectors: a is a scalar, $\boldsymbol{a}$ is a vector, and $\boldsymbol{A}$ is a matrix. Matrix $\boldsymbol{I}$ denotes the identity square matrix, whereas $\boldsymbol{O}$ denotes a zero matrix of appropriate dimensions. The notations $\mathbb{N}$, $\mathbb{R}$, $\mathbb{R}_+$, and $\mathbb{R}_{++}$ denote the set of natural, real, non-negative, and positive real numbers. The notations $\mathbb{R}^n$ and $\mathbb{R}^{p \times q}$ denote the sets of row vectors with n elements and matrices with size p-by-q with elements in $\mathbb{R}$. The notation $\mathbb{S}^n$ denotes the set of n-dimensional symmetric matrices. For any vector $\boldsymbol{x} \in \mathbb{R}^n$, $\|\boldsymbol{x}\|_2$ denotes the Euclidean norm of $\boldsymbol{x}$, defined as $\|\boldsymbol{x}\|_2 := \sqrt{\boldsymbol{x}^\top \boldsymbol{x}}$, where $\boldsymbol{x}^\top$ is the transpose of $\boldsymbol{x}$. For $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$, inner product is defined as $\langle \boldsymbol{x}, \boldsymbol{y} \rangle := \boldsymbol{x}^\top \boldsymbol{y}$. If $\boldsymbol{A}$ is a matrix, $\|\boldsymbol{A}\|_2$ denotes its induced 2-norm, $\|\boldsymbol{A}\|_F$ denotes its Frobenius norm, and $A_{(i,j)}$ denotes its i-th and j-th element. The set $\mathbb{I}(n)$ is defined as $\mathbb{I}(n) := \{i \in \mathbb{N} \,|\, 1 \leq i \leq n\}$, which is usually used to represent the set of indices. The notions D_x and ∇_x denote the Jacobian matrix and gradient vector with respect to vector $\boldsymbol{x}$. For a square matrix $\boldsymbol{A}$, $\lambda_{\max}(\boldsymbol{A})$ and $\lambda_{\min}(\boldsymbol{A})$ respectively return the maximum and minimum eigenvalues of $\boldsymbol{A}$. The set of all intervals containing real numbers is denoted by $\mathbb{IR}$, whereas an interval vector of dimension n is denoted by $\mathbb{IR}^n$.

II. Problem Formulation

In this paper, we consider any NDS of the following form

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{G}\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) + \boldsymbol{B}\boldsymbol{u}(t), \tag{2}$$

where vectors $\boldsymbol{x} \in \mathcal{X} \subset \mathbb{R}^n$ and $\boldsymbol{u} \in \mathcal{U} \subset \mathbb{R}^m$ represent the state and input of the system, function $\boldsymbol{f} : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^g$ captures any nonlinear phenomena, and $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{G}$ are known constant matrices of appropriate dimensions. For brevity, we drop the time dependence from now on such that $\boldsymbol{x} := \boldsymbol{x}(t)$; the same also applies to $\boldsymbol{u}$. We define $\boldsymbol{\Omega} := \mathcal{X} \times \mathcal{U}$ with $\boldsymbol{\Omega} \subset \mathbb{R}^p$ so that the domain of $\boldsymbol{f}(\cdot)$ can be simply referred to as $\boldsymbol{\Omega}$. The following assumptions are considered in this paper.

Assumption 1. *The conditions below apply to NDS* (2)

1) $\boldsymbol{\Omega} := \prod_{i \in \mathbb{I}(p)} \Omega_i$ *is a nonempty p-dimensional orthotope (may also be referred to as a hyperrectangle or box). That is, $\Omega_i = [\underline{\omega}_i, \bar{\omega}_i]$ where $\underline{\omega}_i, \bar{\omega}_i \in \mathbb{R}$ are known bounds.*

2) $\boldsymbol{f}(\cdot) \in \mathcal{C}^1$, *i.e., it is continuously differentiable in $\boldsymbol{\Omega}$.*

The first assumption dictates that each x_i and u_j belongs to convex compact sets $\mathcal{X}_i$ and $\mathcal{U}_j$ respectively. This assumption is not restrictive, seeing that cyber-physical systems have bounded states and inputs, where each upper and lower bounds define the operating regions. The second assumption ensures that the partial derivatives of $\boldsymbol{f}(\cdot)$ exist and the gradient vector $\nabla_x f_i(\cdot)$ for all $i \in \mathbb{I}(g)$ are bounded within the set $\boldsymbol{\Omega}$ [32].

In this paper, we are concerned with obtaining a way to parameterize the nonlinear function $\boldsymbol{f}(\cdot)$ for the function sets listed in Tab. I. This essentially boils down to computing the

corresponding constants for each nonlinearity—this is referred to as *parameterization*. Assumption 1 implies that $\boldsymbol{f}(\cdot)$ is Lipschitz continuous, and thus it also follows that $\boldsymbol{f}(\cdot)$ belongs to the OSL and QIB function sets [4]. As the consequence, one can utilize a plethora of observer/controller designs for NDS of the form (2). It is worthwhile mentioning that, since $(\boldsymbol{x}, \boldsymbol{u}) \in \boldsymbol{\Omega}$, these function sets are valid in the region of interest $\boldsymbol{\Omega}$.

To parameterize $\boldsymbol{f}(\cdot)$, first it is truly crucial to obtain well-defined formulations that allow the parameterization of these function sets to be performed effectively by employing optimization routines. The next section demonstrates how NDS parameterization can be posed as a global maximization problem over the set $\boldsymbol{\Omega}$.

III. Synthesis of Global Maximization Problems for NDS Parameterization

This section presents comprehensive formulations that pose the parameterization of $\boldsymbol{f}(\cdot)$ as global maximization problems in which the objective functions are all given in closed-form expressions, thereby making them applicable for the majority of deterministic global optimization algorithms.

A. Bounded Jacobian

Bounds for partial derivatives of $\boldsymbol{f}(\cdot)$ denoted by $\underline{f}_{ij}$ and $\bar{f}_{ij}$ for each $i \in \mathbb{I}(g)$ and $j \in \mathbb{I}(n)$ can be computed as

$$\bar{f}_{ij} = \max_{(\boldsymbol{x},\boldsymbol{u}) \in \boldsymbol{\Omega}} \frac{\partial f_i}{\partial x_j}(\boldsymbol{x}, \boldsymbol{u}) \tag{3a}$$

$$\underline{f}_{ij} = \min_{(\boldsymbol{x},\boldsymbol{u}) \in \boldsymbol{\Omega}} \frac{\partial f_i}{\partial x_j}(\boldsymbol{x}, \boldsymbol{u}) = -\max_{(\boldsymbol{x},\boldsymbol{u}) \in \boldsymbol{\Omega}} -\frac{\partial f_i}{\partial x_j}(\boldsymbol{x}, \boldsymbol{u}). \tag{3b}$$

Note that there is a possibility for $\frac{\partial f_i}{\partial x_j}(\cdot)$ not to include each x_j and u_j. It is possible to speed up the computation in solving (3) by reducing the search space $\boldsymbol{\Omega}$ to $\boldsymbol{\Omega}_{\boldsymbol{r}_i} := \mathcal{X}_{\boldsymbol{r}_i} \times \mathcal{U}_{\boldsymbol{r}_i'}$ where $\mathcal{X}_{\boldsymbol{r}_i} = \prod_{j \in \mathbb{I}(r_i)} \mathcal{X}_j$ and $\mathcal{U}_{\boldsymbol{r}_i'} = \prod_{j \in \mathbb{I}(r_i')} \mathcal{U}_j$ such that every $x_j \in \mathcal{X}_j$ and $u_j \in \mathcal{U}_j$ are variables included in $\frac{\partial f_i}{\partial x_j}(\cdot)$.

Remark 1. *The reduction of $\boldsymbol{\Omega}$ can also be applied for parameterizing other function sets whenever the objective function to be maximized does not include all variables from $\boldsymbol{x}$ and $\boldsymbol{u}$. This potentially allows faster computational time as well as reduces the memory size required to perform the computation.*

B. Lipschitz Continuous

Since it is assumed that Assumption 1 holds for NDS (2), $\boldsymbol{f}(\cdot)$ always satisfies the Lipschitz continuity condition. Thus, the problem here is determining the tightest Lipschitz constant for $\boldsymbol{f}(\cdot)$ that is still useful for observer/controller design. According to [33], [34], Lipschitz constant γ_l for such NDS is nothing but the maximum value of the norm of Jacobian matrix of $\boldsymbol{f}(\cdot)$ over the set $\boldsymbol{\Omega}$, that is

$$\gamma_l = \max_{(\boldsymbol{x},\boldsymbol{u}) \in \boldsymbol{\Omega}} \|\mathrm{D}_x \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u})\|_2. \tag{4}$$

Although (4) provides a straightforward method to compute γ_l, it is unfortunately difficult to obtain a closed-form expression of (4) due to the use of the 2-norm operator for the Jacobian matrix. Since we use euclidean norms $\|\cdot\|_2$ on $\mathbb{R}^n$, the value of the operator norm (see [33]) is given by

$$\|\mathrm{D}_x \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u})\|_2 = \sqrt{\lambda_{\max}\left(\mathrm{D}_x \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u})\, \mathrm{D}_x \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u})^\top\right)}.$$

3

We note that having a closed-form formulation for γ_l is important as many deterministic global optimization methods necessitate the closed-form expression of the objective function. With that in mind, the following provides a closed-form formulation to compute γ_l, which is given in Theorem 1—the following *mean value theorem* and *key lemma* are presented beforehand due to its significant role in constructing this theorem.

Lemma 1 (*Mean Value Theorem*). *Let the function $f : \mathbb{R} \to \mathbb{R}$ be continuous on $[a, b]$ and differentiable on (a, b) for $a, b \in \mathbb{R}$. Then there exists $\bar{\tau} \in (a, b)$ such that*

$$\frac{df}{dx}(\bar{\tau}) = \frac{f(b) - f(a)}{b - a}.$$

The proof of Lemma 1 is available in [32]. We emphasize that Lemma 1 holds for scalar functions f only, and not for vector-valued $\boldsymbol{f}$ having values in $\mathbb{R}^m$. In particular, for a vector valued function $\boldsymbol{f} : [a, b] \to \mathbb{R}^m$, it is not true that there exists a $a < \bar{\tau} < b$ such that

$$\boldsymbol{f}(b) - \boldsymbol{f}(a) = \boldsymbol{f}'(\bar{\tau})(b - a) \tag{5}$$

holds. This is shown on p.112 of [35]. However, it is true that a $a < \bar{\tau} < b$ exists with

$$\|\boldsymbol{f}(b) - \boldsymbol{f}(a)\|_2 \le \|\boldsymbol{f}'(\bar{\tau})\|_2 |b - a|. \tag{6}$$

(see Theorem 5.19 in [35]). In deducing bounds for Lipschitz constants we will therefore proceed carefully and prove the next Lemma first, which we call key lemma in view of its importance. It will allow us to give a uniform and streamlined derivation of our bounds for Lipschitz constants, OSL constants and constants for the QIB condition. We note that the idea of the key lemma (taking the scalar product of $\boldsymbol{f}(b) - \boldsymbol{f}(a)$ with an auxiliary vector $\boldsymbol{w}$) is inspired by [35] (see also [34], Lemma 3.1). A first consequence of the key lemma is the explicit estimate (9) for the Lipschitz constant. An alternative proof can be given by combining (4) with the following inequality

$$\|\mathrm{D}_x \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u})\|_2 \le \|\mathrm{D}_x \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u})\|_F,$$

While therefore it might be argued that (9) is a direct consequence of well-known results, the key lemma is applicable in more general situations, and can be used to derive explicit bounds for OSL (see Section III-C) and QIB constants (see Section III-D). These results are, to the best of our knowledge, fully novel.

Lemma 2 (*Key Lemma*). *Let $\boldsymbol{F} \in \mathbb{R}^{g \times g}$ be an arbitrary matrix and $\boldsymbol{w} \in \mathbb{R}^g$ and $\boldsymbol{x}, \hat{\boldsymbol{x}} \in \mathbb{R}^n, \boldsymbol{u} \in \mathbf{R}^m$ such that $(\boldsymbol{x}, \boldsymbol{u}) \in \boldsymbol{\Omega}$. Then there exists a $\bar{\tau} \in (0, 1)$ such that*

$$\langle \boldsymbol{F}(\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) - \boldsymbol{f}(\hat{\boldsymbol{x}}, \boldsymbol{u})), \boldsymbol{w} \rangle = \\ \langle \boldsymbol{\Xi}(\hat{\boldsymbol{x}} + \bar{\tau}(\boldsymbol{x} - \hat{\boldsymbol{x}}), \boldsymbol{u})(\boldsymbol{x} - \hat{\boldsymbol{x}}), \boldsymbol{w} \rangle. \tag{7a}$$

where $\boldsymbol{\Xi} := \boldsymbol{\Xi}(\boldsymbol{x}, \boldsymbol{u}) \in \mathbb{R}^{g \times n}$ is given by

$$\boldsymbol{\Xi}_{(i,j)}(\boldsymbol{x}, \boldsymbol{u}) := \sum_{k \in \mathbb{I}(g)} F_{(i,k)} \frac{\partial f_k}{\partial x_j}(\boldsymbol{x}, \boldsymbol{u}). \tag{7b}$$

Proof. Let $(\boldsymbol{x}, \boldsymbol{u}), (\hat{\boldsymbol{x}}, \boldsymbol{u}) \in \boldsymbol{\Omega}$. Define two functions $\phi : \mathbb{R} \to \mathbb{R}$ and $\boldsymbol{g} : \mathbb{R} \to \mathbb{R}^n$ as follows

$$\phi(\tau) := \langle \boldsymbol{F} \boldsymbol{f}(\boldsymbol{g}(\tau)), \boldsymbol{w} \rangle, \quad \boldsymbol{g}(\tau) := \hat{\boldsymbol{x}} + \tau(\boldsymbol{x} - \hat{\boldsymbol{x}}),$$

Note that due to the assumed convexity of $\boldsymbol{\Omega}$, $(\boldsymbol{g}(\tau), \boldsymbol{u})$ is contained in $\boldsymbol{\Omega}$ for all $\tau \in [0, 1]$. Since $\phi(\cdot)$ is continuous on $[0, 1]$ and differentiable on $(0, 1)$, then according to mean value

theorem there exists $\bar{\tau} \in (0, 1)$ such that

$$\frac{d\phi}{d\tau}(\bar{\tau}) = \frac{\phi(1) - \phi(0)}{1 - 0} = \phi(1) - \phi(0). \tag{8a}$$

The left-hand side of (8a) reads

$$\sum_{i \in \mathbb{I}(g)} \sum_{k \in \mathbb{I}(g)} F_{(i,k)} \frac{\partial f_k}{\partial x_j}(\boldsymbol{g}(\bar{\tau}), \boldsymbol{u})(x_j - \hat{x}_j)w_i,$$

yielding the conclusion of our lemma. ∎

As a first application of the key lemma, in what follows we obtain an expression for the Lipschitz constant.

Theorem 1. *The nonlinear function $\boldsymbol{f} : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^g$ in (2) is locally Lipschitz continuous in $\boldsymbol{\Omega}$ such that for any $(\boldsymbol{x}, \boldsymbol{u}), (\hat{\boldsymbol{x}}, \boldsymbol{u}) \in \boldsymbol{\Omega}$ the following condition holds*

$$\|\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) - \boldsymbol{f}(\hat{\boldsymbol{x}}, \boldsymbol{u})\|_2 \le \gamma_{l_1} \|\boldsymbol{x} - \hat{\boldsymbol{x}}\|_2, \tag{9a}$$

where γ_{l_1} can be obtained from solving the following problem

$$\gamma_{l_1} = \left(\max_{(\boldsymbol{x}, \boldsymbol{u}) \in \boldsymbol{\Omega}} \sum_{i \in \mathbb{I}(g)} \|\nabla_x f_i(\boldsymbol{x}, \boldsymbol{u})\|_2^2 \right)^{1/2}. \tag{9b}$$

Proof. Applying Lemma 2 with $\boldsymbol{F} = \boldsymbol{I}$ and $\boldsymbol{w} := \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) - \boldsymbol{f}(\hat{\boldsymbol{x}}, \boldsymbol{u})$ yields

$$\boldsymbol{\Xi}_{(i,j)}(\boldsymbol{x}, \boldsymbol{u}) := \frac{\partial f_i}{\partial x_j}(\boldsymbol{x}, \boldsymbol{u}),$$

and therefore the right-hand side of (7a) reads

$$\Psi := \sum_{i \in \mathbb{I}(g)} \sum_{j \in \mathbb{I}(n)} \frac{\partial f_i}{\partial x_j}(\hat{\boldsymbol{x}} + \bar{\tau}(\boldsymbol{x} - \hat{\boldsymbol{x}}), \boldsymbol{u})(x_j - \hat{x}_j)w_i.$$

We now proceed to estimate Ψ. By applying the Cauchy-Schwarz inequality twice (first to the sum over j and then over i), one can verify that

$$|\Psi|^2 \le \left(\sum_{i \in \mathbb{I}(g)} \|\nabla_{\boldsymbol{x}} f_i(\hat{\boldsymbol{x}} + \bar{\tau}(\boldsymbol{x} - \hat{\boldsymbol{x}}), \boldsymbol{u})\|_2 \|\boldsymbol{x} - \hat{\boldsymbol{x}}\|_2 w_i \right)^2$$

$$\le \left(\sum_{i \in \mathbb{I}(g)} \|\nabla_{\boldsymbol{x}} f_i(\hat{\boldsymbol{x}} + \bar{\tau}(\boldsymbol{x} - \hat{\boldsymbol{x}}), \boldsymbol{u})\|_2^2 \right) \|\boldsymbol{x} - \hat{\boldsymbol{x}}\|_2^2 \|\boldsymbol{w}\|_2^2$$

$$\le \left(\max_{(\boldsymbol{x}, \boldsymbol{u}) \in \boldsymbol{\Omega}} \sum_{i \in \mathbb{I}(g)} \|\nabla_{\boldsymbol{x}} f_i(\boldsymbol{x}, \boldsymbol{u})\|_2^2 \right) \|\boldsymbol{x} - \hat{\boldsymbol{x}}\|_2^2 \|\boldsymbol{w}\|_2^2.$$

Revisiting (7a), we see that we have just shown

$$|\langle f(\boldsymbol{x}, \boldsymbol{u}) - \boldsymbol{f}(\hat{\boldsymbol{x}}, \boldsymbol{u}), \boldsymbol{w} \rangle|^2 \le \gamma_{l_1}^2 \|\boldsymbol{x} - \hat{\boldsymbol{x}}\|_2^2 \|\boldsymbol{w}\|_2^2$$

Using our definition of $\boldsymbol{w} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) - \boldsymbol{f}(\hat{\boldsymbol{x}}, \boldsymbol{u})$ this reduces to

$$\|f(\boldsymbol{x}, \boldsymbol{u}) - \boldsymbol{f}(\hat{\boldsymbol{x}}, \boldsymbol{u})\|_2^4 \le \gamma_{l_1}^2 \|\boldsymbol{x} - \hat{\boldsymbol{x}}\|^2 \|\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) - \boldsymbol{f}(\hat{\boldsymbol{x}}, \boldsymbol{u})\|_2^2$$

which immediately implies (9a) provided that $\|\boldsymbol{w}\|_2^2 > 0$. In the case that $\|\boldsymbol{w}\|_2^2 = 0$, (9a) holds trivially, so this completes the proof of the theorem. ∎

Since the closed-form expression of (9b) can be determined, unlike the one provided in (4), Lipschitz constant γ_{l_1} can be computed using any deterministic global optimization algorithms. Notice that when the domain of interest $\boldsymbol{\Omega}$ is large and the function $\boldsymbol{f}(\cdot)$ is of high dimension, solving (9b) may take a significant amount of computational time. In this regard, it is possible to 'split' the maximization problem (9b) into several smaller sub-problems so that it may be solved in a distributed fashion, as described in the following corollary.

Corollary 1. *The nonlinear function $\boldsymbol{f} : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^g$ in (2) is locally Lipschitz continuous in $\boldsymbol{\Omega}$ satisfying (9a) with Lipschitz constant*

$$\gamma_{l_2} = \left(\sum_{i \in \mathbb{I}(g)} \max_{(\boldsymbol{x},\boldsymbol{u}) \in \boldsymbol{\Omega}} \|\nabla_x f_i(\boldsymbol{x},\boldsymbol{u})\|_2^2 \right)^{1/2}. \quad (10)$$

Proof. For all $(\boldsymbol{x}^*, \boldsymbol{u}^*) \in \boldsymbol{\Omega}$ and all $i \in \mathbb{I}(g)$,

$$\|\nabla_x f_i(\boldsymbol{x}^*, \boldsymbol{u}^*)\|_2^2 \leq \max_{(\boldsymbol{x},\boldsymbol{u}) \in \boldsymbol{\Omega}} \|\nabla_x f_i(\boldsymbol{x},\boldsymbol{u})\|_2^2$$

and hence

$$\sum_{i \in \mathbb{I}(g)} \|\nabla_x f_i(\boldsymbol{x}^*, \boldsymbol{u}^*)\|_2^2 \leq \sum_{i \in \mathbb{I}(g)} \max_{(\boldsymbol{x},\boldsymbol{u}) \in \boldsymbol{\Omega}} \|\nabla_x f_i(\boldsymbol{x},\boldsymbol{u})\|_2^2 \quad (11)$$

This shows that $\gamma_{l_1} \leq \gamma_{l_2}$ and hence γ_{l_2} in (10) is also a Lipschitz constant satisfying (9a). ∎

From $\gamma_{l_1} \leq \gamma_{l_2}$ we see that using γ_{l_2} as a Lipschitz constant is potentially more conservative than γ_{l_1} albeit (10) can be solved individually for each $f_i(\cdot)$ to reduce computational time. This introduces trade-offs between (9b) and (10).

C. One-Sided Lipschitz (OSL)

OSL is another function set that is a generalization of Lipschitz continuity: a Lipschitz continuous function is also OSL [4]. Realize that OSL constant γ_s can be any real number while Lipschitz constant can only be nonnegative, as described in Tab. I. The following example illustrates this point.

Example 1. *Consider the function $f : \mathbb{R} \to \mathbb{R}$ defined as*

$$f(x) = -x^3 - 100x. \quad (12)$$

This function is globally OSL with constant $\gamma_s = -100$. This can be seen from expanding the left-hand side of the OSL condition described in Tab. I

$$(f(x) - f(\hat{x}))(x - \hat{x}) = (-x^3 + \hat{x}^3 - 100(x - \hat{x}))(x - \hat{x})$$
$$\leq -100(x - \hat{x})^2,$$

which is due to $(-x^3 + \hat{x}^3)(x - \hat{x}) \leq 0$. On the other hand, an estimation of Lipschitz constant for $f(\cdot)$, say on the interval $\mathcal{X} = [-1, 1]$ would be at least equal to $\gamma_l = 103$. Thus, the sign information $\gamma_s < 0$ might give an advantage over γ_l in their application for observer design, as argued in [4]. Fig. 1 provides an illustration on function $f(\cdot)$ described in (12) with Lipschitz and OSL conditions.

In this section, we derive numerical methods to compute OSL constant γ_s for NDS of the form (2) so that it is possible to have $\gamma_s \in \mathbb{R}$. In this regard, first we propose numerical formulations that provide lower and upper bounds towards the left-hand side of OSL condition presented in Tab. I, which is summarized in the following proposition.

Proposition 1. *For the nonlinear function $\boldsymbol{f} : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^g$ in (2), there exist $\bar{\gamma}, \underline{\gamma} \in \mathbb{R}$ such that for any $(\boldsymbol{x}, \boldsymbol{u}), (\hat{\boldsymbol{x}}, \boldsymbol{u}) \in \boldsymbol{\Omega}$ the following condition holds*

$$\underline{\gamma}\|\boldsymbol{x} - \hat{\boldsymbol{x}}\|_2^2 \leq \langle \boldsymbol{G}(\boldsymbol{f}(\boldsymbol{x},\boldsymbol{u}) - \boldsymbol{f}(\hat{\boldsymbol{x}},\boldsymbol{u})), \boldsymbol{x} - \hat{\boldsymbol{x}} \rangle \leq \bar{\gamma}\|\boldsymbol{x} - \hat{\boldsymbol{x}}\|_2^2, \quad (13a)$$

where $\bar{\gamma}$ and $\underline{\gamma}$ are given as

$$\bar{\gamma} = \max_{(\boldsymbol{x},\boldsymbol{u}) \in \boldsymbol{\Omega}} \lambda_{\max} \left(\frac{1}{2} \left(\boldsymbol{\Xi}(\boldsymbol{x},\boldsymbol{u}) + \boldsymbol{\Xi}^\top(\boldsymbol{x},\boldsymbol{u}) \right) \right) \quad (13b)$$
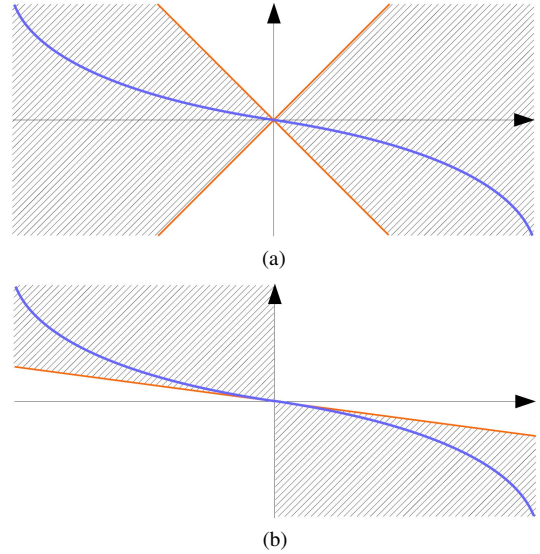

(a)

(b)

Figure 1. Since the function $f(x) = -x^3 - 100x$ (in blue) lies in the shaded area, then it satisfies two conditions: *(a)* Lipschitz continuous and *(b)* OSL. Notice that the OSL condition is characterized by a single straight line (in orange) having negative slope.

$$\underline{\gamma} = \min_{(\boldsymbol{x},\boldsymbol{u}) \in \boldsymbol{\Omega}} \lambda_{\min} \left(\frac{1}{2} \left(\boldsymbol{\Xi}(\boldsymbol{x},\boldsymbol{u}) + \boldsymbol{\Xi}^\top(\boldsymbol{x},\boldsymbol{u}) \right) \right), \quad (13c)$$

where each of the i-th and j-th element of $\boldsymbol{\Xi}(\cdot)$ is specified as

$$\boldsymbol{\Xi}_{(i,j)}(\boldsymbol{x},\boldsymbol{u}) := \sum_{k \in \mathbb{I}(g)} G_{(i,k)} \frac{\partial f_k}{\partial x_j}(\boldsymbol{x},\boldsymbol{u}). \quad (13d)$$

Proof. Applying key lemma with $\boldsymbol{w} := \boldsymbol{x} - \hat{\boldsymbol{x}}$, we get

$$\langle \boldsymbol{G}(\boldsymbol{f}(\boldsymbol{x},\boldsymbol{u}) - \boldsymbol{f}(\hat{\boldsymbol{x}},\boldsymbol{u})), (\boldsymbol{x} - \hat{\boldsymbol{x}}) \rangle$$
$$= (\boldsymbol{x} - \hat{\boldsymbol{x}})^\top \boldsymbol{\Xi}^\top(\boldsymbol{z},\boldsymbol{u})(\boldsymbol{x} - \hat{\boldsymbol{x}}) \quad (14)$$

where the matrix $\boldsymbol{\Xi}(\cdot)$ in (14) is described in (13d) and $\boldsymbol{z} = \hat{\boldsymbol{x}} + \bar{\tau}(\boldsymbol{x} - \hat{\boldsymbol{x}})$ for some $\bar{\tau} \in [0, 1]$. Now (14) is equal to

$$(\boldsymbol{x} - \hat{\boldsymbol{x}})^\top \left(\frac{1}{2} \left(\boldsymbol{\Xi}(\boldsymbol{z},\boldsymbol{u}) + \boldsymbol{\Xi}^\top(\boldsymbol{z},\boldsymbol{u}) \right) \right) (\boldsymbol{x} - \hat{\boldsymbol{x}}).$$

Since $\left(\frac{1}{2} \left(\boldsymbol{\Xi}(\boldsymbol{z},\boldsymbol{u}) + \boldsymbol{\Xi}^\top(\boldsymbol{z},\boldsymbol{u}) \right) \right)$ is symmetric, we can apply the Rayleigh quotient [36] to obtain

$$\underline{\gamma}\|\boldsymbol{x} - \hat{\boldsymbol{x}}\|_2^2 \leq \langle \boldsymbol{G}(\boldsymbol{f}(\boldsymbol{x},\boldsymbol{u}) - \boldsymbol{f}(\hat{\boldsymbol{x}},\boldsymbol{u})), \boldsymbol{x} - \hat{\boldsymbol{x}} \leq \bar{\gamma}\|\boldsymbol{x} - \hat{\boldsymbol{x}}\|_2^2,$$

which establishes (13a) whilst $\bar{\gamma}$ and $\underline{\gamma}$ are given by (13b) and (13c) respectively. ∎

From Proposition 1, the OSL constant for NDS (2) is given by $\gamma_s = \bar{\gamma}$. This result generalizes the approach to compute OSL constant in [4] in two ways. First, our result applies for a more general form of NDS expressed in (2) and secondly, we also obtain a lower bound for the left-hand side of OSL condition presented in Tab. I, which is useful for determining QIB constants, as later explained in the next section. Yet, the non closed-form expression for $\bar{\gamma}$ described in (13b) and similarly in [4] makes it difficult to compute via many deterministic global optimization methods. To that end, instead of solving (13b), it is more desirable to solve another problem that, if successfully solved, produces an upper bound towards the solution of (13b). In what follows we discuss several solutions to this problem. First, a more direct conservative way to compute γ_s is proposed below.

Theorem 2. *The nonlinear function $\boldsymbol{f} : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^g$ in*

(2) *is OSL continuous in $\boldsymbol{\Omega}$ satisfying*

$$\langle \boldsymbol{G}(\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) - \boldsymbol{f}(\hat{\boldsymbol{x}}, \boldsymbol{u})), \boldsymbol{x} - \hat{\boldsymbol{x}} \rangle \leq \gamma_s \|\boldsymbol{x} - \hat{\boldsymbol{x}}\|_2^2 \quad \text{(15a)}$$

for all $(\boldsymbol{x}, \boldsymbol{u}), (\hat{\boldsymbol{x}}, \boldsymbol{u}) \in \boldsymbol{\Omega}$ *with*

$$\gamma_s = \left(\max_{(\boldsymbol{x}, \boldsymbol{u}) \in \boldsymbol{\Omega}} \sum_{i,j \in \mathbb{I}(g)} \left| \sum_{k \in \mathbb{I}(g)} G_{(i,k)} \frac{\partial f_k}{\partial x_j}(\boldsymbol{x}, \boldsymbol{u}) \right|^2 \right)^{1/2}. \quad \text{(15b)}$$

Proof. We simply find an upper bound for $\bar{\gamma}$ given in (13b). For fixed, but arbitrary $(\boldsymbol{x}, \boldsymbol{u}) \in \boldsymbol{\Omega}$, let $\boldsymbol{v} \neq 0$ be an eigenvector for the maximal eigenvalue $\lambda_{\max}(\boldsymbol{x}, \boldsymbol{u})$ of $\frac{1}{2}(\boldsymbol{\Xi} + \boldsymbol{\Xi}^\top)$. From the eigenvalue equation and the fact that $\langle \boldsymbol{\Xi} \boldsymbol{v}, \boldsymbol{v} \rangle = \langle \boldsymbol{\Xi}^\top \boldsymbol{v}, \boldsymbol{v} \rangle$ we can deduce that $\lambda_{\max}(\boldsymbol{x}, \boldsymbol{u}) \|\boldsymbol{v}\|_2^2 = \langle \boldsymbol{\Xi}(\boldsymbol{x}, \boldsymbol{u}) \boldsymbol{v}, \boldsymbol{v} \rangle$ and hence

$$\lambda_{\max}(\boldsymbol{x}, \boldsymbol{u}) \|\boldsymbol{v}\|_2^2 \leq \|\boldsymbol{\Xi}(\boldsymbol{x}, \boldsymbol{u})\|_2 \|\boldsymbol{v}\|_2^2. \quad \text{(16)}$$

To proceed, note that (15b) holds if $\bar{\gamma} \leq 0$. Otherwise, there is some point $(\boldsymbol{x}^*, \boldsymbol{u}^*)$ at which $\bar{\gamma} = \lambda_{\max}(\boldsymbol{x}^*, \boldsymbol{u}^*) > 0$. We can square (16) and recall the definition of $\boldsymbol{\Xi}(\cdot)$ to get

$$\lambda_{\max}(\boldsymbol{x}^*, \boldsymbol{u}^*)^2 \leq \max_{(\boldsymbol{x}, \boldsymbol{u}) \in \boldsymbol{\Omega}} \sum_{i,j \in \mathbb{I}(g)} \left| \sum_{k \in \mathbb{I}(g)} G_{(i,k)} \frac{\partial f_k}{\partial x_j}(\boldsymbol{x}, \boldsymbol{u}) \right|^2,$$

and thus the desired result. ∎

Another less straightforward, that is potentially less conservative approach to calculate γ_s than (15b) is to make use one of the consequence from *Gershgorin's circle theorem*. In essence, this particular theorem guarantees that each eigenvalue of a matrix is always confined by a disk characterized by the diagonal and non-diagonal entries of that matrix [37]. The next proposition recapitulates this approach to compute an upper bound for the greatest eigenvalue of a symmetric matrix.

Proposition 2. *For any $\boldsymbol{\Psi} \in \mathbb{S}^n$, the following inequality holds*

$$\lambda_{\max}(\boldsymbol{\Psi}) \leq \max_{i \in \mathbb{I}(n)} \left(\Psi_{(i,i)} + \sum_{j \in \mathbb{I}(n) \setminus i} |\Psi_{(i,j)}| \right). \quad \text{(17)}$$

Proof. Suppose $\lambda \in \mathbb{R}$ be an eigenvalue of $\boldsymbol{\Psi} \in \mathbb{S}^n$ and $\boldsymbol{v} \in \mathbb{R}^n$ be the corresponding eigenvector. Since $\boldsymbol{v} \neq \boldsymbol{0}$, then there exists v_i such that $|v_j| \leq |v_i|$ for all $j \in \mathbb{I}(n) \setminus i$. We then perform the following transformation: $\boldsymbol{v} \leftarrow \text{sgn}(v_i) \frac{\boldsymbol{v}}{|v_i|}$. Here, $\text{sgn}(\cdot)$ denotes the sign function. Since the new $\boldsymbol{v}$ also satisfies $(\boldsymbol{\Psi} - \lambda \boldsymbol{I})\boldsymbol{v} = \boldsymbol{0}$, then at the i-th row we have

$$\lambda v_i = \sum_{j \in \mathbb{I}(n)} \Psi_{(i,i)} v_j = \Psi_{(i,i)} v_i + \sum_{j \in \mathbb{I}(n) \setminus i} \Psi_{(i,j)} v_j. \quad \text{(18a)}$$

Observe that the prior transformation leads to $v_i = 1$ and $|v_j| \leq 1$ for all $j \in \mathbb{I}(n) \setminus i$. Therefore, from (18a) and applying the triangle inequality, we obtain

$$\lambda - \Psi_{(i,i)} \leq \sum_{j \in \mathbb{I}(n) \setminus i} |\Psi_{(i,j)}| |v_j| \leq \sum_{j \in \mathbb{I}(n) \setminus i} |\Psi_{(i,j)}|. \quad \text{(18b)}$$

As (18b) applies to any eigenvalue of $\boldsymbol{\Psi}$, then (17) holds. ∎

Proposition 2 provides an amenable way which can be used for computing OSL constant γ_s provided that $\boldsymbol{\Psi} = \frac{1}{2}(\boldsymbol{\Xi}(\boldsymbol{x}, \boldsymbol{u}) + \boldsymbol{\Xi}^\top(\boldsymbol{x}, \boldsymbol{u}))$. That is,

$$\gamma_s = \max_{i \in \mathbb{I}(n)} \left(\max_{(\boldsymbol{x}, \boldsymbol{u}) \in \boldsymbol{\Omega}} \left(\Psi_{(i,i)} + \sum_{j \in \mathbb{I}(n) \setminus i} |\Psi_{(i,j)}| \right) \right).$$

Note that in (17) it is possible for $\lambda_{\max}(\boldsymbol{\Psi})$ to be nonpositive assuming that $\boldsymbol{\Psi}$ is diagonally dominant with $\Psi_{(i,i)} \leq 0$ for each $i \in \mathbb{I}(n)$. In the following theorem, we develop an alternative method, aside from (17), to provide an upper bound for the greatest eigenvalue of any symmetric matrix.

Theorem 3. *It holds for any $\boldsymbol{\Psi} \in \mathbb{S}^n$ that*

$$\lambda_{\max}(\boldsymbol{\Psi}) \leq \max_{i \in \mathbb{I}(n)} \left(\Psi_{(i,i)} + \zeta_n \max_{j \in \mathbb{I}(n) \setminus i} |\Psi_{(i,j)}| \right), \quad \text{(19a)}$$

where $\zeta_n \in \mathbb{R}_{++}$ is a scalar that depends on the dimension n and the optimal value of the following maximization problem

$$\zeta_n = \max_{\boldsymbol{v}} \frac{1}{v_i} - 1 \quad \text{(19b)}$$

$$\text{subject to} \quad \sum_{j \in \mathbb{I}(n)} |v_j| = 1, \ \boldsymbol{v} \in \mathbb{R}^n \quad \text{(19c)}$$

$$v_i > 0, \ |v_j| \leq v_i, \ \forall j \in \mathbb{I}(n) \setminus i. \quad \text{(19d)}$$

Proof. Suppose that $\lambda \in \mathbb{R}$ is an eigenvalue of $\boldsymbol{\Psi} \in \mathbb{S}^n$ and $\boldsymbol{v} \in \mathbb{R}^n$ is the corresponding eigenvector. Since $\boldsymbol{v} \neq \boldsymbol{0}$, there exists v_i such that $|v_j| \leq |v_i|$ for every $j \in \mathbb{I}(n) \setminus i$. Next, the following transformation is applied to $\boldsymbol{v}$

$$\boldsymbol{v} \leftarrow \text{sgn}(v_i) \frac{\boldsymbol{v}}{\sum_{j \in \mathbb{I}(n)} |v_j|}, \quad \text{(20a)}$$

where $\sum_{j \in \mathbb{I}(n)} |v_j| = 1$. As $(\boldsymbol{\Psi} - \lambda \boldsymbol{I})\boldsymbol{v} = \boldsymbol{0}$ for such $\boldsymbol{v}$ described in (20a), then at the i-th row we have

$$\lambda v_i = \sum_{j \in \mathbb{I}(n)} \Psi_{(i,i)} v_j = \Psi_{(i,i)} v_i + \sum_{j \in \mathbb{I}(n) \setminus i} \Psi_{(i,j)} v_j. \quad \text{(20b)}$$

Notice that the transformation (20a) yields $v_i > 0$ and $|v_j| \leq v_i$ for all $j \in \mathbb{I}(n) \setminus i$. Thus, from (20b) and applying the triangle inequality, we obtain

$$\lambda - \Psi_{(i,i)} \leq \frac{1}{v_i} \left(\sum_{j \in \mathbb{I}(n) \setminus i} |\Psi_{(i,j)}| |v_j| \right)$$

$$\leq \frac{1}{v_i} \left(\max_{j \in \mathbb{I}(n) \setminus i} |\Psi_{(i,j)}| \sum_{j \in \mathbb{I}(n) \setminus i} |v_j| \right)$$

$$= \frac{\sum_{j \in \mathbb{I}(n)} |v_j| - v_i}{v_i} \max_{j \in \mathbb{I}(n) \setminus i} |\Psi_{(i,j)}|$$

$$= \left(\frac{1}{v_i} - 1 \right) \max_{j \in \mathbb{I}(n) \setminus i} |\Psi_{(i,j)}|. \quad \text{(20c)}$$

Finally, since (20c) is valid for every eigenvalue of $\boldsymbol{\Psi}$, then we eventually obtain (19a) where $\zeta_n = \frac{1}{v_i} - 1$ is computed through solving optimization problem specified in (19b)-(19d). ∎

Note that finding ζ_n is a difficult task on its own and the reasons are as follows. Albeit $\frac{1}{v_i}$ appearing in (19b) is a convex function for $v_i \in (0, \infty)$, maximizing it yields a nonconvex optimization problem. Another difficulty comes from the equality constraint involving absolute values (19c). To that end, we provide an approach to compute ζ_n via a convex optimization problem—summarized in the following proposition.

Proposition 3. *The optimal value of the problem described in* (19b)-(19d) *is given as $\zeta_n^* = \frac{1}{v_i^*} - 1$ where v_i^* is the optimal value of the following convex optimization problem*

$$v_i^* = \min_{\boldsymbol{v}, \boldsymbol{w} \in \mathbb{R}^n} v_i \quad \text{(21a)}$$

$$\text{subject to} \quad \sum_{j \in \mathbb{I}(n)} w_j = 1, \quad \text{(21b)}$$

$$v_i > 0, \ w_j \leq v_i, \ \forall j \in \mathbb{I}(n) \setminus i. \quad \text{(21c)}$$

$$|v_i| = w_i, \ \forall i \in \mathbb{I}(n). \quad \text{(21d)}$$

Proof. First, since $v_i > 0$, then maximizing $\frac{1}{v_i}$ pushes v_i to be as close as possible to zero. Hence, it is equivalent to minimizing v_i given that $v_i > 0$. Second, we introduce a new optimization variable $\boldsymbol{w} \in \mathbb{R}^n$ such that $w_i := |v_i|, \ \forall i \in \mathbb{I}(n)$. As such, the constraints (19c) and (19d) can be transformed into (21b) and (21c), respectively. This results in a convex optimization problem (21) and ζ_n^* is computed as $\zeta_n^* = \frac{1}{v_n^*} - 1$. ∎

In summary, this section proposes three different approaches to compute OSL constant, all of which are based on computing an upper bound for the maximum eigenvalue of matrix $\boldsymbol{\Xi}(\cdot)$. Define $\gamma_{s_1}, \gamma_{s_2}$, and γ_{s_3} as OSL constants obtained from (15b), (17), and (19a) respectively. Since Theorem 2 gives a nonnegative upper bound towards maximum eigenvalue, then we have $\gamma_{s_2} \leq \gamma_{s_1}$ and $\gamma_{s_3} \leq \gamma_{s_1}$. However, further investigation is required to determine the relation between γ_{s_2} and γ_{s_3}.

D. Quadratic Inner-Boundedness (QIB)

The concept of QIB has been extensively used alongside OSL condition for a wide variety of observer design with applications ranging from state estimation to feedback stabilization [4], [6], [17]–[20]. Analogously to OSL, the constants γ_{q1} and γ_{q2} for QIB given in Tab. I may also be any real numbers. The following theorem summarizes our result for computing QIB constants.

Theorem 4. *The nonlinear function $\boldsymbol{f} : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^g$ in (2) is locally QIB in $\boldsymbol{\Omega}$ such that for any $(\boldsymbol{x}, \boldsymbol{u}), (\hat{\boldsymbol{x}}, \boldsymbol{u}) \in \boldsymbol{\Omega}$ the following holds*

$$\langle \boldsymbol{G}(\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) - \boldsymbol{f}(\hat{\boldsymbol{x}}, \boldsymbol{u})), \boldsymbol{G}(\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) - \boldsymbol{f}(\hat{\boldsymbol{x}}, \boldsymbol{u})) \rangle \leq$$
$$\gamma_{q1}\|\boldsymbol{x} - \hat{\boldsymbol{x}}\|_2^2 + \gamma_{q2}\langle \boldsymbol{G}(\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) - \boldsymbol{f}(\hat{\boldsymbol{x}}, \boldsymbol{u})), \boldsymbol{x} - \hat{\boldsymbol{x}} \rangle, \quad (22a)$$

where for $\epsilon_1, \epsilon_2 \in \mathbb{R}_+$, $\gamma_{q2} = \epsilon_2 - \epsilon_1$ and γ_{q1} is specified as

$$\gamma_{q1} = \epsilon_1 \bar{\gamma} - \epsilon_2 \underline{\gamma} + \max_{(\boldsymbol{x}, \boldsymbol{u}) \in \boldsymbol{\Omega}} \sum_{i \in \mathbb{I}(n)} \|\nabla_x \xi_i(\boldsymbol{x}, \boldsymbol{u})\|_2^2, \quad (22b)$$

where $\xi_i(\boldsymbol{x}, \boldsymbol{u}) := \sum_{j \in \mathbb{I}(g)} G_{(i,j)} f_j(\boldsymbol{x}, \boldsymbol{u})$ for $i \in \mathbb{I}(n)$ and $\bar{\gamma}$ and $\underline{\gamma}$ are the optimal values of (13b) and (13c).

Proof. Applying again Lemma 2 with $\boldsymbol{w} = \boldsymbol{G}(\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) - \boldsymbol{f}(\hat{\boldsymbol{x}}, \boldsymbol{u}))$, we can deduce the following result

$$\langle \boldsymbol{G}(\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) - \boldsymbol{f}(\hat{\boldsymbol{x}}, \boldsymbol{u})), \boldsymbol{G}(\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) - \boldsymbol{f}(\hat{\boldsymbol{x}}, \boldsymbol{u})) \rangle \leq$$
$$\left(\max_{(\boldsymbol{x}, \boldsymbol{u}) \in \boldsymbol{\Omega}} \sum_{i \in \mathbb{I}(n)} \|\nabla_x \xi_i(\boldsymbol{x}, \boldsymbol{u})\|_2^2 \right) \|\boldsymbol{x} - \hat{\boldsymbol{x}}\|_2^2. \quad (23a)$$

From (13a), we get for $\epsilon_1, \epsilon_2 \in \mathbb{R}_+$

$$0 \leq \epsilon_1 \bar{\gamma} \|\boldsymbol{x} - \hat{\boldsymbol{x}}\|_2^2 - \epsilon_1 \langle \boldsymbol{G}(\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) - \boldsymbol{f}(\hat{\boldsymbol{x}}, \boldsymbol{u})), \boldsymbol{x} - \hat{\boldsymbol{x}} \rangle \quad (23b)$$

$$0 \leq -\epsilon_2 \underline{\gamma} \|\boldsymbol{x} - \hat{\boldsymbol{x}}\|_2^2 + \epsilon_2 \langle \boldsymbol{G}(\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) - \boldsymbol{f}(\hat{\boldsymbol{x}}, \boldsymbol{u})), \boldsymbol{x} - \hat{\boldsymbol{x}} \rangle. \quad (23c)$$

Combine these two inequalities by multiplying both with (-1) and adding them to

$$-(\epsilon_2 - \epsilon_1)\langle \boldsymbol{G}(\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) - \boldsymbol{f}(\hat{\boldsymbol{x}}, \boldsymbol{u})), \boldsymbol{x} - \hat{\boldsymbol{x}} \rangle$$
$$- (\epsilon_1 \bar{\gamma} - \epsilon_2 \underline{\gamma})\|\boldsymbol{x} - \hat{\boldsymbol{x}}\|_2^2 \leq 0 \quad (23d)$$

Finally, we can deduce (22a) by using (23a), (23d) and noting that $k\|\boldsymbol{x} - \hat{\boldsymbol{x}}\|_2^2$ equals

$$k\|\boldsymbol{x} - \hat{\boldsymbol{x}}\|_2^2 + (\varepsilon_2 \underline{\gamma} - \varepsilon_1 \bar{\gamma})\langle \boldsymbol{G}(\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) - \boldsymbol{f}(\hat{\boldsymbol{x}}, \boldsymbol{u})), \boldsymbol{x} - \hat{\boldsymbol{x}} \rangle$$
$$+ (\varepsilon_1 \bar{\gamma} - \varepsilon_2 \underline{\gamma})\langle \boldsymbol{G}(\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) - \boldsymbol{f}(\hat{\boldsymbol{x}}, \boldsymbol{u})), \boldsymbol{x} - \hat{\boldsymbol{x}} \rangle$$
$$\leq \gamma_{q1}\|\boldsymbol{x} - \hat{\boldsymbol{x}}\|_2^2 + (\varepsilon_2 \underline{\gamma} - \varepsilon_1 \bar{\gamma})\langle \boldsymbol{G}(\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) - \boldsymbol{f}(\hat{\boldsymbol{x}}, \boldsymbol{u})), \boldsymbol{x} - \hat{\boldsymbol{x}} \rangle$$

with γ_{q1} given in (22b). ∎

This result allows the QIB constants γ_{q1} and γ_{q2} to be parametrized with nonnegative variables ϵ_1 and ϵ_2, hence giving one kind of degree of freedom that can be useful for observer/controller design. From (23a) and using a similar approach as in Corollary 1, one can also verify that

$$\max_{(\boldsymbol{x}, \boldsymbol{u}) \in \boldsymbol{\Omega}} \sum_{i \in \mathbb{I}(n)} \|\nabla_x \xi_i(\boldsymbol{x}, \boldsymbol{u})\|_2^2 \leq \sum_{i \in \mathbb{I}(n)} \max_{(\boldsymbol{x}, \boldsymbol{u}) \in \boldsymbol{\Omega}} \|\nabla_x \xi_i(\boldsymbol{x}, \boldsymbol{u})\|_2^2,$$

which prospectively allows γ_{q1} to be computed more efficiently in a distributive manner.

E. Quadratic Boundedness (QB)

Unlike the other function sets, QB condition is mainly utilized to generate conditions for designing stabilizing feedback control actions, with a much less common applications in observer design [38]. This condition is introduced in [12] and has been widely applied since then for numerous stabilization purposes [13]. Following [12], [13], the subsequent assumptions for $\boldsymbol{f}(\cdot)$ are considered: $\boldsymbol{f}(\cdot)$ is independent of $\boldsymbol{u}$ and $\boldsymbol{f}(0) = 0$. As such the following result is established.

Theorem 5. *The nonlinear function $\boldsymbol{f} : \mathbb{R}^n \to \mathbb{R}^g$ in (2) is locally QB in $\boldsymbol{\mathcal{X}}$ such that*

$$\langle \boldsymbol{f}(\boldsymbol{x}), \boldsymbol{f}(\boldsymbol{x}) \rangle \leq \boldsymbol{x}^\top \boldsymbol{\Gamma}^\top \boldsymbol{\Gamma} \boldsymbol{x}, \quad (24a)$$

for $\boldsymbol{x} \in \boldsymbol{\mathcal{X}}$ where $\boldsymbol{\Gamma} \in \mathbb{R}^{n \times n}$ can be computed as follow

$$\boldsymbol{\Gamma} = \mathrm{Diag}\left(\left\{ \max_{\boldsymbol{x} \in \boldsymbol{\mathcal{X}}} \sqrt{n \sum_{i \in \mathbb{I}(g)} \left(\frac{\partial f_i}{\partial x_j}(\boldsymbol{x}) \right)^2} \right\}_{j=1}^n \right). \quad (24b)$$

Proof. Let $\boldsymbol{x}, \hat{\boldsymbol{x}} \in \boldsymbol{\mathcal{X}}$ and define two functions $\phi : \mathbb{R} \to \mathbb{R}$ and $\boldsymbol{g} : \mathbb{R} \to \mathbb{R}^n$ as follows

$$\phi(\tau) := \langle \boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{f}(\hat{\boldsymbol{x}}), \boldsymbol{f}(\boldsymbol{g}(\tau)) \rangle, \quad \boldsymbol{g}(\tau) := \hat{\boldsymbol{x}} + \tau(\boldsymbol{x} - \hat{\boldsymbol{x}}).$$

Since $\phi(\cdot)$ is continuous on $[0, 1]$ and differentiable on $(0, 1)$, then from mean value theorem there exists such $\bar{\tau} \in (0, 1)$ that

$$\frac{d\phi}{d\tau}(\bar{\tau}) = \frac{\phi(1) - \phi(0)}{1 - 0} = \phi(1) - \phi(0). \quad (25a)$$

For brevity, define $\tilde{\boldsymbol{f}}(\boldsymbol{x}, \hat{\boldsymbol{x}}) := \boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{f}(\hat{\boldsymbol{x}})$. The left-hand side of (25a) is equivalent to

$$\frac{d\phi}{d\tau}(\bar{\tau}) = \sum_{i \in \mathbb{I}(g)} \tilde{f}_i(\boldsymbol{x}, \hat{\boldsymbol{x}}) \frac{df_i(\boldsymbol{g}(\bar{\tau}))}{d\tau}$$
$$= \sum_{i \in \mathbb{I}(g)} \tilde{f}_i(\boldsymbol{x}, \hat{\boldsymbol{x}}) \left(\sum_{j \in \mathbb{I}(n)} \frac{\partial f_i(\boldsymbol{g}(\bar{\tau}))}{\partial g_j} \cdot \frac{dg_j}{d\tau} \right). \quad (25b)$$

Since $\frac{dg_j}{d\tau} = x_j - \hat{x}_j$ and $\boldsymbol{g}(\bar{\tau}) \in (\hat{\boldsymbol{x}}, \boldsymbol{x})$ for which $\bar{\tau} \in (0, 1)$ implies that there exists $\boldsymbol{z} \in (\hat{\boldsymbol{x}}, \boldsymbol{x})$, (25b) is equivalent to

$$\frac{d\phi}{d\tau}(\bar{\tau}) = \sum_{i \in \mathbb{I}(g)} \tilde{f}_i(\boldsymbol{x}, \hat{\boldsymbol{x}}) \left(\sum_{j \in \mathbb{I}(n)} \frac{\partial f_i(\boldsymbol{z})}{\partial x_j}(x_j - \hat{x}_j) \right). \quad (25c)$$

By realizing that the right hand side of (25a) is

$$\phi(1) - \phi(0) = \langle \tilde{\boldsymbol{f}}(\boldsymbol{x}, \hat{\boldsymbol{x}}), \boldsymbol{f}(\boldsymbol{g}(1)) - \boldsymbol{f}(\boldsymbol{g}(0)) \rangle = \left\| \tilde{\boldsymbol{f}}(\boldsymbol{x}, \hat{\boldsymbol{x}}) \right\|_2^2,$$

then from the above, (25a), and (25c), one can verify that

$$\left\| \tilde{\boldsymbol{f}}(\boldsymbol{x}, \hat{\boldsymbol{x}}) \right\|_2^2 = \sum_{i \in \mathbb{I}(g)} \tilde{f}_i(\boldsymbol{x}, \hat{\boldsymbol{x}}) \left(\sum_{j \in \mathbb{I}(n)} \frac{\partial f_i(\boldsymbol{z})}{\partial x_j}(x_j - \hat{x}_j) \right)$$

$$\leq \sum_{i\in\mathbb{I}(g)}\left|\tilde{f}_i(\boldsymbol{x},\hat{\boldsymbol{x}})\left(\sum_{j\in\mathbb{I}(n)}\frac{\partial f_i(\boldsymbol{z})}{\partial x_j}(x_j-\hat{x}_j)\right)\right|$$

$$\leq \left\|\tilde{\boldsymbol{f}}(\boldsymbol{x},\hat{\boldsymbol{x}})\right\|_2\left(\sum_{i\in\mathbb{I}(g)}\left|\sum_{j\in\mathbb{I}(n)}\frac{\partial f_i(\boldsymbol{z})}{\partial x_j}(x_j-\hat{x}_j)\right|^2\right)^{1/2},$$

which implies that

$$\tilde{\boldsymbol{f}}(\boldsymbol{x},\hat{\boldsymbol{x}})^\top\tilde{\boldsymbol{f}}(\boldsymbol{x},\hat{\boldsymbol{x}}) \leq \sum_{i\in\mathbb{I}(g)}\left|\sum_{j\in\mathbb{I}(n)}\frac{\partial f_i(\boldsymbol{z})}{\partial x_j}(x_j-\hat{x}_j)\right|^2$$

$$\leq \sum_{i\in\mathbb{I}(g)}\left(\sum_{j\in\mathbb{I}(n)}\left|\frac{\partial f_i(\boldsymbol{z})}{\partial x_j}(x_j-\hat{x}_j)\right|\right)^2$$

$$\leq \sum_{i\in\mathbb{I}(g)}n\left(\sum_{j\in\mathbb{I}(n)}\left(\frac{\partial f_i(\boldsymbol{z})}{\partial x_j}\right)^2(x_j-\hat{x}_j)^2\right)$$

$$\leq \max_{\boldsymbol{x}\in\mathcal{X}}\sum_{i\in\mathbb{I}(g)}\sum_{j\in\mathbb{I}(n)}n\left(\frac{\partial f_i(\boldsymbol{x})}{\partial x_j}\right)^2(x_j-\hat{x}_j)^2$$

$$\leq \sum_{j\in\mathbb{I}(n)}\left(\max_{\boldsymbol{x}\in\mathcal{X}}\sum_{i\in\mathbb{I}(g)}n\left(\frac{\partial f_i(\boldsymbol{x})}{\partial x_j}\right)^2\right)(x_j-\hat{x}_j)^2$$

$$= (\boldsymbol{x}-\hat{\boldsymbol{x}})^\top\boldsymbol{\Gamma}^\top\boldsymbol{\Gamma}(\boldsymbol{x}-\hat{\boldsymbol{x}}), \tag{25d}$$

where the diagonal matrix $\boldsymbol{\Gamma}$ is described in (24b). Since $\boldsymbol{f}(0)=0$, then from (25d) one can immediately obtain (24a). ∎

As seen in (24b), the matrix $\boldsymbol{\Gamma}$ is diagonal. As such, one can indeed solve the global maximization problem for each diagonal entry of $\boldsymbol{\Gamma}$ individually in a distributive manner without trading computational time with conservativeness.

Remark 2. *(Difference between QB and QIB) Although the last two function sets—QB and QIB—seemingly share canny resemblance due to their terminology, the two sets are drastically different. Admittedly, the terminology is somewhat confusing since it suggests a connection between the two. QB is simply a bound on the values the function can attain, implying that the norm of $\boldsymbol{f}(\cdot)$ is bounded by a quadratic function. On the other hand, QIB is akin to the Lipschitz condition, since it compares the values of $\boldsymbol{f}(\cdot)$ at two different points.*

F. Remark on Improving Scalability and New Insights

So far we have presented systematic methods of transforming NDS parameterization as global maximization problems for various function sets. Herein, we *(a)* discuss some methods to improve scalability of the proposed approaches for NDS parameterization and *(b)* review the relation between Lipschitz continuous and QIB function sets and provide some necessary corrections.

We now discuss important remarks and strategies for improving the scalability when solving global optimization problems for NDS parameterization—listed as follows.

- The first strategy, as mentioned in Remark 1, can be referred to as *search space reduction*. For instance, in the context of Lipschitz parameterization where not every components of $\boldsymbol{x}$ and $\boldsymbol{u}$ appears as an argument in $\nabla_x f_i(\boldsymbol{x},\boldsymbol{u})$, one can neglect these components from $\boldsymbol{\Omega}$ such that it is sufficient to solve

$\max_{(\boldsymbol{x},\boldsymbol{u})\in\boldsymbol{\Omega}_{r_i}}\|\nabla_x f_i(\boldsymbol{x},\boldsymbol{u})\|_2^2$ where $\boldsymbol{\Omega}_{r_i}$ denotes the reduced search space.

- The second strategy exploits the mathematical structure of $\boldsymbol{f}(\cdot)$. Most of NDS representing many modern infrastructures are comprised of several individual subsystems that are intertwined together, forming complex networked NDS. The majority of these subsystems, to some extent, share similar forms of nonlinear dynamics. As an illustration, consider a network of highway traffic [39], [40]. Assuming a free-flow condition, the nonlinearity on the i-th highway segment not connected to ramps is given as $f_i(\boldsymbol{x})=\delta\left(x_i^2-x_{i-1}^2\right)$ where δ is a constant. If the $i+j$-th highway segment is also not connected to ramps for some index j, then $f_{i+j}(\boldsymbol{x})=\delta\left(x_{i+j}^2-x_{i+j-1}^2\right)$ and thus from (10), it suffices to solve $\max_{(\boldsymbol{x},\boldsymbol{u})\in\boldsymbol{\Omega}_i}\|\nabla_x f_i(\boldsymbol{x},\boldsymbol{u})\|_2^2$ once since the optimal values for i and $i+j$ are the same (assuming that $\boldsymbol{\Omega}_i=\boldsymbol{\Omega}_{i+j}$ for some index j).

- The third strategy employs a distributed computing paradigm, where in the context of computing Lipschitz constant, its computational burden is shared within several computers (or machines) as the problem $\max_{(\boldsymbol{x},\boldsymbol{u})\in\boldsymbol{\Omega}}\|\nabla_x f_i(\boldsymbol{x},\boldsymbol{u})\|_2^2$ is solved on each machine for all $i\in\mathbb{I}(g)$, thereby reducing the overall computational time.

Note that the second strategy implies that the complexity of our approaches does *not* necessarily depend on the problem dimension—instead, it relies on the number of different nonlinearities in the NDS. At last, it is always possible to combine these strategies altogether to maximize the computational efficiency.

Next, we discuss some recent results on Lipschitz continuity and QIB function sets. It is generally known that, for $\boldsymbol{G}=\boldsymbol{I}$ and $\boldsymbol{f}(\cdot)$ satisfies Lipschitz continuity condition with constant γ_l, then $\boldsymbol{f}(\cdot)$ is also QIB with $\gamma_{q1}=\gamma_l^2$ and $\gamma_{q2}=0$ [30]. It is claimed in [30], however, that the converse does not hold in general. The following statement is proven in that paper: if $\boldsymbol{f}(\cdot)$ is OSL and QIB with γ_{q2} positive, then $\boldsymbol{f}(\cdot)$ is Lipschitz continuous. Our investigation reveals that if a function $\boldsymbol{f}(\cdot)$ is QIB then it *is necessarily* Lipschitz continuous. This finding is summarized in the following theorem.

Theorem 6. *Suppose that $\boldsymbol{f}(\cdot)$ is QIB with constants $\gamma_{q1},\gamma_{q2}\in\mathbb{R}$ and $\boldsymbol{G}=\boldsymbol{I}$. Then, $\boldsymbol{f}(\cdot)$ is also Lipschitz continuous where the constants γ_{q1},γ_{q2} necessarily satisfy the inequality*

$$2\gamma_{q1}+|\gamma_{q2}|^2\geq 0. \tag{26}$$

The above result (proved in [29]) shows that QIB *implies* Lipschitz continuity. Moreover since Lipschitz continuity implies QIB, it can be concluded that both class of functions are the same. In fact, for given QIB constants γ_{q1},γ_{q2}, the Lipschitz constant is computed as $\gamma_l=\sqrt{2\gamma_{q1}+|\gamma_{q2}|^2}$. Note that the condition (26) is necessary, i.e., it holds for every QIB functions, but *not* sufficient. In addition to this, our recent work [29] also pinpoints some mistakes in the numerical section of [6], [30], where we provide the corresponding correct conditions. The next section focuses on the development of a simple interval-based algorithm to solve global maximization problems.

IV. A Derivative-Free Interval-Based Global Maximization via Branch-and-Bound Routines

After showing that the parameterization of NDS can actually be transformed into solving global maximization problems, our next step is finding a way to solve such problems. In general there are two distinct approaches that can be pursued in solving global optimization problems: *point-based method* and *interval-based method* [28]. Note that the disadvantages of the former include the lack of optimality guarantees and the possibility of such methods converging to local maxima/minima [28]. In the context of NDS parameterization, if the solution from point-based optimization algorithms is suboptimal, then the resulting parameter is an *under-approximation*. For the interval-based algorithm, however, as it provides both upper and lower bounds towards the optimal solution, the resulting upper bound can be used to compute the corresponding parameter of the NDS. Based on the above considerations, we utilize an interval-based global optimization method for parameterizing the NDS. The following sequel succinctly discusses the interval-based global optimization method considered in this paper.

A. Fundamentals of Interval Arithmetic and Interval Function

An interval $[a, b] \subset \mathbb{R}$ is formally defined as follows

$$[a, b] := \{x \in \mathbb{R} \mid a \leq x \leq b\}.$$

In IA, it is common to represent a real variable x by its interval so that $x \in [x] := [\underline{x}, \bar{x}]$. It follows directly from its definition that $\underline{x} = \inf([x])$ and $\bar{x} = \sup([x])$. An interval $[x]$ is said to be *degenerate* if it contains one element x such that $x = \underline{x} = \bar{x}$. The midpoint (or center) and width of an interval $[x]$ are respectively defined as

$$\text{mid}([x]) := \frac{1}{2}(\underline{x} + \bar{x}), \quad \text{width}([x]) := \bar{x} - \underline{x}.$$

For brevity, we define $\|[x]\| := \text{width}([x])$. A binary operation of two intervals $X := [x]$ and $Y := [y]$ is defined as follow

$$X \diamond Y := \{z \in \mathbb{R} \mid z = x \diamond y, \ x \in X, \ y \in Y\},$$

where the notation $\diamond$ represents any binary operator from the set of elementary arithmetic operators $\{+, -, \times, \div\}$. Readers are referred to [25], [41] for more comprehensive IA operations.

An n-orthotope $\mathbfcal{S} = \prod_{i \in \mathbb{I}(n)} \mathcal{S}_i$ can be regarded as an *element* of n-dimensional intervals $\mathbb{IR}^n$, i.e., $\mathbfcal{S} \in \mathbb{IR}^n$ and consequently $\mathcal{S}_i \in \mathbb{IR}$. By definition, the notation $\mathbf{z} \in \mathbfcal{S}$ means that $z_i \in \mathcal{S}_i$ for all $i \in \mathbb{I}(n)$. For two n-orthotopes $\mathbfcal{S}, \mathbfcal{S}' \in \mathbb{IR}^n$, the notion $\mathbfcal{S} \subseteq \mathbfcal{S}'$ holds if for all $i \in \mathbb{I}(n)$ we have $\mathcal{S}_i \subseteq \mathcal{S}_i'$ [42]. The interval representation of a real-valued function $f(\cdot)$ is denoted by $f^I(\cdot)$ [43]. Such function is generally referred to as *interval-valued function*. An interval function $f^I(\cdot)$, which is an interval representation of $f(\cdot)$, is termed as an *interval extension* of $f(\cdot)$—see Definition 1.

Definition 1. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a mapping with $\mathbfcal{S} \subset \mathbb{R}^n$ as the domain of interest. An interval function $f^I : \mathbb{IR}^n \to \mathbb{IR}$ is said to be an interval extension of $f(\cdot)$ if, for all $\mathbf{z} \in \mathbfcal{S}$, $f(\mathbf{z}) = f^I([\mathbf{z}, \mathbf{z}])$ where $[\mathbf{z}, \mathbf{z}] = \prod_{i \in \mathbb{I}(n)}[z_i, z_i]$.*

The construction of $f^I(\cdot)$ can be done by simply substituting each occurrence of x_i in $f(\cdot)$ with $\mathcal{S}_i$ for all $i \in \mathbb{I}(n)$ [28]. At this point, we should be aware of one particular disadvantage of IA referred to as *dependency effect*: the interval evaluation

of $f(\cdot)$ depends on the expression of $f(\cdot)$ used to implement the computation [28]. With that in mind, in using IA for solving global optimization problems, one has to carefully choose the way of expressing the interval extension of the objective value in order to obtain the best results. The next definition presents one important property of an interval extension, namely *inclusion isotonic*, which is useful for computing interval extensions.

Definition 2. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a mapping with $f^I : \mathbb{IR}^n \to \mathbb{IR}$ as its interval extension. For $\mathbfcal{S}, \mathbfcal{S}' \in \mathbb{IR}^n$ such that $\mathcal{S}_i \subseteq \mathcal{S}_i'$ for every $i \in \mathbb{I}(n)$, then $f^I(\cdot)$ is inclusion isotonic if and only if $f^I(\mathbfcal{S}) \subseteq f^I(\mathbfcal{S})'$.*

Herein, it is assumed that most computer programs represent real numbers using outward rounding [44]. This assumption allows every interval extension of any real function satisfies the inclusion isotonic property [43]. The fundamental theorem of interval arithmetic, presented below, guarantees that any interval extension of a real function satisfying the inclusion isotonicity encompasses the range of that function [43].

Lemma 3 (Fundamental Theorem of Interval Arithmetic)**.** *Let $f^I : \mathbb{IR}^n \to \mathbb{IR}$ be an interval extension of $f : \mathbb{R}^n \to \mathbb{R}$ which is inclusion isotonic. Then, for all $z_i \in \mathcal{S}_i$ where $i \in \mathbb{I}(n)$ and $\mathbfcal{S} \in \mathbb{IR}^n$, $f^I(\mathbfcal{S})$ contains the range of $f(\mathbf{z})$.*

The proof of Lemma 3 can be obtained from [43]. In the above lemma, the range of $f(\cdot)$ for a given domain $\mathbfcal{S}$ is defined as $\text{range}(f) := [\inf_{\mathbf{z} \in \mathbfcal{S}} f(\mathbf{z}), \sup_{\mathbf{z} \in \mathbfcal{S}} f(\mathbf{z})]$. If we know the domain of $f(\cdot)$, then we simply refer the upper and lower bounds of the range of $f(\cdot)$ as $\sup f(\mathbf{z})$ and $\inf f(\mathbf{z})$—the same goes for expressing the upper and lower bounds of any interval-valued function. For the sake of simplicity, we write $f(\mathbfcal{S})$ to represent the range of $f(\cdot)$, that is, $f(\mathbfcal{S}) = \text{range}(f)$. The fundamental theorem of arithmetic ensures that $f(\mathbfcal{S}) \subseteq f^I(\mathbfcal{S})$ always holds for any valid $f^I(\cdot)$. An interval extension $f^I(\cdot)$ satisfying $f^I(\mathbfcal{S}) = f(\mathbfcal{S})$ is said to be *minimal* or *exact*, and we say that its endpoints are *sharp* [28], [43]. Readers are referred to [42], [43] for more thorough basics on interval arithmetic, functions, and analysis. Next, we discuss the principle of solving global maximization problems based on IA.

B. An Interval-Based BnB Algorithm for Global Maximization

We start this section by considering the following problem

$$h := \max_{\boldsymbol{\omega} \in \boldsymbol{\Omega}} h(\boldsymbol{\omega}), \tag{27}$$

where the function $h : \mathbb{R}^p \to \mathbb{R}$ denotes the objective function to be maximized inside a p-orthotope $\boldsymbol{\Omega}$ and h denotes the objective function value. Since we do not have any constraint other than $\boldsymbol{\Omega}$, then Problem (27) is categorized as *unconstrained global maximization problem*. The above problem generalizes the way of computing the corresponding constants for each class of nonlinearity presented in Tab. I. In this paper, the notion $\boldsymbol{\omega}^*$ denotes any maximizer of (27) and $h^* = h(\boldsymbol{\omega}^*)$ denotes the optimal value of (27). For this problem to be meaningful, it is also assumed that $\boldsymbol{\omega}^* \in \boldsymbol{\Omega}$.

The term *interval-based algorithm* usually refers to any algorithm that utilizes IA for solving global optimization problem. IA has been widely utilized over past decades for solving global optimization problems, either constrained or unconstrained—see [28], [42]–[48]. Global maximization problems, in general,

Algorithm 1: Interval-Based Algorithm for Unconstrained Global Maximization

---

1 **input:** $\boldsymbol{\Omega}$, ϵ_h, ϵ_Ω, $h(\cdot)$, $h^I(\cdot)$
2 **initialize:** $\boldsymbol{\mathcal{S}}_1 = \boldsymbol{\Omega}$, $C = \{\boldsymbol{\mathcal{S}}_1\}$, $\tilde{C} = \emptyset$
3 **compute:** $\bar{\boldsymbol{\mathcal{S}}} = \arg\max_{\boldsymbol{\mathcal{S}}_i \in C} \sup\left(h^I(\boldsymbol{\mathcal{S}}_i)\right)$,
  $u = \sup\left(h^I(\bar{\boldsymbol{\mathcal{S}}})\right)$, $l = h\left(\mathrm{mid}\left(\boldsymbol{\mathcal{S}}_1\right)\right)$
4 **while** $u - l > \epsilon_h$ **and** $\left|\bar{\boldsymbol{\mathcal{S}}}\right| > \epsilon_\Omega$ **do**
5 $\quad$ $\{C, \bar{\boldsymbol{\mathcal{S}}}, l, u\} \leftarrow \mathrm{OptBnB}\left(C, \bar{\boldsymbol{\mathcal{S}}}, l\right)$
6 **if** $u - l > \epsilon_h$ **then**
7 $\quad$ **foreach** $\boldsymbol{\mathcal{S}}_j \in C$ **and** $|\boldsymbol{\mathcal{S}}_j| > \epsilon_\Omega$ **do**
8 $\quad\quad$ $\tilde{C} \leftarrow \tilde{C} \cup \{\boldsymbol{\mathcal{S}}_j\}$
9 **while** $u - l > \epsilon_h$ **and** $\tilde{C} \neq \emptyset$ **do**
10 $\quad$ $\underline{\boldsymbol{\mathcal{S}}} \leftarrow \arg\max_{\boldsymbol{\mathcal{S}}_i \in \tilde{C}} \inf\left(h^I(\boldsymbol{\mathcal{S}}_i)\right)$
11 $\quad$ $\{C, \bar{\boldsymbol{\mathcal{S}}}, l, u\} \leftarrow \mathrm{OptBnB}\left(C, \underline{\boldsymbol{\mathcal{S}}}, l\right)$, $\tilde{C} \leftarrow \emptyset$
12 $\quad$ **foreach** $\boldsymbol{\mathcal{S}}_j \in C$ **and** $|\boldsymbol{\mathcal{S}}_j| > \epsilon_\Omega$ **do**
13 $\quad\quad$ $\tilde{C} \leftarrow \tilde{C} \cup \{\boldsymbol{\mathcal{S}}_j\}$
14 **if** $u - l > \epsilon_h$ **then**
15 $\quad$ **foreach** $\boldsymbol{\mathcal{S}}_j \in C$ **do**
16 $\quad\quad$ $l \leftarrow \max\left(l, h\left(\inf\left(\boldsymbol{\mathcal{S}}_j\right)\right), h\left(\sup\left(\boldsymbol{\mathcal{S}}_j\right)\right)\right)$
17 $\quad\quad$ **foreach** $\boldsymbol{\mathcal{S}}_k \in C$ **do**
18 $\quad\quad\quad$ **if** $\sup\left(h^I(\boldsymbol{\mathcal{S}}_k)\right) < l$ **then**
19 $\quad\quad\quad\quad$ $C \leftarrow C \setminus \{\boldsymbol{\mathcal{S}}_k\}$
20 $\quad\quad$ **if** $u - l \leq \epsilon_h$ **then**
21 $\quad\quad\quad$ **break**
22 **output:** C, l, u

Algorithm 2: *OptBnB*

---

1 **input:** C, $\boldsymbol{\mathcal{S}}'$, l
2 $C \leftarrow C \setminus \{\boldsymbol{\mathcal{S}}'\}$, $\boldsymbol{\Omega}' = \arg\max_{\boldsymbol{\Omega}_i \in \boldsymbol{\mathcal{S}}'} |\boldsymbol{\Omega}_i|$
3 $\boldsymbol{\Omega}'_l = [\inf\left(\boldsymbol{\Omega}'\right), |\boldsymbol{\Omega}'|]$, $\boldsymbol{\Omega}'_r = [|\boldsymbol{\Omega}'|, \sup\left(\boldsymbol{\Omega}'\right)]$
4 $\boldsymbol{\mathcal{S}}_l = \boldsymbol{\Omega}'_1 \times \cdots \times \boldsymbol{\Omega}'_l \times \cdots \times \boldsymbol{\Omega}'_p$,
  $\boldsymbol{\mathcal{S}}_r = \boldsymbol{\Omega}'_1 \times \cdots \times \boldsymbol{\Omega}'_r \times \cdots \times \boldsymbol{\Omega}'_p$
5 $C \leftarrow C \cup \{\boldsymbol{\mathcal{S}}_l, \boldsymbol{\mathcal{S}}_r\}$
6 $\boldsymbol{\mathcal{S}}_m = \arg\max_{\boldsymbol{\mathcal{S}}_i \in C} \inf\left(h^I(\boldsymbol{\mathcal{S}}_i)\right)$
7 $l \leftarrow \max\left(l, h\left(\mathrm{mid}\left(\boldsymbol{\mathcal{S}}_m\right)\right)\right)$
8 **foreach** $\boldsymbol{\mathcal{S}}_i \in C$ **do**
9 $\quad$ **if** $\sup\left(h^I(\boldsymbol{\mathcal{S}}_i)\right) < l$ **then**
10 $\quad\quad$ $C \leftarrow C \setminus \{\boldsymbol{\mathcal{S}}_i\}$
11 $\boldsymbol{\mathcal{S}}'' = \arg\max_{\boldsymbol{\mathcal{S}}_i \in C} \sup\left(h^I(\boldsymbol{\mathcal{S}}_i)\right)$, $u = \sup\left(h^I(\boldsymbol{\mathcal{S}}'')\right)$
12 **output:** $C, \boldsymbol{\mathcal{S}}'', l, u$

---

can be categorized into *fathoming* and *localization* problems [28]. The former focuses on finding the value of the global maximum where the latter concerns with identifying the location of the global maximum. Indeed, various interval-based algorithms for global optimization have been developed since several past decades—for some references, see [28], [42]–[48]. However, as pointed out in [28], these algorithms suffer from several drawbacks: *(i)* they combine the fathoming problem with localization problem, *(ii)* they do not provide any insight on the optimality of the computed solutions, *(iii)* for localization, they do not consider inner-approximation (finding hyperrect-

angles contained within a specified distance $\delta > 0$) and outer-approximation (finding hyperrectangles covering that distance).

In addition to the aforementioned limitations, the currently available interval-based global optimization tools utilize sophisticated methods from point-based optimization with Newton's method [42], [43], [46] being the most prevalent. Note that Newton's method requires the computation of Jacobian matrix as well as the Hessian (see [46]) which requires the objective function $h(\cdot)$ to be belonged to $\mathcal{C}^2$. To have $h(\cdot) \in \mathcal{C}^2$, then it is necessary that $\boldsymbol{f}(\cdot) \in \mathcal{C}^3$, which potentially restricts the applicability of our method to parameterize NDS. With that in mind, since it is assumed that $\boldsymbol{f}(\cdot) \in \mathcal{C}^1$ as presented in Assumption 1, we decide to utilize a derivative-free interval-based global optimization algorithm developed via a branch-and-bound (BnB) framework. In particular, we adopt the Algorithm MS3 developed in [28], which is an improved variation of the Moore-Skelboe algorithm [47], [49] with stopping criteria specified by some thresholds—see Algorithm 1. Readers are referred to [28] for the detailed advantages of this algorithm relative to other contemporaries for solving global optimization problems. The following definition describes the concept of *cover* [50] that is pivotal for the algorithm.

Definition 3. *Let $\boldsymbol{\Omega}$ be a set. If $C = \{\boldsymbol{\mathcal{S}}_i\}_{i=1}^N$ is a sequence of N nonempty subsets of $\boldsymbol{\Omega}$, then we say that C is a cover in $\boldsymbol{\Omega}$. Specifically, if the following holds*

$$\boldsymbol{\Omega} = \bigcup_{i \in \mathbb{I}(N)} \boldsymbol{\mathcal{S}}_i,$$

then we say that C is a cover for $\boldsymbol{\Omega}$.

In the above definition, $N := \mathrm{card}(C)$, which denotes the number of elements in C. The objective of the interval-based algorithm is to find the best (*smallest*) interval $[l, u]$ containing h^*. If this interval is degenerate, then we have successfully found h^*. Otherwise in the case when $l < u$, we are assured that the optimal solution cannot be greater than u and smaller than l. This can be achieved by iteratively decreasing the upper bound u and increasing the lower bound l. The reduction of u can be performed by splitting a subset $\boldsymbol{\mathcal{S}} \in C$ having the biggest upper bound of interval evaluation in C. Likewise, increasing l can be done by splitting subset $\boldsymbol{\mathcal{S}} \in C$ having the smallest lower bound in C. To that end, we present an interval-based algorithm—given in Algorithm 1—for solving (27). In this algorithm, it is necessary to define two positive constants ϵ_h and ϵ_Ω: the constant ϵ_h is useful to bound the interval containing the optimal solution, whereas ϵ_Ω is useful to provide a lower bound towards the width of all subsets $\boldsymbol{\mathcal{S}}_i$ in the cover—if the width of a subset $\boldsymbol{\mathcal{S}}_i$ is smaller than ϵ_Ω, then we assume that $\boldsymbol{\mathcal{S}}_i$ cannot be split into two smaller subsets. The *width* of a subset $\boldsymbol{\mathcal{S}}_i$ is defined as $|\boldsymbol{\mathcal{S}}_i| := \max_{\boldsymbol{\Omega}_j \in \boldsymbol{\mathcal{S}}_i} |\Omega_j|$. The algorithm is said to be successful in computing the best solution when it holds that $u - l \leq \epsilon_h$—in this case, we say that h^* is ϵ_h−*optimal*. Otherwise, $u - l$ is the smallest interval containing h^* that can be obtained. After Algorithm 1 terminates, we can use u—the upper bound—to compute the corresponding constants for function sets listed in Tab. I; see Section V. It is important to note that this algorithm is a modification of Algorithm MS3 proposed in [28] with two main differences. Firstly, Algorithm 1 is specifically designed for solving global maximization problem (27) and presented

Algorithm 3: Computing Lipschitz Constants γ_{l_1} and γ_{l_2}

---

1 input: $\boldsymbol{\Omega}$, ϵ_h, ϵ_Ω, $\boldsymbol{f}(\cdot)$, *case*
2 switch *case* **do**
3   **case** 1 **do**
4   **define:** $h(\boldsymbol{x}) := \sum_{i \in \mathbb{I}(n)} \|\nabla_x f_i(\boldsymbol{x})\|_2^2$ and construct $h^I(\boldsymbol{x})$
5   **obtain:** u and l from Algorithm 1 given $\boldsymbol{\Omega}$, ϵ_h, ϵ_Ω, $h(\cdot)$, $h^I(\cdot)$
6   **output:** $\gamma_{l_1} = \sqrt{u}$
7   **case** 2 **do**
8   **foreach** $z \in \{a, b, c, d, e\}$ **do**
9   **define:** $h_z(\boldsymbol{x}) := \|\nabla_x f_z(\boldsymbol{x})\|_2^2$ and construct $h_z^I(\boldsymbol{x})$ as well as $\boldsymbol{\Omega}_z$
10   **obtain:** u_z and and l_z from Algorithm 1 given $\boldsymbol{\Omega}_z$, ϵ_h, ϵ_Ω, $h_z(\cdot)$, $h_z^I(\cdot)$
11   **output:**
        $\gamma_{l_2} = \sqrt{c_a u_a + c_b u_b + c_c u_c + c_d u_d + c_e u_e}$

---

using rigorous mathematical notations for the ease of analysis and implementation, and secondly, we have added an *explicit* additional step to ensure that the splitting of a subset $\boldsymbol{\mathcal{S}} \in C$ is possible only when the width of $\boldsymbol{\mathcal{S}}$ is greater than ϵ_Ω. The next theorem—see Appendix A for the proof—shows that Algorithm 1 computes the best interval for h^*.

Theorem 7. *When Algorithm 1 terminates, then*

1) $h^* \in [l, u]$, *where* $[l, u]$ *is the best interval for* h^*,
2) $\boldsymbol{\omega}^* \in C = \prod_{i \in \mathbb{I}(N)} \boldsymbol{\mathcal{S}}_i$ *where* C *is a cover in* $\boldsymbol{\Omega}$ *at the end of the algorithm.*

Since the lower bound l is not required for computing NDS' parameters, basically one can stop Algorithm 1 when the first loop in Step 4 terminates. However, by obtaining the best lower bound l, it gives us a sense on the location of h^* as well as an optimality certificate through the definition of ϵ_h-optimal.

Remark 3. *The lower bound l in Algorithm 1 is determined via the evaluation of objective function using the midpoint of the corresponding subset, say, $\boldsymbol{\mathcal{S}}_i$. As discussed in [28], one can also use the evaluation of either corners of $\boldsymbol{\mathcal{S}}_i$. Other than these approaches, deterministic point-based techniques can also be employed such as gradient ascent, Newton method, conjugate gradient method, and interior-point method, as well as stochastic ones such as quasi-random algorithms and heuristics.*

V. NUMERICAL EXAMPLES

In this section we showcase the proposed methodologies and algorithms for parameterizing some NDS that include traffic networks, synchronous generator, and the motion of a moving object into the following function sets: Lipschitz continuous, OSL, and QIB. All simulations are performed using MATLAB R2020a running on a 64-bit Windows 10 with 3.4GHz Intel[R] Core[TM] i7-6700 CPU and 16 GB of RAM. YALMIP's [51] optimization package together with MOSEK's [52] SDP solver are used to solve any semidefinite programs. All MATLAB codes associated in this section are available on GitHub [31].
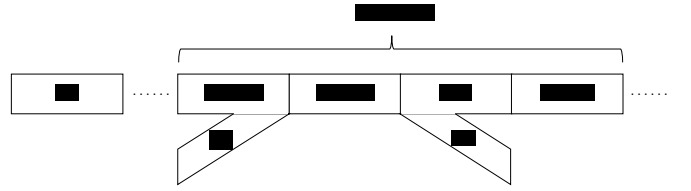

Figure 2. The structure of the highway for traffic networks parameterization. Each Section i for $i = 1, 2, \ldots, s$ consists of four mainline segments, one on-ramp, and one off-ramp.
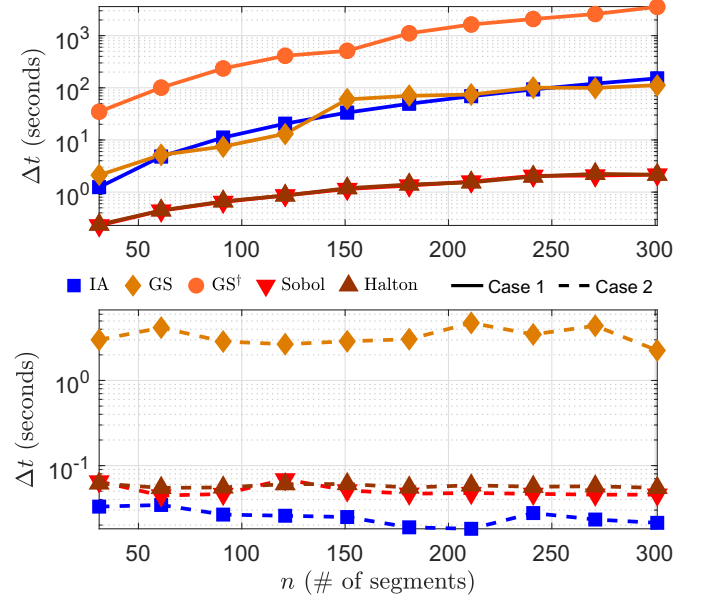

Figure 3. Total computation time on calculating Lipschitz constants for the two cases with interval-based algorithm (IA), MATLAB's global search (GS), and stochastic, quasi-random search with Sobol and Halton sequences. The results from MATLAB's global search indicated by GS† are obtained from using the 2-norm of Jacobian matrix (4).

A. Lipschitz Parameterization for Traffic Networks

The dynamic model of traffic networks, particularly on a stretched highway consisting of on- and off-ramps, can be modeled by partitioning it into several segments of equal length l such that the dynamics of each highway segment can be expressed in the form of ordinary differential equations [39]

$$\dot{\rho}_i(t) = \frac{1}{l}\left(\sum_{j \in \mathcal{Q}_i^{\text{in}}} q_j(t) - \sum_{j \in \mathcal{Q}_i^{\text{out}}} q_j(t)\right),$$

where $\mathcal{Q}_i^{\text{in}}$ and $\mathcal{Q}_i^{\text{out}}$ represent the sets of all incoming and outgoing traffic flows q_j for segment i, and ρ_i represents traffic density on that segment. The above model assumes that the highway is uncongested—in this situation, the traffic density satisfies $\rho_i \in [0, \rho_c]$ where ρ_c denotes the critical density. By employing Greenshield's fundamental diagram to represent the relation between traffic density and traffic flow [53], the nonlinearity of the overall dynamic model for this particular system (supposing that no segment is connected to both on- and off-ramps) can be distinguished into five different types of nonlinearities, detailed as follows [39], [40]

$$a) \quad f_a(\boldsymbol{x}) = \delta x_i^2 \tag{28a}$$
$$b) \quad f_b(\boldsymbol{x}) = \delta\left(x_i^2 - x_{i-1}^2 - x_j^2\right) \tag{28b}$$
$$c) \quad f_c(\boldsymbol{x}) = \delta\left(x_i^2 - x_{i-1}^2\right) \tag{28c}$$
$$d) \quad f_d(\boldsymbol{x}) = \delta\left(x_i^2 - x_{i-1}^2 + \alpha_j x_j^2\right) \tag{28d}$$

11

Table II
NUMERICAL RESULTS FOR LIPSCHITZ CONSTANTS COMPUTATION ON TRAFFIC NETWORKS. NOTATIONS: γ_{l_1} AND γ_{l_2} ARE LIPSCHITZ CONSTANTS, $\underline{\gamma}_{l_1}$ AND $\underline{\gamma}_{l_2}$ ARE THE CORRESPONDING LOWER BOUNDS, *gap* REFERS TO OPTIMALITY GAP, WHICH ESSENTIALLY COMPUTES THE DIFFERENCE BETWEEN u AND l, ϵ_h-*optimal* IS CHECKED WHEN THE SOLUTION IS ϵ_h−OPTIMAL, AND Δt IS THE TOTAL COMPUTATION TIME.

n	Case 1					Case 2					Analytical [40]
	γ_{l_1}	$\underline{\gamma}_{l_1}$	gap $(\times 10^{-5})$	ϵ_h-optimal?	Δt (s)	γ_{l_2}	$\underline{\gamma}_{l_2}$	gap $(\times 10^{-5})$	ϵ_h-optimal?	Δt (s)	γ_l
31	0.4579	0.4578	9.8531	✓	1.24	0.4579	0.4552	7.6168	✓	0.0330	0.5069
61	0.6445	0.6444	9.9494	✓	4.81	0.6445	0.6408	7.6168	✓	0.0345	0.7141
91	0.7881	0.7881	9.9974	✓	11.13	0.7881	0.7836	7.6168	✓	0.0266	0.8735
121	0.9093	0.9093	9.9975	✓	20.55	0.9093	0.9041	7.6168	✓	0.0258	1.0080
151	1.0162	1.0161	9.9736	✓	33.46	1.0162	1.0103	7.6168	✓	0.0249	1.1265
181	1.1128	1.1128	9.9976	✓	49.47	1.1128	1.1064	7.6168	✓	0.0188	1.2338
211	1.2017	1.2017	9.9976	✓	68.99	1.2017	1.1948	7.6168	✓	0.0181	1.3324
241	1.2844	1.2844	9.9977	✓	93.13	1.2844	1.2770	7.6168	✓	0.0277	1.4242
271	1.3622	1.3621	9.9857	✓	120.51	1.3622	1.3543	7.6168	✓	0.0232	1.5104
301	1.4357	1.4357	9.9977	✓	151.54	1.4357	1.4274	7.6168	✓	0.0213	1.5919

Table III
LIPSCHITZ CONSTANTS OF HIGHWAY TRAFFIC NETWORK COMPUTED WITH DIFFERENT METHODS: INTERVAL-BASED ALGORITHM (IA), POINT-BASED GLOBAL SEARCH (GS), AND POINT-BASED QUASI-RANDOM SEARCH WITH SOBOL (S) AND HALTON (H) SEQUENCES.

n	Case 1					Case 2					$\max_{\boldsymbol{x}\in\boldsymbol{\Omega}}\|\mathrm{D}_x\boldsymbol{f}(\boldsymbol{x})\|_2$
	γ_{l_1}(IA)	$\underline{\gamma}_{l_1}$(IA)	γ_{l_1}(GS)	γ_{l_1}(S)	γ_{l_1}(H)	γ_{l_2}(IA)	$\underline{\gamma}_{l_2}$(IA)	γ_{l_2}(GS)	γ_{l_2}(S)	γ_{l_2}(H)	γ_l(GS)
31	0.4579	0.4578	0.4579	0.3426	0.3396	0.4579	0.4552	0.4579	0.4523	0.4497	0.1328
61	0.6445	0.6444	0.6445	0.4556	0.4435	0.6445	0.6408	0.6445	0.6365	0.6328	0.1329
91	0.7881	0.7881	0.7881	0.5314	0.5196	0.7881	0.7836	0.7881	0.7783	0.7738	0.1330
121	0.9093	0.9093	0.9093	0.6054	0.5832	0.9093	0.9041	0.9093	0.8980	0.8927	0.1330
151	1.0162	1.0161	1.0162	0.6686	0.6373	1.0162	1.0103	1.0162	1.0035	0.9976	0.1330
181	1.1128	1.1128	1.1128	0.7242	0.6964	1.1128	1.1064	1.1128	1.0990	1.0925	0.1330
211	1.2017	1.2017	1.2017	0.7791	0.7524	1.2017	1.1948	1.2017	1.1867	1.1797	0.1330
241	1.2844	1.2844	1.2844	0.8239	0.8086	1.2844	1.2770	1.2844	1.2684	1.2610	0.1330
271	1.3622	1.3621	1.3622	0.8713	0.8540	1.3622	1.3543	1.3622	1.3452	1.3373	0.1330
301	1.4357	1.4357	1.4357	0.9083	0.9019	1.4357	1.4274	1.4357	1.4178	1.4095	0.1330

$$e)\quad f_e(\boldsymbol{x}) = -\delta\alpha_i x_i^2, \tag{28e}$$

where $x_i := \rho_i$ and $\delta := \frac{v_f}{l\rho_m}$; v_f and ρ_m denote the free flow speed and maximum density (maximum density ρ_m is equal to $2\rho_c$). In this case, the following values, which are adapted from [54], are considered: $v_f = 31.3$ m/s, $\rho_m = 0.053$ vehicles/m, and $l = 500$ m. The splitting ratio for off-ramps are chosen to be $\alpha_i = 0.5$. The highway consists of a certain structure, depicted in Fig. 2, such that it has s sections and is containing n number of segments in total (including ramps) where $n = 6s + 1$.

Herein, two cases are considered which depend on the way Lipschitz constants are computed: *(i) Case 1* which uses (9b) provided in Theorem 1, and *(ii) Case 2* which uses (10) provided in Corollary 1. Specific for Case 2, firstly, the corresponding set $\boldsymbol{\Omega}$ is reduced so that the variables considered in the computation of $\max_{\boldsymbol{x}\in\boldsymbol{\Omega}}\|\nabla_x f_z(\boldsymbol{x})\|_2^2$ for each $z \in \{a, b, c, d, e\}$ in (28) are only the ones that appear in the expression of (28). Secondly, we only solve five different maximization problems corresponding to each type of nonlinearity in (28). Both of these steps apply the approaches discussed in Section III-F. The expressions of $\|\nabla_x f_z(\boldsymbol{x})\|_2^2$ for each z are listed as follows

$$a)\quad \|\nabla_x f_a(\boldsymbol{x})\|_2^2 = 4\delta^2 x_i^2 \tag{29a}$$
$$b)\quad \|\nabla_x f_b(\boldsymbol{x})\|_2^2 = 4\delta^2\left(x_i^2 + x_{i-1}^2 + x_j^2\right) \tag{29b}$$
$$c)\quad \|\nabla_x f_c(\boldsymbol{x})\|_2^2 = 4\delta^2\left(x_i^2 + x_{i-1}^2\right) \tag{29c}$$
$$d)\quad \|\nabla_x f_d(\boldsymbol{x})\|_2^2 = 4\delta^2\left(x_i^2 + x_{i-1}^2 + \alpha_j^2 x_j^2\right) \tag{29d}$$

$$e)\quad \|\nabla_x f_e(\boldsymbol{x})\|_2^2 = 4\delta^2\alpha_i^2 x_i^2. \tag{29e}$$

Due to the particular structure of the traffic network illustrated in Fig. 2, then for every $z \in \{a, b, c, d, e\}$, there exist $c_z \in \mathbb{N}$ highway segments which nonlinearities are of type z. Algorithm 3 presents the steps to compute Lipschitz constants for both cases—note that γ_{l_1} corresponds to Case 1 whereas γ_{l_2} corresponds to Case 2. In this test, we compare different number of total segments $n = 31, 61, \ldots, 301$. Constants ϵ_h and ϵ_Ω for Algorithm 3 are set to be 1×10^{-4} and 1×10^{-7} respectively whereas the region of interest is set to be $\boldsymbol{\Omega} = [0, \rho_c]^n$ since the traffic is assumed to be uncongested.

After running Algorithm 3 using the aforementioned settings, we obtain results that are detailed in Tab. II. All of the computed Lipschitz constants γ_{l_1} and γ_{l_2} are ϵ_h−optimal, since the gap between u and l are all less than the predefined ϵ_h. Interestingly, notice that the values of γ_{l_1} and γ_{l_2} are the same for both cases. This demonstrates that the formulations posed in (9b) and (10) can indeed return similar Lipschitz constants with a fine precision. For both cases, the resulting Lipschitz constants also considerably smaller than analytical results, which can be advantageous in observer/controller design. It is worth noting that, despite giving the same constants, Case 1 requires a significantly higher computation time than Case 2. This observation is expected since in Case 2, only five small optimization problems
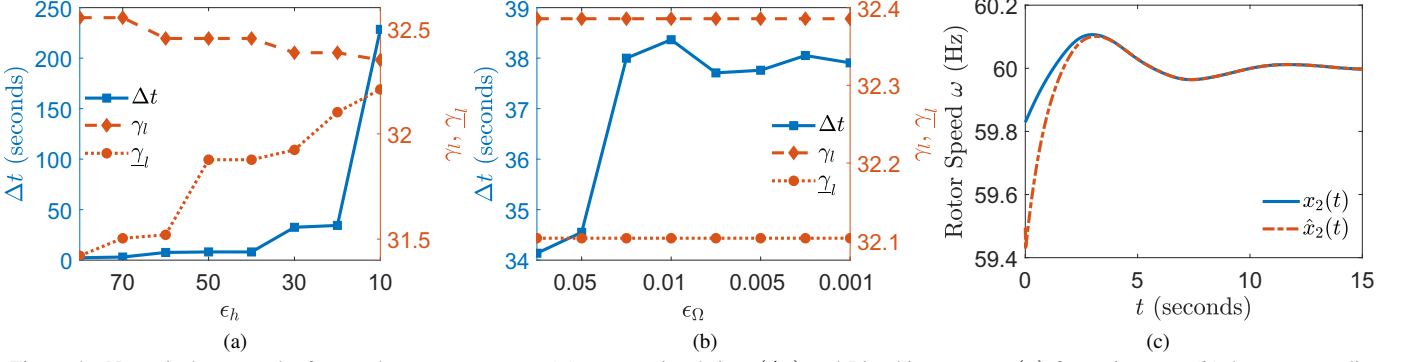
Figure 4. Numerical test results for synchronous generator: *(a)* computational time (Δt) and Lipschitz constant (γ) for various ϵ_h, *(b)* the corresponding results for various ϵ_Ω, and *(c)* the trajectories of estimated and actual generator rotor speed (ω) using a Lipschitz-based observer [1] with $\gamma_l = 32.3514$.

are solved and as such, the problem's complexity does *not* depend on the dimension of $\boldsymbol{f}(\cdot)$.

Next, we compare the interval-based algorithm (Algorithm 3) with stochastic point-based algorithm based on quasi-random *low discrepancy sequences* (LDS). This approach is utilized in [22] for Lipschitz constant approximation of synchronous generator model. We apply Algorithm 1 from [22] with 10^4 points from Sobol and Halton sequences generated through MATLAB's `sobolset` and `haltonset` functions. In addition, we also implement MATLAB's global search function `GlobalSearch` along with `fmincon` to find Lipschitz constants. The results of this comparison are provided in Tab. III and Fig. 3, where we also compare the results from using (4) with MATLAB's global search. It can be seen from this table that, for both cases, the Lipschitz constants computed via MATLAB's global search match with the ones from interval-based algorithm, while in contrast, the results from quasi-random search unsurprisingly are considerably lower for Case 1 but not so much for Case 2 (this numerically confirms that (9b) is indeed less conservative than (10)). Notice the increasing gaps of Lipschitz constants computed from utilizing interval-based algorithm and quasi-random search in Case 1 as n increases, which in turn suggests that the Lipshitz under-approximations computed via quasi-random search are getting worse as the problem's dimension increases. Lipschitz constant computation using (4) with MATLAB's global search returns values that are relatively constant, which is 0.1330. Despite giving the smallest Lipschitz constants, it is difficult to justify whether this value satisfies (9a) since MATLAB's global search does not guarantee optimality. On the matter of computational time however (see Fig. 3), particularly for Case 1, both interval-based algorithm and MATLAB's global search require much higher computation time compared to using LDS, which are much more scalable considering that 10^4 points from the LDS are used. Nonetheless, for Case 2, our interval-based algorithm outperforms the other methods with the lowest computation time whereas MATLAB's global search requires much higher computation time. Note that MATLAB's global search with (4) is the *least* scalable compared to other methods in both cases.

B. Lipschitz Parameterization for Synchronous Generator

The synchronous generator model considered in this example is taken from [22], [55], which is a fourth-order differential

Table IV
COMPARISON OF LIPSCHITZ CONSTANTS FOR SYNCHRONOUS GENERATOR MODEL COMPUTED VIA INTERVAL-BASED ALGORITHM (IA), POINT-BASED GLOBAL SEARCH (GS), AND POINT-BASED QUASI-RANDOM SEARCH WITH SOBOL (S) AND HALTON (H) SEQUENCES.

Method	IA	GS	GS†	S	H
γ_l	32.3514	32.3360	32.3356	32.1041	31.9437
$\underline{\gamma}_l$	32.2115	-	-	-	-

† the Lipschitz constant is computed using (4).

equation where the nonlinear dynamics are represented by the function $\boldsymbol{f} : \mathbb{R}^4 \times \mathbb{R}^4 \to \mathbb{R}^4$ detailed the following form

$$f_1(\boldsymbol{x}, \boldsymbol{u}) = -\alpha_1 \tag{30a}$$

$$
\begin{aligned}
f_2(\boldsymbol{x}, \boldsymbol{u}) = {} & \alpha_3 x_4 u_4 \cos x_1 - \alpha_3 x_3 u_4 \sin x_1 - \alpha_3 x_4 u_3 \sin x_1 \\
& - \alpha_3 x_3 u_3 \cos x_1 + \alpha_4 u_3 u_4 \cos 2x_1 \\
& + \tfrac{1}{2}\alpha_4 \left(u_4^2 - u_3^2\right) \sin 2x_1 + \alpha_6
\end{aligned}
\tag{30b}
$$

$$f_3(\boldsymbol{x}, \boldsymbol{u}) = \alpha_8 u_4 \cos x_1 - \alpha_8 u_3 \sin x_1 \tag{30c}$$

$$f_4(\boldsymbol{x}, \boldsymbol{u}) = \alpha_{10} u_3 \cos x_1 + \alpha_{10} u_4 \sin x_1, \tag{30d}$$

where α_1, α_3, α_4, α_6, α_8, and α_{10} are constants, $\boldsymbol{x}$ is the state vector, and $\boldsymbol{u}$ is the input vector such that

$$\boldsymbol{x} = \begin{bmatrix} x_1 & x_2 & x_3 & x_4 \end{bmatrix}^\top = \begin{bmatrix} \delta & \omega & e_{\mathrm{q}}' & e_{\mathrm{d}}' \end{bmatrix}^\top$$

$$\boldsymbol{u} = \begin{bmatrix} u_1 & u_2 & u_3 & u_4 \end{bmatrix}^\top = \begin{bmatrix} T_{\mathrm{m}} & E_{\mathrm{fd}} & i_{\mathrm{R}} & i_{\mathrm{I}} \end{bmatrix}^\top,$$

where δ is the rotor angle, ω is the rotor speed, and e_{q}' and e_{d}' are the transient voltage along q and d axes, T_{m} is the mechanical torque, E_{fd} is the internal field voltage, i_{R} and i_{I} are the real and imaginary currents of the generator [56]. In this case we consider a 16-machine, 68-bus system that is extracted from the PST toolbox [57] by considering Generator 16 in the network, in which the corresponding parameters are obtained from [57]. The generator's dynamic response are generated from applying a three-phase fault at bus 6 of line $6-11$ where each generator is using a transient model with IEEE Type DC1 excitation system and a simplified turbine-governor system [58]. To obtain lower and upper bounds on $\boldsymbol{x}$ and $\boldsymbol{u}$ (which are needed to construct $\boldsymbol{\Omega}$), their maxima and minima are measured from the dynamic response within a 10 seconds time frame. We use the expression from (9b) to compute the corresponding Lipschitz constant. As $\boldsymbol{f}(\cdot)$ in (30) contains sinusoidal functions, then we implement the n-th degree approximations of interval extensions of $\sin(\cdot)$ and $\cos(\cdot)$ (denoted by $\sin_n^I(\cdot)$ and $\cos_n^I(\cdot)$ respectively) from [41], which are specified as follows
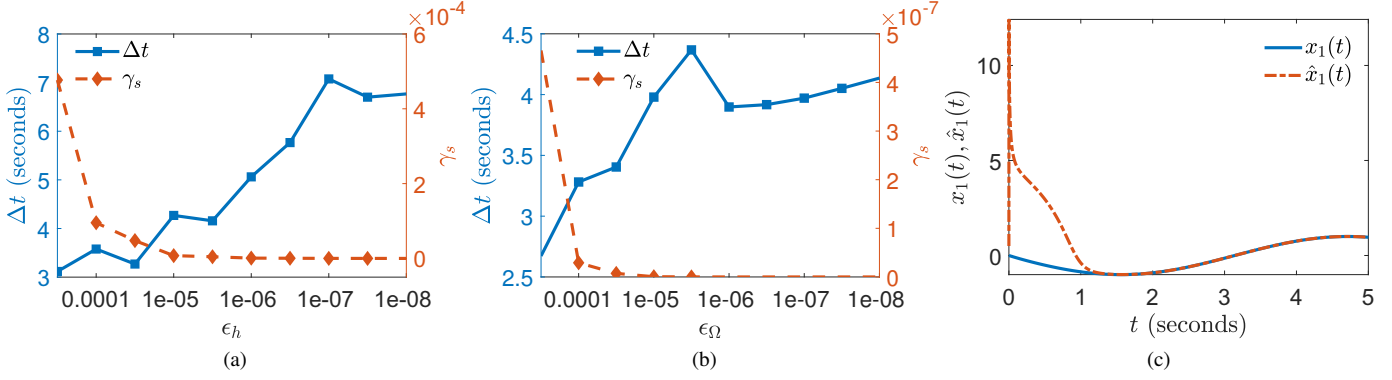
13

Figure 5. Numerical test results for dynamics of a moving object: *(a)* computational time (Δt) and OSL constant (γ_s) for various ϵ_h, *(b)* the corresponding results for various ϵ_Ω, and *(c)* the trajectories of actual $x_1(t)$ and estimated $\hat{x}_1(t)$ state with $\gamma_s = 0$, $\gamma_{q1} = 25015$, and $\gamma_{q2} = -9999.89$.

$$\sin_n^I(\omega) = \begin{cases} [\underline{s}(\underline{\omega}, n), \bar{s}(\bar{\omega}, n)], & \text{if } \omega \subseteq \left[-\frac{\underline{\pi}(n)}{2}, \frac{\underline{\pi}(n)}{2} \right] \\ [\underline{s}(\bar{\omega}, n), \bar{s}(\underline{\omega}, n)], & \text{if } \omega \subseteq \left[-\frac{\bar{\pi}(n)}{2}, \underline{\pi}(n) \right] \\ [\min(\underline{s}(\underline{\omega}, n), \underline{s}(\bar{\omega}, n)), 1], & \text{if } \omega \subseteq [0, \underline{\pi}(n)] \\ -\sin_n^I(-\omega), & \text{if } \omega \subseteq [-\underline{\pi}(n), 0] \\ [-1, 1], & \text{otherwise} \end{cases}$$

$$\cos_n^I(\omega) = \begin{cases} [\underline{c}(\bar{\omega}, n), \bar{c}(\underline{\omega}, n)], & \text{if } \omega \subseteq [0, \underline{\pi}(n)] \\ \cos_n^I(-\omega), & \text{if } \omega \subseteq [-\underline{\pi}(n), 0] \\ [\min(\underline{c}(\underline{\omega}, n), \underline{c}(\bar{\omega}, n)), 1], & \text{if } \omega \subseteq \left[-\frac{\underline{\pi}(n)}{2}, \frac{\underline{\pi}(n)}{2} \right] \\ [-1, 1], & \text{otherwise} \end{cases}$$

where in the above expressions, $\underline{s}(x, n)$ and $\bar{s}(x, n)$ are the n-th degree partial approximations of $\sin(x)$ whereas $\underline{c}(x, n)$ and $\bar{c}(x, n)$ are the n-th degree partial approximations of $\cos(x)$ such that the following inequalities hold [41]

$$\underline{s}(x, n) \le \sin(x) \le \bar{s}(x, n), \quad \underline{c}(x, n) \le \cos(x) \le \bar{c}(x, n).$$

Likewise, $\underline{\pi}(n)$ and $\bar{\pi}(n)$ are the n-th degree partial approximations of the irrational constant π such that $\underline{\pi}(n) \le \pi \le \bar{\pi}(n)$; interested readers can refer to [41] for the detailed mathematical expressions of these approximations. Of course, higher value of n can yield better approximations. In this test, we consider $n = 4$ as it provides sufficient approximations for this model. To obtain sharper bounds when evaluating interval functions, the corresponding subset is subdivided into 10 segments (or smaller subsets) and the resulting final intervals are computed based on the union of each intervals from the evaluation of these segments—see [45] for more details.

The goals of the first numerical experiment in this section are twofold. First, we would like to find the relation between the values of the computed Lipschitz constant and computation time with different precision on Algorithm 1 which is carried out by assigning different values for ϵ_h and ϵ_Ω. Secondly, we are interested in investigating the practical usefulness of the computed Lipschitz constant for actual observer design. To achieve the first objective, two scenarios are considered: first, the value for ϵ_h is fixed while changing the value of ϵ_Ω and second we consider the other way, i.e., fixing ϵ_Ω while using different values for ϵ_h. The results of this experiment are given in Figs. 4a and 4b. It can be seen from Fig. 4a that the computed Lipschitz constants γ_l are decreasing as the values of ϵ_h becoming smaller, while on the other hand, the corresponding lower bounds $\underline{\gamma}_l$ are increasing. This indicates

that the gaps between γ_l and $\underline{\gamma}_l$ are getting smaller as ϵ_h is reduced, in spite of the expense of increasing computation time. Similar trend is not observed when ϵ_h is fixed while ϵ_Ω varies—resulting in non-varying Lipschitz values and their lower bounds—see Fig. 4b. These results suggest that, for this particular case, a better Lipschitz constant can be obtained from increasing the algorithm's precision—that is, reducing the value of ϵ_h. In comparison with the analytical Lipschitz constant computed using the formula provided in [22] which is equal to 749.8279, the Lipschitz constant obtained using the proposed approach is indeed far less conservative. Second, to answer the second objective, we use the computed Lipschitz constant $\gamma_l = 32.3514$ and implement a Lipschitz-based observer for NDS developed in [1] for performing dynamic state estimation on this generator. We find that, with the given Lipschitz constant, the computed observer gain matrix is able to provide a converging estimation error. This finding is corroborated by the converging estimated generator rotor speed ω, denoted by $\hat{x}_2(t)$, to the actual one $x_2(t)$, as shown in Fig. 4c. The results of this numerical test showcase the applicability of the proposed approach in computing Lipschitz constant for practical observer design and state estimation.

In the second numerical test, we again compare the interval-based algorithm (Algorithm 1) with point-based methods including MATLAB's global search and quasi-random search algorithms with Sobol and Halton sequences. The interval-based algorithm is configured with $\epsilon_h = 10$ and $\epsilon_\Omega = 10^{-4}$. In addition to using (9b), we also use the 2-norm of the Jacobian matrix as provided in (4) to compute Lipschitz constant with MATLAB's global search. The results of this test are shown in Tab. IV. It is observed that the interval-based algorithm provides both Lipschitz constant and its lower bound with relatively tight gap, while the Lipschitz constants obtained from other approaches are comparable. Note that MATLAB's global search with (4) gives slightly smaller value than from using (9b), while quasi-random search again gives under-approximations of Lipschitz constant.

C. OSL and QIB Parameterization

In the final numerical example, we consider the equation that represents the motion of a moving object in $2-$D plane given in [4] and described in the following state-space equation

$$\dot{\boldsymbol{x}} = \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix} \boldsymbol{x} + \begin{bmatrix} -x_1(x_1^2 + x_2^2) \\ -x_2(x_1^2 + x_2^2) \end{bmatrix}, \quad \boldsymbol{y} = \begin{bmatrix} 0 & 1 \end{bmatrix} \boldsymbol{x}, \quad (31)$$

with domain of interest $\boldsymbol{\Omega} = \{\boldsymbol{x} \in \mathbb{R}^2 \,|\, x_i \in [-r, r], \, i = 1, 2\}$ where $r = 5$. Here we aim to *(a)* find the OSL constant γ_s and QIB constants γ_{q1} and γ_{q2} that are *(b)* useful for observer design. To that end, first we focus on finding γ_s using the method given in Proposition 2. The corresponding matrix $\boldsymbol{\Xi}(\boldsymbol{x})$ for the nonlinear functions in (31) is of the form

$$\boldsymbol{\Xi}(\boldsymbol{x}) = \begin{bmatrix} -3x_1^2 - x_2^2 & -2x_1 x_2 \\ -2x_1 x_2 & -x_1^2 - 3x_1^2 \end{bmatrix}.$$

Since $\boldsymbol{\Xi}(\boldsymbol{x})$ in the above is already symmetric, then according to (17), γ_s can be computed as

$$\gamma_s = \max_{i \in \{1,2\}} \left(\max_{\boldsymbol{x} \in \boldsymbol{\Omega}} \left(\boldsymbol{\Xi}_{(i,i)} + \sum_{j \in \{1,2\} \backslash i} \left| \boldsymbol{\Xi}_{(i,j)} \right| \right) \right). \quad (32)$$

In using Algorithm 1 to solve (32), we first fix ϵ_Ω to 10^{-8} and use various values for ϵ_h ranging from 5×10^{-4} to 10^{-8} then in turn, we fix ϵ_h such that $\epsilon_h = 10^{-12}$ and use various values for ϵ_Ω. The results of this numerical example is illustrated in Figs. 5a and 5b. It can be seen from these figures that, as ϵ_h getting smaller, the value of γ_s is decreasing to a value near zero. Therefore, we conclude that the OSL constant for this system is zero. This results corroborates the fact that, as proven in [4], the system is indeed globally OSL with $\gamma_s = 0$. Notice that in Fig. 5b, the computation time is increasing until $\epsilon_\Omega = 5 \times 10^{-6}$ and stays at around 4 seconds afterwards since for $\epsilon_\Omega > 5 \times 10^{-6}$, the condition $u - l < \epsilon_h$ can be achieved without the need for optimizing the lower bound.

Next, we put our attention on determining the QIB constant. To obtain such constant, we solve the following problems using Algorithm 1

$$\underline{\gamma} = \min_{i \in \{1,2\}} \left(\min_{\boldsymbol{x} \in \boldsymbol{\Omega}} \left(\boldsymbol{\Xi}_{(i,i)} - \sum_{j \in \{1,2\} \backslash i} \left| \boldsymbol{\Xi}_{(i,j)} \right| \right) \right)$$
$$\gamma_m = \max_{\boldsymbol{x} \in \boldsymbol{\Omega}} \sum_{i \in \{1,2\}} \|\nabla_x f_i(\boldsymbol{x})\|_2^2,$$

in which we obtain $\underline{\gamma} = -150$ and $\gamma_m = 2.5 \times 10^4$. According to Proposition 4, for any $\epsilon_1, \epsilon_2 \in \mathbb{R}_+$, the QIB constant can be constructed as $\gamma_{q2} = \epsilon_2 - \epsilon_1$ and $\gamma_{q1} = \epsilon_1 \gamma_s - \epsilon_2 \underline{\gamma} + \gamma_m$. By setting the constants $\epsilon_1 = 10^5$ and $\epsilon_2 = 10^{-1}$, we obtain $\gamma_{q1} = 25015$ and $\gamma_{q2} = -9999.89$. To test the applicability of the computed OSL and QIB constants for state estimation, we implement an observer developed in [6] using the computed constants, in which we are successfully able to get a converging estimation error. Fig. 5c depicts the trajectories of the first actual and estimated state for this system.

VI. Summary and Future Directions

To the best of our knowledge, this work is the first thorough research effort that presents analytical and computational methods to parameterize NDS and various corresponding function sets. Specifically, we propose a novel, unified framework for NDS parameterization for bounded Jacobian, Lipschitz continuous, OSL, QIB, and QB function sets. Our contributions are *(1)* posing and analytically deriving closed-form global maximization problems for NDS parameterization, *(2)* investigating a derivative-free, interval-based algorithm for NDS parameterization, and *(3)* showcasing the proposed methodology for performing parameterization for some NDS models in traffic

and power networks. In addition, we offer various strategies for reducing the complexity in parameterizing NDS using the proposed interval-based global maximization algorithm.

The presented methods in this paper are not devoid of limitations; several future research directions are worthy of further investigation. First, there exist other classes of nonlinearities that are not considered in this paper such as *incremental quadratic nonlinearity* [59] and *generalized sector nonlinearity* [60]. Parameterizing NDS for these kind of nonlinearities will be studied in our future work. Second, since we do not consider any disturbance—that may be caused by modeling uncertainty or time-varying parameters—on the nonlinearities, then future work will also include investigating robust NDS parameterization. Third, investigating the impact of obtaining function sets and bounding constants, through our proposed algorithms, on LMI/SDP feasibility for observer and controller designs is another direction worthy of exploring.

References

[1] G. Phanomchoeng and R. Rajamani, "Observer design for lipschitz nonlinear systems using riccati equations," in *Proceedings of the 2010 American Control Conference*, June 2010, pp. 6060–6065.

[2] D. Ichalal, B. Marx, S. Mammar, D. Maquin, and J. Ragot, "Observer for lipschitz nonlinear systems: mean value theorem and sector nonlinearity transformation," in *2012 IEEE International Symposium on Intelligent Control*. IEEE, 2012, pp. 264–269.

[3] A. Zemouche and M. Boutayeb, "Observer design for lipschitz nonlinear systems: the discrete-time case," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 53, no. 8, pp. 777–781, 2006.

[4] M. Abbaszadeh and H. J. Marquez, "Nonlinear observer design for one-sided lipschitz systems," in *Proceedings of the 2010 American Control Conference*, June 2010, pp. 5284–5289.

[5] M. Benallouch, M. Boutayeb, and M. Zasadzinski, "Observer design for one-sided lipschitz discrete-time systems," *Systems & Control Letters*, vol. 61, no. 9, pp. 879–886, 2012.

[6] W. Zhang, H. Su, H. Wang, and Z. Han, "Full-order and reduced-order observers for one-sided lipschitz nonlinear systems using riccati equations," *Communications in Nonlinear Science and Numerical Simulation*, vol. 17, no. 12, pp. 4968–4977, 2012.

[7] G. Phanomchoeng and R. Rajamani, "The bounded jacobian approach to nonlinear observer design," in *Proceedings of the 2010 American Control Conference*, June 2010, pp. 6083–6088.

[8] G. Phanomchoeng, R. Rajamani, and D. Piyabongkarn, "Nonlinear observer for bounded jacobian systems, with applications to automotive slip angle estimation," *IEEE Transactions on Automatic Control*, vol. 56, no. 5, pp. 1163–1170, May 2011.

[9] Y. Wang, R. Rajamani, and A. Zemouche, "Multi-objective nonlinear observer design using bmis," in *2018 Annual American Control Conference (ACC)*, June 2018, pp. 1346–1351.

[10] A. Zemouche and M. Boutayeb, "On lmi conditions to design observers for lipschitz nonlinear systems," *Automatica*, vol. 49, no. 2, pp. 585–591, 2013.

[11] M. Jin, H. Feng, and J. Lavaei, "Multiplier-based observer design for large-scale lipschitz systems," 2018.

[12] D. D. Siljak and D. M. Stipanovic, "Robust stabilization of nonlinear systems: The lmi approach." *Mathematical Problems in Engineering*, vol. 6, no. 5, pp. 461–493, 2000.

[13] D. D. Siljak, D. M. Stipanovic, and A. I. Zecevic, "Robust decentralized turbine/governor control using linear matrix inequalities," *IEEE Transactions on Power Systems*, vol. 17, no. 3, pp. 715–722, Aug 2002.

[14] M. Yadegar, A. Afshar, and M. Davoodi, "Observer-based tracking controller design for a class of lipschitz nonlinear systems," *Journal of Vibration and Control*, vol. 24, no. 11, pp. 2112–2119, 2018.

[15] A. Zemouche, R. Rajamani, H. Kheloufi, and F. Bedouhene, "Robust observer-based stabilization of lipschitz nonlinear uncertain systems via lmis-discussions and new design procedure," *International Journal of Robust and Nonlinear Control*, vol. 27, no. 11, pp. 1915–1939, 2017.

[16] M. Ekramian, "Observer-based controller for lipschitz nonlinear systems," *International Journal of Systems Science*, vol. 48, no. 16, pp. 3411–3418, 2017.

[17] J. Song and S. He, "Robust finite-time h_∞ control for one-sided lipschitz nonlinear systems via state feedback and output feedback," *Journal of the Franklin Institute*, vol. 352, no. 8, pp. 3250–3266, 2015.

[18] L. Liu and X. Song, "Static output tracking control of nonlinear systems with one-sided lipschitz condition," *Mathematical Problems in Engineering*, vol. 2014, 2014.

[19] A. Rastegari, M. M. Arefi, and M. H. Asemani, "Robust h_∞ sliding mode observer-based fault-tolerant control for one-sided lipschitz nonlinear systems," *Asian Journal of Control*, vol. 21, no. 1, pp. 114–129, 2019.

[20] H. Gholami and T. Binazadeh, "Observer-based h_∞ finite-time controller for time-delay nonlinear one-sided lipschitz systems with exogenous disturbances," *Journal of Vibration and Control*, vol. 25, no. 4, pp. 806–819, 2019.

[21] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," 2014.

[22] S. A. Nugroho, A. F. Taha, and J. Qi, "Characterizing the nonlinearity of power system generator models," in *2019 American Control Conference (ACC)*. IEEE, July 2019, pp. 1936–1941.

[23] G. Wood and B. Zhang, "Estimation of the lipschitz constant of a function," *Journal of Global Optimization*, vol. 8, no. 1, pp. 91–103, 1996.

[24] R. Paulavičius and J. Žilinskas, "Analysis of different norms and corresponding lipschitz constants for global optimization," *Technological and Economic Development of Economy*, vol. 12, no. 4, pp. 301–306, 2006.

[25] M. S. Darup and M. Mönnigmann, "Fast computation of lipschitz constants on hyperrectangles using sparse codelists," *Computers and Chemical Engineering*, vol. 116, pp. 135 – 143, 2018.

[26] M. Fazlyab, A. Robey, H. Hassani, M. Morari, and G. J. Pappas, "Efficient and accurate estimation of lipschitz constants for deep neural networks," *CoRR*, vol. abs/1906.04893, 2019.

[27] A. Chakrabarty, D. K. Jha, G. T. Buzzard, Y. Wang, and K. G. Vamvoudakis, "Safe approximate dynamic programming via kernelized lipschitz estimation," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–15, 2020.

[28] B. Moa, "Interval methods for global optimization," Ph.D. dissertation, University of Victoria, 2007.

[29] S. A. Nugroho, V. Hoang, M. Radosz, S. Wang, and A. F. Taha, "New Insights on One-Sided Lipschitz and Quadratically Inner-Bounded Nonlinear Dynamic Systems," in *The 2020 American Control Conference, Denver, Colorado*, 2020. [Online]. Available: https://arxiv.org/pdf/2002.02361.pdf

[30] M. Abbaszadeh and H. J. Marquez, "Design of nonlinear state observers for one-sided lipschitz systems," *CoRR*, vol. abs/1302.5867, 2013. [Online]. Available: http://arxiv.org/abs/1302.5867

[31] S. A. Nugroho. (2020) Nonlinear dynamic systems parameterization. Accessed: May 2020. [Online]. Available: https://github.com/antimon40/Nonlinear_Dynamic_Systems_Parameterization.git

[32] W. Trench, *Introduction to Real Analysis*. Prentice Hall/Pearson Education, 2003.

[33] H. Márquez, *Nonlinear Control Systems: Analysis and Design*.

[34] H. Khalil, *Nonlinear Systems*, ser. Pearson Education. Prentice Hall, 2002.

[35] W. Rudin, *Principles of Mathematical Analysis*, 3rd ed. New York: McGraw-Hill Book Co., 1976, international Series in Pure and Applied Mathematics.

[36] R. Horn and C. Johnson, *Matrix Analysis*. Cambridge University Press, 2013.

[37] I. Bárány and J. Solymosi, "Gershgorin disks for multiple eigenvalues of non-negative matrices," in *A Journey Through Discrete Mathematics*. Springer, 2017, pp. 123–133.

[38] Y. Guo, Y. Chen, and C. Zhang, "Decentralized state-observer-based traffic density estimation of large-scale urban freeway network by dynamic model," *Information*, vol. 8, no. 3, 2017.

[39] S. A. Nugroho, A. F. Taha, and C. Claudel, "Traffic density modeling and estimation on stretched highways: The case for lipschitz-based observers," in *2019 American Control Conference (ACC)*. IEEE, July 2019, pp. 2658–2663.

[40] S. A. Nugroho, A. F. Taha, and C. G. Claudel, "A control-theoretic approach for scalable and robust traffic density estimation using convex optimization," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–15, 2019.

[41] M. Daumas, D. Lester, and C. Munoz, "Verified real number calculations: A library for interval arithmetic," *IEEE Transactions on Computers*, vol. 58, no. 2, pp. 226–237, 2009.

[42] R. E. Moore, R. B. Kearfott, and M. J. Cloud, *Introduction to Interval Analysis*. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 2009.

[43] E. Hansen and G. Walster, *Global Optimization Using Interval Analysis: Revised And Expanded*. CRC Press, 2003.

[44] E. Van Kampen, "Global optimization using interval analysis: Interval optimization for aerospace applications," Ph.D. dissertation, Delft University of Technology, 2010.

[45] K. Ichida and Y. Fujii, "An interval arithmetic method for global optimization," *Computing*, vol. 23, no. 1, pp. 85–97, 1979.

[46] E. Hansen, "Global optimization using interval analysis – the multidimensional case," *Numer. Math.*, vol. 34, no. 3, p. 247–270, Sep. 1980.

[47] S. Skelboe, "Computation of rational interval functions," *BIT Numerical Mathematics*, vol. 14, no. 1, pp. 87–95, 1974.

[48] L. Jaulin, M. Kieffer, O. Didrit, and E. Walter, *Applied Interval Analysis: With Examples in Parameter and State Estimation, Robust Control and Robotics*. Springer Science & Business Media, 2001.

[49] R. E. Moore, "On computing the range of a rational function of n variables over abounded region," *Computing*, vol. 15, pp. 1–15, 1976.

[50] M. Van Emden and B. Moa, "Termination criteria in the moore-skelboe algorithm for global optimization by interval arithmetic," in *Frontiers in Global Optimization*. Springer, 2004, pp. 585–597.

[51] J. Löfberg, "YALMIP: A toolbox for modeling and optimization in matlab," in *Proc. IEEE Int. Symp. Computer Aided Control Syst. Design*, 2004, pp. 284–289.

[52] E. D. Andersen and K. D. Andersen, "The mosek interior point optimizer for linear programming: an implementation of the homogeneous algorithm," in *High performance optimization*. Springer, 2000, pp. 197–232.

[53] B. Greenshields, W. Channing, H. Miller *et al.*, "A study of traffic capacity," in *Highway research board proceedings*, vol. 1935. National Research Council (USA), Highway Research Board, 1935.

[54] S. Contreras, P. Kachroo, and S. Agarwal, "Observability and sensor placement problem on highway segments: A traffic dynamics-based approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 3, pp. 848–858, 2016.

[55] S. A. Nugroho, A. F. Taha, and J. Qi, "Robust dynamic state estimation of synchronous machines with asymptotic state estimation error performance guarantees," *IEEE Transactions on Power Systems*, vol. 35, no. 3, pp. 1923–1935, 2020.

[56] P. Kundur, N. J. Balu, and M. G. Lauby, *Power system stability and control*. McGraw-hill New York, 1994, vol. 7.

[57] J. H. Chow and K. W. Cheung, "A toolbox for power system dynamics and control engineering education and research," *Power Systems, IEEE Transactions on*, vol. 7, no. 4, pp. 1559–1564, 1992.

[58] J. Qi, A. F. Taha, and J. Wang, "Comparing kalman filters and observers for power system dynamic state estimation with model uncertainty and malicious cyber attacks," *IEEE Access*, vol. 6, pp. 77 155–77 168, 2018.

[59] B. Açıkmeşe and M. Corless, "Observers for systems with nonlinearities satisfying incremental quadratic constraints," *Automatica*, vol. 47, no. 7, pp. 1339 – 1348, 2011.

[60] K. Vijayaraghavan, "Observer design for generalized sector-bounded noisy nonlinear systems," *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, vol. 228, no. 9, pp. 645–657, 2014.

[61] R. Wu, W. Zhang, F. Song, Z. Wu, and W. Guo, "Observer-based stabilization of one-sided lipschitz systems with application to flexible link manipulator," *Advances in Mechanical Engineering*, vol. 7, no. 12, p. 1687814015619555, 2015.

APPENDIX A
PROOF OF THEOREM 7

The proof requires the following propositions:

Proposition 4. *Let ω^* be a maximizer for problem* (27)*. If $C = \{\mathcal{S}_i\}_{i=1}^N$ is a cover in $\mathbf{\Omega}$ that contains a maximizer, then we have $h^* \in [l, u]$ where*

$$l = \max_{\mathcal{S}_i \in C} \inf\left(h^I(\mathcal{S}_i)\right), \quad u = \max_{\mathcal{S}_i \in C} \sup\left(h^I(\mathcal{S}_i)\right). \quad (33)$$

Proof. Suppose that the cover $C = \{\mathcal{S}_i\}_{i=1}^N$ in $\mathbf{\Omega}$ contains a maximizer for problem (27). As C contains a maximizer, then $\omega^* \in \mathcal{S}_j$ for some $j \in \mathbb{I}(N)$, which in turn implies that $h^* \leq \sup\left(h^I(\mathcal{S}_j)\right)$. Since $\sup\left(h^I(\mathcal{S}_j)\right) \leq u$, then it follows that $h^* \leq u$. Next, consider $\mathcal{S}_k \in C$ such that $l = \inf\left(h^I(\mathcal{S}_k)\right)$. As $\mathcal{S}_k \neq 0$, then there exists some points $\omega \in \mathcal{S}_k$ for which $h(\omega) \leq h^*$. However, since $\inf\left(h^I(\mathcal{S}_k)\right) \leq h(\omega)$, then $l \leq h^*$. This shows that $l \leq h^* \leq u$. ∎

Proposition 5. *Let ω^* be a maximizer for problem* (27) *and $C = \{\mathcal{S}_i\}_{i=1}^N$ be a cover in $\mathbf{\Omega}$ containing a maximizer. If there exists $\mathcal{S}_j \in C$ for some $j \in \mathbb{I}(N)$ such that $\sup\left(h^I(\mathcal{S}_j)\right) < l$, then $\omega^* \notin \mathcal{S}_j$.*

Proof. Note that, as the cover C contains a maximizer, then from Proposition 4 it holds that $l \leq h^* \leq u$. By contradiction, suppose that there exists such $\mathcal{S}_j \in C$, where $j \in \mathbb{I}(N)$, that $\sup\left(h^I(\mathcal{S}_j)\right) < l$ and $\omega^* \in \mathcal{S}_j$. However, since the assertion $\omega^* \in \mathcal{S}_j$ implies $h^* \leq \sup\left(h^I(\mathcal{S}_j)\right)$, we must have $h^* < l$, which is a contradiction. ∎

We start the proof by assuming that $\mathbf{\Omega}$ is nonempty such that $\omega^* \in \mathbf{\Omega}$. From the initialization, obviously C contains the maximizer. In the first part, we will show that the first loop in Step 4 computes the best upper bound for h^*. Note that, due to Proposition 4, at every instance we always have $l \leq h^* \leq u$ with $\omega^* \in C$. Since $u = \sup\left(h^I(\bar{\mathcal{S}})\right)$ where $\bar{\mathcal{S}}$ is an element of C with the greatest upper bound, it is divided into two subsets namely $\mathcal{S}_l$ and $\mathcal{S}_r$. As $\mathcal{S}_l \subset \bar{\mathcal{S}}$ and $\mathcal{S}_r \subset \bar{\mathcal{S}}$, by inclusion isotonicity, it then holds that $h^I(\mathcal{S}_l) \subset h^I(\bar{\mathcal{S}})$ and $h^I(\mathcal{S}_r) \subset h^I(\bar{\mathcal{S}})$, implying that $\sup\left(h^I(\mathcal{S}_l)\right) < \sup\left(h^I(\bar{\mathcal{S}})\right)$ and $\sup\left(h^I(\mathcal{S}_r)\right) < \sup\left(h^I(\bar{\mathcal{S}})\right)$. This shows that splitting of $\bar{\mathcal{S}}$ yields a better upper bound. In Algorithm 2, any $\mathcal{S}_i \in C$ satisfying $\sup\left(h^I(\mathcal{S}_i)\right) < l$ may be discarded since, according to Proposition 5, such subset does not contain any maximizer. Thus, we always have $\omega^* \in C$ and $l \leq h^* \leq u$. When the first loop in Step 4 terminates, we have $u - l \leq \epsilon_h$—therefore, h^* is ϵ_h−optimal and the algorithm terminates—or $u - l > \epsilon_h$. In the latter, $\left|\bar{\mathcal{S}}\right| \leq \epsilon_\Omega$ hence it cannot be split further. Either case, we have u as the best upper bound for h^*.

In the second part, it will be shown that l provides the best lower bound for h^*. Keep in mind that the temporary cover $\tilde{C}$ may be nonempty only if $u - l > \epsilon_h$ and there exists $\mathcal{S}_i \in C$ such that $|\mathcal{S}_i| > \epsilon_\Omega$. Note that $\bar{\mathcal{S}} \notin \tilde{C}$. Since we split $\underline{\mathcal{S}}$ into two subsets namely $\mathcal{S}_l$ and $\mathcal{S}_r$ where $\underline{\mathcal{S}}$ is an element of $\tilde{C}$ with the biggest lower bound, then as demonstrated above, we have $h^I(\mathcal{S}_l) \subset h^I(\underline{\mathcal{S}})$ and $h^I(\mathcal{S}_r) \subset h^I(\underline{\mathcal{S}})$, implying that $\inf\left(h^I(\underline{\mathcal{S}})\right) < \inf\left(h^I(\mathcal{S}_l)\right)$ and $\inf\left(h^I(\underline{\mathcal{S}})\right) < \inf\left(h^I(\mathcal{S}_r)\right)$. This shows that we attain a better lower bound by splitting $\underline{\mathcal{S}}$. When the second loop in Step 9 terminates, we have obtained a ϵ_h−optimal solution or every $\mathcal{S}_i \in C$ satisfies $|\mathcal{S}_i| \leq \epsilon_\Omega$—thus $\tilde{C} = \emptyset$. At this point, every subsets in C cannot be split further. Thus, when $u - l > \epsilon_h$, l can be updated by evaluating objective function evaluation of corner points on each $\mathcal{S}_i \in C$

and removing $\mathcal{S}_j \in C$ satisfying $\sup\left(h^I(\mathcal{S}_j)\right) < l$. At the end of Algorithm 1, we have achieved $u - l \leq \epsilon_h$ or $u - l > \epsilon_h$—in the former case, we have a ϵ_h−optimal solution. Either case, $[l, u]$ is the best interval containing h^*. Since we know that $\omega^* \in C$, there exists $\mathcal{S}_j$ for some $j \in \mathbb{I}(N)$ for which $\omega^* \in \mathcal{S}_j$. Thus, it immediately follows that $\omega^* \in \prod_{i \in \mathbb{I}(N)} \mathcal{S}_i$. This ends the proof. ∎

OBSERVER, CONTROLLER, AND OBSERVER-BASED STABILIZATION FOR DIFFERENT CLASSES OF NDS

Table V summarizes several type of observer design procedure via LMI for NDS which nonlinearities satisfy one of the following condition: bounded Jacobian, Lipschitz continuous, one-sided Lipschitz, and quadratically inner-bounded. Table VI shows some controller and observer-based stabilization design via LMI for NDS with such nonlinearities.

Table V
OBSERVER DESIGNS FOR NDS WITH DIFFERENT CLASSES OF NONLINEARITY THROUGH SDPs OR LMIs [1], [6], [11], [40].

Dynamics and Nonlinearity Class	Observer Structures	Observer Design via LMI
$\dot{\boldsymbol{x}} = \boldsymbol{Ax} + \boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{Bu}$ $\boldsymbol{y} = \boldsymbol{Cx} + \boldsymbol{Du}$ **Glob./Loc. Lipschitz:** $\|\boldsymbol{f}(\boldsymbol{x}) - f(\hat{\boldsymbol{x}})\|_2 \leq \gamma_l \|\boldsymbol{x} - \hat{\boldsymbol{x}}\|_2$	$\dot{\hat{\boldsymbol{x}}} = \boldsymbol{A\hat{x}} + \boldsymbol{f}(\hat{\boldsymbol{x}}) + \boldsymbol{Bu}$ $\quad + \boldsymbol{L}(\boldsymbol{y} - \hat{\boldsymbol{y}})$ $\hat{\boldsymbol{y}} = \boldsymbol{C\hat{x}} + \boldsymbol{Du}$ $\boldsymbol{L} = \boldsymbol{P}^{-1}\boldsymbol{Y}$ Variables: $\boldsymbol{P}, \boldsymbol{Y}, \kappa$	find $\boldsymbol{P} \succ 0, \boldsymbol{Y}, \kappa > 0$ s.t. $\begin{bmatrix} \boldsymbol{A}^\top \boldsymbol{P} + \boldsymbol{PA} - \boldsymbol{YC} & & \\ -\boldsymbol{C}^\top \boldsymbol{Y}^\top + \kappa \gamma_l^2 \boldsymbol{I} & * & \\ \boldsymbol{P} & & -\kappa \boldsymbol{I} \end{bmatrix} \prec 0,$
$\dot{\boldsymbol{x}} = \boldsymbol{Ax} + \boldsymbol{Gf}(\boldsymbol{x}) + \boldsymbol{Bu}$ $\boldsymbol{y} = \boldsymbol{Cx} + \boldsymbol{Du}$ **One-Sided Lipschitz, Quadratically** **Inner-Bounded:** constants $\gamma_s, \gamma_{q1}, \gamma_{q2}$ (see Tab. I)	$\dot{\hat{\boldsymbol{x}}} = \boldsymbol{A\hat{x}} + \boldsymbol{Gf}(\hat{\boldsymbol{x}}) + \boldsymbol{Bu}$ $\quad + \boldsymbol{L}(\boldsymbol{y} - \hat{\boldsymbol{y}})$ $\hat{\boldsymbol{y}} = \boldsymbol{C\hat{x}} + \boldsymbol{Du}$ $\boldsymbol{L} = \frac{1}{2}\sigma \boldsymbol{P}^{-1}\boldsymbol{C}^\top$ Variables: $\boldsymbol{P}, \sigma, \epsilon_1, \epsilon_2$	find $\boldsymbol{P} \succ 0, \{\sigma, \epsilon_1, \epsilon_2\} > 0$ s.t. $\begin{bmatrix} \boldsymbol{A}^\top \boldsymbol{P} + \boldsymbol{PA} + \epsilon_1 \gamma_s \boldsymbol{I} & & \\ + \epsilon_2 \gamma_{q1}\boldsymbol{I} - \sigma \boldsymbol{C}^\top \boldsymbol{C} & * & \\ \boldsymbol{G}^\top \boldsymbol{P} + \frac{\gamma_{q2}\epsilon_2 - \epsilon_1}{2}\boldsymbol{I} & -\epsilon_2 \boldsymbol{I} \end{bmatrix} \prec 0$
$\dot{\boldsymbol{x}} = \boldsymbol{Ax} + \boldsymbol{B}_u \boldsymbol{u} + \boldsymbol{Gf}(\boldsymbol{x}) + \boldsymbol{B}_w \boldsymbol{w}$ $\boldsymbol{y} = \boldsymbol{Cx} + \boldsymbol{D}_u \boldsymbol{u} + \boldsymbol{D}_w \boldsymbol{w}$ **Glob./Loc. Lipschitz:** $\|\boldsymbol{f}(\boldsymbol{x}) - f(\hat{\boldsymbol{x}})\|_2 \leq \gamma_l \|\boldsymbol{x} - \hat{\boldsymbol{x}}\|_2$	$\dot{\hat{\boldsymbol{x}}} = \boldsymbol{A\hat{x}} + \boldsymbol{Gf}(\hat{\boldsymbol{x}}) + \boldsymbol{Bu}$ $\quad + \boldsymbol{L}(\boldsymbol{y} - \hat{\boldsymbol{y}})$ $\hat{\boldsymbol{y}} = \boldsymbol{C\hat{x}} + \boldsymbol{D}_u \boldsymbol{u}$ $\boldsymbol{L} = \boldsymbol{P}^{-1}\boldsymbol{Y}$ Variables: $\boldsymbol{P}, \boldsymbol{Y}, \kappa, \mu_1, \mu_2$ Constant: $\alpha > 0$	min $\mu_1 + \mu_2$ s.t. $\begin{bmatrix} \boldsymbol{A}^\top \boldsymbol{P} + \boldsymbol{PA} - \boldsymbol{YC} & & & \\ -\boldsymbol{C}^\top \boldsymbol{Y}^\top + \kappa \gamma_l^2 \boldsymbol{I} + \alpha \boldsymbol{P} & * & * & \\ \boldsymbol{G}^\top \boldsymbol{P} & -\kappa \boldsymbol{I} & * & \\ \boldsymbol{B}_w^\top \boldsymbol{P} - \boldsymbol{D}_w^\top \boldsymbol{Y}^\top & \boldsymbol{O} & -\alpha \mu_1 \boldsymbol{I} \end{bmatrix} \prec 0$ $\begin{bmatrix} -\boldsymbol{P} & * & * \\ \boldsymbol{O} & -\mu_2 \boldsymbol{I} & * \\ \boldsymbol{Z} & \boldsymbol{O} & -\boldsymbol{I} \end{bmatrix} \prec 0$ $\boldsymbol{P} \succ 0, \boldsymbol{Y}, \{\kappa, \mu_1, \mu_2\} > 0$
$\dot{\boldsymbol{x}} = \boldsymbol{Ax} + \boldsymbol{Gf}(\boldsymbol{x}) + \boldsymbol{Bu}$ $\boldsymbol{y} = \boldsymbol{Cx} + \boldsymbol{Du}$ **Bounded Jacobian:** $\underline{\boldsymbol{f}} \leq \mathrm{D}_x \boldsymbol{f}(\boldsymbol{x}) \leq \bar{\boldsymbol{f}}$ $\underline{b}_{ij} := \underline{\boldsymbol{f}}_{(i,j)}, \ \bar{b}_{ij} := \bar{\boldsymbol{f}}_{(i,j)}$	$\dot{\hat{\boldsymbol{x}}} = \boldsymbol{A\hat{x}} + \boldsymbol{Gf}(\hat{\boldsymbol{x}}) + \boldsymbol{Bu}$ $\quad + \boldsymbol{L}(\boldsymbol{y} - \hat{\boldsymbol{y}})$ $\hat{\boldsymbol{y}} = \boldsymbol{C\hat{x}} + \boldsymbol{Du}$ $\boldsymbol{L} = \boldsymbol{P}^{-1}\boldsymbol{Y}$ Variables: $\boldsymbol{P}, \boldsymbol{Y}, \boldsymbol{\Lambda}$ Constants: $\underline{c}_{ij} = \frac{1}{2}\left(\underline{b}_{ij} + \bar{b}_{ij}\right), \forall i, j$ $\bar{c}_{ij} = \frac{1}{2}\left(\underline{b}_{ij} - \bar{b}_{ij}\right), \forall i, j$ $\boldsymbol{W} = \boldsymbol{I}_g \otimes \boldsymbol{1}_{1 \times n_x}$	find $\boldsymbol{P} \succ 0, \boldsymbol{Y}, \boldsymbol{\Lambda}$ s.t. $\begin{bmatrix} \boldsymbol{A}^\top \boldsymbol{P} + \boldsymbol{PA} - \boldsymbol{YC} & & \\ -\boldsymbol{C}^\top \boldsymbol{Y}^\top + \boldsymbol{\Theta}_1(\boldsymbol{\Lambda}) & * & \\ \boldsymbol{W}^\top \boldsymbol{G}^\top \boldsymbol{P} + \boldsymbol{\Theta}_2(\boldsymbol{\Lambda}) & \boldsymbol{\Theta}_3(\boldsymbol{\Lambda}) \end{bmatrix} \prec 0$ $\boldsymbol{\Theta}_1(\boldsymbol{\Lambda}) = \mathrm{Diag}\left(\left\{\sum_{i=1}^n \lambda_{ij}\left(\bar{c}_{ij}^2 - \underline{c}_{ij}^2\right)\right\}_{j=1}^n\right)$ $\boldsymbol{\Theta}_2(\boldsymbol{\Lambda})^\top = \left[\left\{\mathrm{Diag}\left([\lambda_{i1}\underline{c}_{i1}, \ldots, \lambda_{in_x}\underline{c}_{in_x}]\right)\right\}_{i=1}^n\right]$ $\boldsymbol{\Theta}_3(\boldsymbol{\Lambda}) = \mathrm{Diag}\left(\mathrm{vec}(\boldsymbol{\Lambda})\right)$

Table VI
CONTROLLER AND OBSERVER-BASED STABILIZATION DESIGNS FOR NDS WITH DIFFERENT CLASSES OF NONLINEARITY THROUGH SDPS OR LMIS [13], [14], [61].

Dynamics and Nonlinearity Class	Control/Observer Structures	Control Design via LMI
$\dot{\boldsymbol{x}} = \boldsymbol{Ax} + \boldsymbol{Gf(x)} + \boldsymbol{Bu}$ **Quadratically Bounded:** $\langle \boldsymbol{f(x)}, \boldsymbol{f(x)} \rangle \leq \boldsymbol{x}^\top \boldsymbol{H}^\top \boldsymbol{Hx}$	$\boldsymbol{u} = \boldsymbol{Kx}$ $\boldsymbol{K} = \boldsymbol{LP}^{-1}$ Variables: $\boldsymbol{P}, \boldsymbol{L}, \gamma$	$\min\ \gamma$ s.t. $\begin{bmatrix} \boldsymbol{AP} + \boldsymbol{PA}^\top + \boldsymbol{BL} + \boldsymbol{L}^\top \boldsymbol{B}^\top & * & * \\ \boldsymbol{G}^\top & -\boldsymbol{I} & * \\ \boldsymbol{HP} & \boldsymbol{O} & -\gamma \boldsymbol{I} \end{bmatrix} \prec 0,$ $\boldsymbol{P} \succ 0, \gamma > 0$
$\dot{\boldsymbol{x}} = \boldsymbol{Ax} + \boldsymbol{f(x)} + \boldsymbol{Bu}$ **Glob./Loc. Lipschitz:** $\|\boldsymbol{f(x)} - f(\hat{\boldsymbol{x}})\|_2 \leq \gamma_l \|\boldsymbol{x} - \hat{\boldsymbol{x}}\|_2$	$\boldsymbol{u} = -\boldsymbol{Kx}$ $\boldsymbol{K} = \boldsymbol{WQ}^{-1}$ Variables: $\boldsymbol{Q}, \boldsymbol{W}$ Constant: $\rho > 0$	find $\boldsymbol{Q} \succ 0, \boldsymbol{W}$ s.t. $\begin{bmatrix} \boldsymbol{AQ} + \boldsymbol{QA}^\top - \boldsymbol{BW} - \boldsymbol{W}^\top \boldsymbol{B}^\top + \frac{1}{\rho} \boldsymbol{I} & * \\ \boldsymbol{Q} & -\frac{1}{\rho \gamma_l^2} \boldsymbol{I} \end{bmatrix} \prec 0,$
$\dot{\boldsymbol{x}} = \boldsymbol{Ax} + \boldsymbol{f(x)} + \boldsymbol{Bu}$ $\boldsymbol{y} = \boldsymbol{Cx}$ **One-Sided Lipschitz, Quadratically Bounded:** constants $\gamma_s, \gamma_{q1}, \gamma_{q2}$ (see Tab. I)	$\dot{\hat{\boldsymbol{x}}} = \boldsymbol{A\hat{x}} + \boldsymbol{f(\hat{x})} + \boldsymbol{Bu}$ $\quad + \boldsymbol{K(y - \hat{y})}$ $\hat{\boldsymbol{y}} = \boldsymbol{C\hat{x}}$ $\boldsymbol{u} = \boldsymbol{F\hat{x}}$ $\boldsymbol{K} = \boldsymbol{R}^{-1} \hat{\boldsymbol{K}}^\top$ $\boldsymbol{F} = \hat{\boldsymbol{F}}^\top \boldsymbol{Q}^{-1}$ Variables: $\boldsymbol{Q}, \boldsymbol{R}, \hat{\boldsymbol{K}}, \hat{\boldsymbol{F}},$ $\qquad \epsilon_1, \epsilon_2, \epsilon_3, \epsilon_4$ Constant: $\phi_1 > 0$	find $\boldsymbol{Q} \succ 0, \boldsymbol{R} \succ 0, \hat{\boldsymbol{K}}, \hat{\boldsymbol{F}}, \epsilon_1 > 0, \epsilon_2 > 0, \epsilon_3 > 0, \epsilon_4 > 0$ s.t. $\begin{bmatrix} \boldsymbol{S}_1 & * \\ \boldsymbol{S}_2^\top & \boldsymbol{S}_3 \end{bmatrix} \prec 0$ $\boldsymbol{S}_1 = \begin{bmatrix} \tilde{\boldsymbol{\Sigma}}_{11}(\boldsymbol{Q}, \hat{\boldsymbol{F}}) & * & * & * \\ \boldsymbol{O} & \tilde{\boldsymbol{\Sigma}}_{22}(\boldsymbol{R}, \hat{\boldsymbol{K}}, \epsilon_3, \epsilon_4, \gamma_s, \gamma_{q1}) & * & * \\ \boldsymbol{I} & \boldsymbol{O} & -2\epsilon_2 \boldsymbol{I} & * \\ \boldsymbol{O} & \boldsymbol{R} + (\epsilon_4 \gamma_{q2} - \epsilon_3) \boldsymbol{I} & \boldsymbol{O} & -2\epsilon_4 \boldsymbol{I} \end{bmatrix}$ $\boldsymbol{S}_2 = \begin{bmatrix} -\boldsymbol{B}\hat{\boldsymbol{F}}^\top & \boldsymbol{O} & \boldsymbol{Q} & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{I} & \boldsymbol{O} & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{O} & \boldsymbol{O} & \boldsymbol{I} \\ \boldsymbol{O} & \boldsymbol{O} & \boldsymbol{O} & \boldsymbol{O} \end{bmatrix}$ $\boldsymbol{S}_3 = \mathrm{Diag}\left(\begin{bmatrix} -\frac{\boldsymbol{Q}}{\phi_1} & -\phi_1 \boldsymbol{Q} & -\psi(\epsilon_1, \epsilon_2) \boldsymbol{I} & -\zeta(\epsilon_1, \epsilon_2) \boldsymbol{I} \end{bmatrix} \right)$