



Article

Unified CACSD Toolbox for Hybrid Simulation and Robust Controller Synthesis with Applications in DC-to-DC Power Converter Control

Mircea Șuşcă *, Vlad Mihaly *, Mihai Stănescu †, Dora Morar † and Petru Dobra †

Department of Automation, Technical University of Cluj-Napoca, Str. G. Barițiu nr. 26-28, 400027 Cluj-Napoca, Romania; mihai.stanescu@aut.utcluj.ro (M.S.); Dora.Sabau@aut.utcluj.ro (D.M.); Petru.Dobra@aut.utcluj.ro (P.D.)

* Correspondence: mircea.susca@aut.utcluj.ro (M.Ș.); vlad.mihaly@aut.utcluj.ro (V.M.)

† These authors contributed equally to this work.

Abstract: The current article presents the design, implementation, validation, and use of a Computer-Aided Control System Design (CACSD) toolbox for nonlinear and hybrid system uncertainty modeling, simulation, and control using μ synthesis. Remarkable features include generalization of classical system interconnection operations to nonlinear and hybrid systems, automatic computation of equilibrium points for nonlinear systems, and optimization of least conservative uncertainty bounds, with direct applicability for μ synthesis. A unified approach is presented for the step-down (buck), step-up (boost), and single-ended primary-inductor (SEPIC) converters to showcase the use and flexibility of the toolbox. Robust controllers were computed by minimization of the \mathcal{H}_∞ norm of the augmented performance systems, encompassing a wide range of uncertainty types, and have been designed using the well-known mixed-sensitivity closed loop shaping μ synthesis method.

Keywords: CACSD toolbox; operating point linearization; automatic uncertainty bound computation; Model-in-the-Loop simulation; hybrid simulation; robust control; \mathcal{H}_∞ control; μ synthesis; DC-to-DC power converters; buck; boost; SEPIC



Citation: Șuşcă, M.; Mihaly, V.; Stănescu, M.; Morar, D.; Dobra, P. Unified CACSD Toolbox for Hybrid Simulation and Robust Controller Synthesis with Applications in DC-to-DC Power Converter Control. *Mathematics* **2021**, *9*, 731. <https://doi.org/10.3390/math9070731>

Academic Editor: Aleksandr Rakhmangulov

Received: 28 February 2021

Accepted: 23 March 2021

Published: 28 March 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Robust Control represents a massive point of interest when it comes to Control Theory, which has been heavily studied over the past decades. However, albeit Robust Control brings many benefits, it is still an open field in research which gathers increasingly more approaches over time. Basically, the goal of a robust controller is to accomplish a specified set of performances for bounded model uncertainties which can occur in practice due to various reasons. In other words, closed loop stability and performance are maintained even for model parameter variations and unmodeled dynamics alike.

Over the years, multiple and various approaches for designing robust controllers have been presented, some of them being implemented into dedicated toolboxes, such as MATLAB's Robust Control Toolbox [1]. This toolbox gathers the most efficient ones based on \mathcal{H}_2 , \mathcal{H}_∞ , and μ synthesis methods, and it is often considered a reference in research. However, while using these types of toolboxes leads to controllers which are optimal for their prescribed criterion, they are not necessarily best in terms of conventional performances. Additionally, of great use for defining and optimizing difficult robust control problems is the Global Optimization Toolbox from in [2], providing ready-to-use solvers using various state-of-the-art algorithms, such as Particle Swarm Optimization (PSO) and Genetic Algorithms (GA). An important work in this direction, for computing optimal weighting functions for the generalized plant model, is presented in [3].

Even though there is a large variety of CACSD toolboxes in the field, their number is still expanding due to the necessity of overcoming drawbacks that the already existing

ones have. At this point, the purpose of new toolboxes is not only to determine robust controllers for a specific process class, but to use a unified approach that would make them work for more types of systems, even multiple interconnected systems, in various configurations. User experience is also more accentuated, which is why some of them incorporate graphical user interfaces (GUIs), for improved usability.

An example is Multivar, which is a MATLAB-based application used for multiple-input and multiple-output (MIMO) control design, presented in [4]. This toolbox supports two working modes. It allows the user to work both in function and GUI mode (which represents a configuration wizard for determining the controller). Multivar can be used for LTI systems with or without time delay and it allows creating a model; converting, approximating, and analyzing it; input–output pairing and decoupling; and controller design and evaluation. Besides this, the user is able to export the control design and compare it with other saved designs. Another GUI-based robust controller design tool, which was created in LabVIEW, is presented in [5], based on the \mathcal{H}_∞ loop shaping method. However, the goal was to provide a simple, user-friendly interface to make it easier to use, especially for educational purposes. Therefore, as mentioned by the authors, it does not provide the same flexibility as other design tools on the market.

LCToolbox, as presented in [6], is another MATLAB software package which is used for robust controller design. One of the advantages of using this toolbox instead of classic MATLAB routines is the fact that it gathers all necessary steps for controller design in one place, while cutting the need of preprocessing steps such as separate construction of the plant, and postprocessing steps, such as closed loop simulation. LCToolbox can be used for both linear time-invariant (LTI) and linear parameter-varying (LPV) models, and it also incorporates system identification methods. The controller is obtained by using the \mathcal{H}_∞ loop shaping method. Other \mathcal{H}_∞ -based CACSD toolboxes have been presented over the past years. One example is represented in [7], which is based on linearizing or convexifying the conventional non-convex constraints on the classical robustness margins of \mathcal{H}_∞ constraints. The controller parameters are then computed by using an optimization solver. This toolbox was created for MATLAB, and some of its main features are represented by the large variety of control problems in which it could be used, such as multi-model systems; the toolbox is designed to work with the output data of MATLAB's System Identification Toolbox [8]. The output of the toolbox is represented by a PID controller, which can be easily implemented. Another example of a \mathcal{H}_∞ -based CACSD toolbox is shown in [9], in which the main advantage is the reduced conservatism of almost all types of model uncertainties which are defined.

Controller order is an important factor when implementing it on real systems. Therefore, this might be an issue in some cases. However, methods that are determining a fixed structure controller are already presented, such as in [10], which is based on the \mathcal{H}_2 controller design method, but can be cumbersome to compute. In order to deal with the high order controller problem, other toolboxes include controller simplification steps to avoid the necessity of postprocessing, as presented in [11].

Currently acknowledged problems in this domain regard closed loop simulation, where performance validation is generally treated ad hoc, from one control problem to another. Another difficulty encountered is when the test cases were done only on the linearized system for which the controller is designed, without checking if the initially proposed performance values are additionally verified for the nonlinear plant model, and, also, uncertainty modeling is a very cumbersome operation. The purpose of the paper is to provide means for treating the previously stated problems in a unified manner, such as implementing automated testing, performance validation, and report generation.

In this current iteration of the toolbox, robust controllers were designed using the well-established routines from in [1]. The interface is scalable and the control logic and validation can be replaced with other user-defined methods, or the current robust control approach can be replaced with open source alternatives for the \mathcal{H}_2 and \mathcal{H}_∞ optimization problems, such as presented in the thesis [12], with the possibility to refine the necessary solutions

of the Algebraic Riccati Equations (AREs) using the algorithms from [13], while for the μ synthesis problem, the thesis [14] provides a flexible, open-source implementation using linear matrix inequalities (LMIs). A clear advantage over the ARE approach is that LMIs are capable of solving singular and close to singular problems. Alternatively, a mixed $\mathcal{H}_2/\mathcal{H}_\infty$ approach for stabilization and optimization using fixed-order controllers can be found in [15]. As such, the current iteration of the proposed toolbox is MATLAB-dependent for certain key functionalities, especially with regards to numerical simulation, robust control, and optimization, although the exposed ideas and mathematical framework can be directly implemented in other software environments, such as Python, Scilab, or LabVIEW.

The remainder of the paper is structured in the following manner. Section 2 describes the software structure and features of the proposed toolbox; Section 3 describes a proposed end-to-end workflow exemplified using modeling, control, validation, and simulation of several DC-to-DC converter topologies; and Section 4 illustrates comparative discussions, proposed improvements, and completions for future work and conclusions.

2. Toolbox Structure and Functionalities

The proposed toolbox has been designed with the target of end-to-end design and implementation of closed loop control systems, starting from the definition of the uncertainty set of plants to be controlled, their required operating point, along with control performance specifications and controller synthesis, and ending with the controller validation for the initial desired plant set.

2.1. Toolbox Features

Proposed features and advantages over existing toolboxes available in the literature:

- specify finite-dimensional dynamical systems with the general framework from Equation (1) to be used with the MATLAB ode framework; ability to interconnect such systems in series, parallel, and linear-fractional transformations; this functionality is described in Sections 2.2.1 and 2.2.2;
- specify hybrid dynamical systems in the framework from in [16] as in Equation (4), with the ability to interconnect such systems in series, parallel, and linear-fractional transformations, upper and lower; this feature is described in Section 2.2.3;
- automatically compute equilibrium points numerically, with the possibility to impose certain states, inputs, and/or outputs, while the remaining ones are deduced through numerical optimization; this feature is presented in Section 2.4;
- automatically compute the uncertainty model as requested alongside a nominal plant: additive, inverse additive, input and output multiplicative, etc. using a global optimization algorithm, such as particle swarm optimization, to be directly used as necessary for robust synthesis methods; removes the burden for the control engineer to manually do this process for each plant; this feature is presented in Section 2.5;
- flexible and scalable, all features are implemented through MATLAB code and does not need the use of Simulink, which can become cumbersome when treating families of plants and not a single, specific, plant at a time; also, to account for the operating point in the case of linearized, nonlinear, and hybrid systems, alike, the same interface for Model-in-the-Loop simulation is provided in the toolbox, as shown in Sections 2.2 and 2.3;
- besides the automatic validation of the frequency response for the desired operating point of the linearized plant family, the toolbox runs tests accounting for the uncertainty behavior of the desired nonlinear plant, not only on the linearization which the controller has been designed for. Every specification imposed in the designed phase will be automatically tested for the entire nonlinear system family, as illustrated in the case studies from Section 3.

2.2. Systems Specification

The scope of software classes implemented and described in this section aims to provide a flexible framework for simulation by using the ordinary-differential equation ode solver exclusively, with the low-level requirement of integrating a differential state equation. As such, exogenous signals would be reference signals and disturbances, known a priori in a simulation context. The intrinsic signals, i.e., commands and corresponding measurements, are passed to their corresponding subsystems by means of ode. Figure 1 encompasses an overview of the toolbox classes described in Sections 2.2–2.4. When the relationship between two classes is of type *inheritance*, the inherited class will not redundantly recall all previous properties and methods from the base class in the diagram, unless they overload the methods and is explicitly noted.

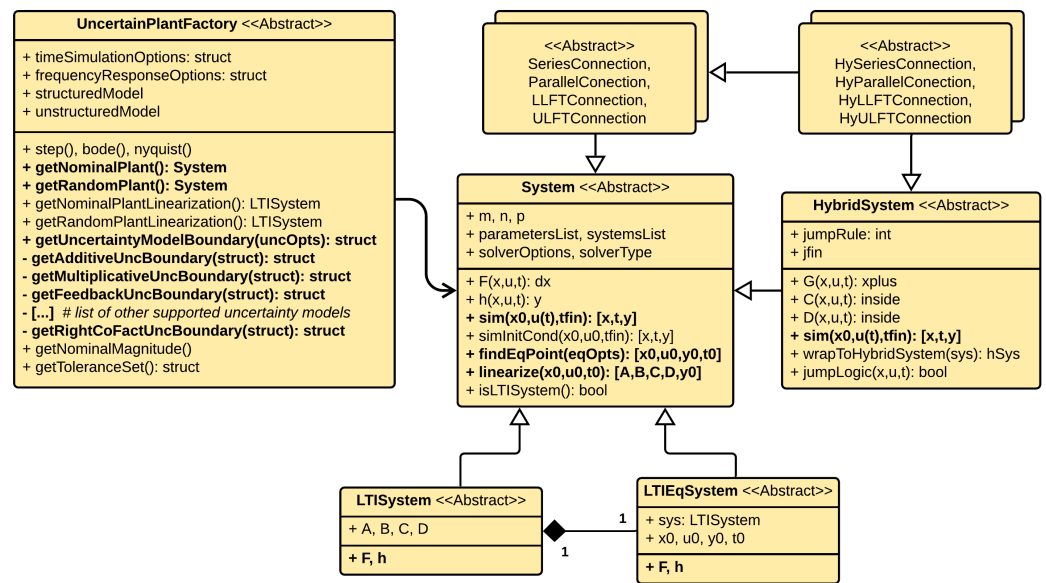


Figure 1. Class diagram for general-purpose nonlinear, LTI, linearized, and hybrid system implementations, along with the uncertain plant factory class, interconnections, and main functionalities.

2.2.1. Nonlinear Systems

For the purpose of this paper, we will focus on finite-dimensional systems: deterministic and stochastic. The so-called explicit or standard system form is obtained by writing the plant model in the following canonical form, using a set of differential equations and a set of output equations:

$$\begin{cases} \dot{\mathbf{x}}(t) = F(\mathbf{x}(t), \mathbf{u}(t), t); & (1a) \\ \mathbf{y}(t) = h(\mathbf{x}(t), \mathbf{u}(t), t), & (1b) \end{cases}$$

with the vector maps F and h being Lipschitz functions. The input signal $\mathbf{u}(t)$ has dimension m , state signal $\mathbf{x}(t)$ with dimension n , and output signal $\mathbf{y}(t)$ with dimension p , with $t \geq 0$. The initial conditions of the system are $\mathbf{x}(0) = \mathbf{x}_0 \in \mathbb{R}^n$.

Dynamical systems of the form (1) are implemented in class System. This will be the baseline interface for all systems the toolbox works with. Its most important methods are `sim`, `findEqPoint`, and `linearize`, which will be briefly described. The method `sim` simulates the dynamical system described by Equation (1a) from the initial condition \mathbf{x}_0 , using the exogenous signal $\mathbf{u}(t)$, which is a predetermined anonymous function with at least the input argument time. `tfIn` can be a scalar time value representing the final simulation time, a simulation interval, or a vector of predetermined time values. The solver options and type are based on MATLAB’s ode framework options and are sent directly to it. The solver type can be selected from any of the supported functions: `ode113`, `ode15s`, `ode15i`, `ode23`, `ode23t`, `ode23s`, `ode23tb`, or `ode45`. After integrating the state equation,

the output signal $y(t)$ can be directly computed using the memoryless function h from (1b). A useful particularization is also the method `simInitCond`, with the only difference being that it replaces the time-varying input signal $u(t)$ with a constant value u_0 , thus obtaining an impulse response. The method `findEqPoint` deduces an equilibrium point for the system given a set of specifications on the input, state, or output vectors and is described in detail in Section 2.4. After obtaining a valid equilibrium point, a linearized system can be obtained using the method `linearize`, also described there.

2.2.2. Linear Systems

Of particular interest for the framework and for control systems in general are linear time-invariant systems, which inherit the software interface from the `System` class, are implemented in the class `LTISystem` and are defined by

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t); & (2a) \\ y(t) = Cx(t) + Du(t). & (2b) \end{cases}$$

Separately, a nonlinear system can be linearized in the vicinity of an operating point, which is an equilibrium point for said system. The operating point (u_0, x_0, y_0, t_0) can be provided by the user or can be computed using the functionality from Section 2.4. The linearized system will work with variations of the initial variables and have the following model:

$$\begin{cases} \Delta \dot{x}(t) = A \cdot \Delta x(t) + B \cdot \Delta u(t); \\ \Delta y(t) = C \cdot \Delta x(t) + D \cdot \Delta u(t); \end{cases} \Leftrightarrow \begin{cases} \dot{x}(t) = A(x(t) - x_0) + B(u(t) - u_0); \\ y(t) = C(x(t) - x_0) + D(u(t) - u_0) + y_0. \end{cases} \quad (3)$$

This latter structure is useful for MiL simulations and is implemented in the auxiliary class `LTIEqSystem`, seen as an affine nonlinear system. The great advantage of having the system from Equation (3) readily available is that it is interchangeable with the initial nonlinear interface in a closed loop context without making further adaptations in the source code and can be used to study the performance degradation obtained by replacing the controller from the linearized system to the nonlinear plant.

2.2.3. Hybrid Systems

A useful extension of framework (1) for hybrid systems, to account for system discontinuities, is with structures described in [16,17]:

$$\begin{cases} \dot{x}(t) = F(x(t), u(t), t), & (x, u, t) \in \mathcal{C}; & (4a) \\ x^+(t) = G(x(t), u(t), t), & (x, u, t) \in \mathcal{D}; & (4b) \\ y(t) = h(x(t), u(t), t), & & (4c) \end{cases}$$

with $F : \mathbb{R}^{n+m+1} \rightarrow \mathbb{R}^n$ as the flow function, $G : \mathbb{R}^{n+m+1} \rightarrow \mathbb{R}^n$ the jump function, and $h : \mathbb{R}^{n+m+1} \rightarrow \mathbb{R}^p$ the output function, while $\mathcal{C} \subset \mathbb{R}^{n+m+1}$ represents the flow set and $\mathcal{D} \subset \mathbb{R}^{n+m+1}$ is the jump set. When executing an ode simulation, a jump condition trigger is permanently verified and, based on the selected configuration, it allows prioritizing the flow logic, the jump logic, or a stochastic behavior which includes randomly selecting any of them. This jump condition will also be needed for hybrid system interconnections.

We propose a separate class in the toolbox, called `HybridSystem`, which inherits the previously described class `System`, includes the ode event-based mechanism from `HyEQ Toolbox` [16], and is extended to support time-varying differential equation systems and exogenous input signals. Besides the base interface from `System`, it also provides methods for functions G , C , and \mathcal{D} . It also provides a wrapper function to promote any `System` object to the type `HybridSystem`, by adding dummy G , C , and \mathcal{D} methods, in order to be compatible for use in hybrid system interconnections. The flexibility added by this

class in the toolbox allows model-in-the-Loop simulations using physical processes with hybrid dynamics, such as switching systems, i.e., electrical machines and power converters, or simulations of the closed loop control system, seen as hybrid system through the interconnection of a continuous-time process and a discrete-time controller, allowing the user to assess several performance analysis steps.

2.3. System Interconnections

After defining individual or *atomic* systems as in previous sections, the necessity for composing system interconnections readily appears. The classical interconnection operations are the series, parallel, lower, and upper linear-fractional transformations (LLFT and ULFT). Moreover, two separate approaches have been considered, i.e., to interconnect general-purpose nonlinear systems modeled by the class `System` and hybrid systems modeled by the class `HybridSystem` separately. The first case is useful for linearization near an operating point, studying its system theoretical properties, and designing control techniques, while the latter becomes useful in a model-in-the-Loop simulation context and for closed loop system property analysis. All provided system interconnections are implemented in classes which inherit the base class `System`.

The software classes presented in this section extend the `series`, `parallel`, `feedback`, and `lft` functions from MATLAB for nonlinear and hybrid systems, based on the interfaces from Equations (1) and (4). For hybrid system interconnections, the continuous and discrete dynamics sets \mathcal{C} and \mathcal{D} , respectively, are obtained using union and intersection set operations.

Moreover, the next discrete state for each subsystem is triggered by its own logic, pre-defined in the jump function G and only when necessary; otherwise, it remains unchanged. For specifying this next discrete state \mathbf{x}^+ logic, as in the interface from Equation (4c), we will use the notation `IF(CONDITION, THEN, ELSE)`, where `CONDITION` will be true when the point in the state-space is in the jump set, i.e., $(\mathbf{x}, \mathbf{u}, t) \in \mathcal{D}$ or $(\mathbf{x}, \mathbf{u}, t) \notin \mathcal{C}$; `THEN` gives the next state if a jump needs to be performed; and `ELSE` gives the next discrete state otherwise.

The state, output, and hybrid domain equations for nonlinear and hybrid system series connection, with the notations used in Figure 2, upper row, implemented in classes `SeriesConnectionSystem` and `HybridSeriesConnectionSystem`, are as follows:

$$\left. \begin{aligned} \begin{bmatrix} \dot{\mathbf{x}}_1 \\ \dot{\mathbf{x}}_2 \end{bmatrix} &= \begin{bmatrix} F_1(\mathbf{x}_1, \mathbf{u}, t) \\ F_2(\mathbf{x}_2, h_1(\mathbf{x}_1, \mathbf{u}, t), t) \end{bmatrix}; \\ \mathbf{y} &= h_2(\mathbf{x}_2, h_1(\mathbf{x}_1, \mathbf{u}, t), t). \end{aligned} \right| \begin{aligned} \begin{bmatrix} \dot{\mathbf{x}}_1 \\ \dot{\mathbf{x}}_2 \end{bmatrix} &= \begin{bmatrix} F_1(\mathbf{x}_1, \mathbf{u}, t) \\ F_2(\mathbf{x}_2, h_1(\mathbf{x}_1, \mathbf{u}, t), t) \end{bmatrix}; \\ \begin{bmatrix} \mathbf{x}_1^+ \\ \mathbf{x}_2^+ \end{bmatrix} &= \begin{bmatrix} \text{if}(\text{jump}_1, G_1(\mathbf{x}_1, \mathbf{u}, t), \mathbf{x}_1) \\ \text{if}(\text{jump}_2, G_2(\mathbf{x}_2, h_1(\mathbf{x}_1, \mathbf{u}, t), t), \mathbf{x}_2) \end{bmatrix}; \\ \mathcal{C}(\mathbf{x}, \mathbf{u}, t) &= \mathcal{C}_1(\mathbf{x}_1, \mathbf{u}, t) \cap \mathcal{C}_2(\mathbf{x}_2, h_1(\mathbf{x}_1, \mathbf{u}, t), t); \\ \mathcal{D}(\mathbf{x}, \mathbf{u}, t) &= \mathcal{D}_1(\mathbf{x}_1, \mathbf{u}, t) \cup \mathcal{D}_2(\mathbf{x}_2, h_1(\mathbf{x}_1, \mathbf{u}, t), t); \\ \mathbf{y} &= h_2(\mathbf{x}_2, h_1(\mathbf{x}_1, \mathbf{u}, t), t). \end{aligned} \quad (5)$$

Given two initial subsystems `Sys1` and `Sys2` with dimensions (m_1, n_1, p_1) and (m_2, n_2, p_2) , respectively, the resulting series connection system will have dimensions $(m = m_1, n = n_1 + n_2, p = p_2)$.

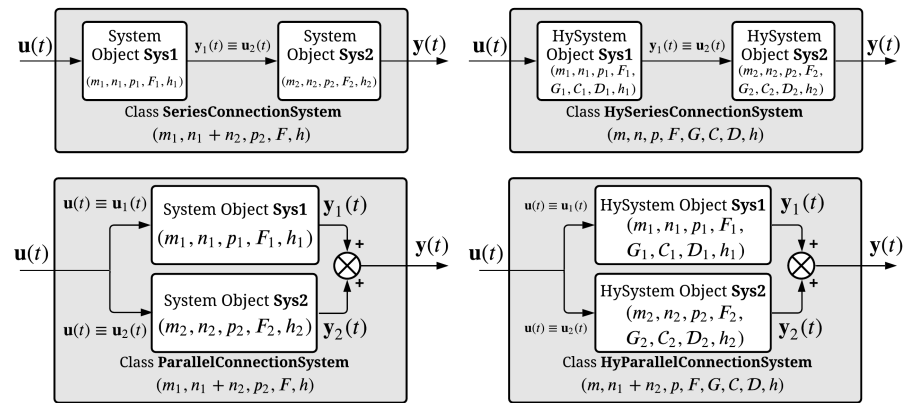


Figure 2. Series and parallel interconnections for general-purpose and hybrid systems.

The state, output, and hybrid domain equations for nonlinear and hybrid system parallel connection, with the notations used in Figure 2, bottom row, implemented in the classes named ParallelConnectionSystem and HybridParallelConnectionSystem, are as follows:

$$\left. \begin{aligned}
 \begin{bmatrix} \dot{\mathbf{x}}_1 \\ \dot{\mathbf{x}}_2 \end{bmatrix} &= \begin{bmatrix} F_1(\mathbf{x}_1, \mathbf{u}, t) \\ F_2(\mathbf{x}_2, \mathbf{u}, t) \end{bmatrix}; \\
 \mathbf{y} &= h_1(\mathbf{x}_1, \mathbf{u}, t) + h_2(\mathbf{x}_2, \mathbf{u}, t).
 \end{aligned}
 \right| \begin{aligned}
 \begin{bmatrix} \dot{\mathbf{x}}_1 \\ \dot{\mathbf{x}}_2 \end{bmatrix} &= \begin{bmatrix} F_1(\mathbf{x}_1, \mathbf{u}, t) \\ F_2(\mathbf{x}_2, \mathbf{u}, t) \end{bmatrix}; \\
 \begin{bmatrix} \mathbf{x}_1^+ \\ \mathbf{x}_2^+ \end{bmatrix} &= \begin{bmatrix} \text{if}(\text{jump}_1, G_1(\mathbf{x}_1, \mathbf{u}, t), \mathbf{x}_1) \\ \text{if}(\text{jump}_2, G_2(\mathbf{x}_2, \mathbf{u}, t), \mathbf{x}_2) \end{bmatrix}; \\
 \mathcal{C}(\mathbf{x}, \mathbf{u}, t) &= \mathcal{C}_1(\mathbf{x}_1, \mathbf{u}, t) \cap \mathcal{C}_2(\mathbf{x}_2, \mathbf{u}, t); \\
 \mathcal{D}(\mathbf{x}, \mathbf{u}, t) &= \mathcal{D}_1(\mathbf{x}_1, \mathbf{u}, t) \cup \mathcal{D}_2(\mathbf{x}_2, \mathbf{u}, t); \\
 \mathbf{y} &= h_1(\mathbf{x}_1, \mathbf{u}, t) + h_2(\mathbf{x}_2, \mathbf{u}, t).
 \end{aligned}
 \tag{6}$$

Given two initial subsystems Sys1 and Sys2 with dimensions (m_1, n_1, p_1) and (m_2, n_2, p_2) , respectively, the resulting parallel connection system will have dimensions $(m = m_1 = m_2, n = n_1 + n_2, p = p_1 = p_2)$.

The state, output, and hybrid domain equations for nonlinear and hybrid system lower linear fractional transformation (LLFT) connection, with the notations used in Figure 3, upper row, implemented in classes LLFTConnectionSystem and HybridLLFTConnectionSystem, are as follows:

$$\left. \begin{aligned}
 \begin{bmatrix} \dot{\mathbf{x}}_1 \\ \dot{\mathbf{x}}_2 \end{bmatrix} &= \begin{bmatrix} F_1(\mathbf{x}_1, \mathbf{u}_1^{\text{LLFT}}, t) \\ F_2(\mathbf{x}_2, \mathbf{u}_2^{\text{LLFT}}, t) \end{bmatrix}; \\
 \mathbf{y} &= \begin{bmatrix} h_1(\mathbf{x}_1, \mathbf{u}_1^{\text{LLFT}}, t) \\ h_2(\mathbf{x}_2, \mathbf{u}_2^{\text{LLFT}}, t) \end{bmatrix}.
 \end{aligned}
 \right| \begin{aligned}
 \begin{bmatrix} \dot{\mathbf{x}}_1 \\ \dot{\mathbf{x}}_2 \end{bmatrix} &= \begin{bmatrix} F_1(\mathbf{x}_1, \mathbf{u}_1^{\text{LLFT}}, t) \\ F_2(\mathbf{x}_2, \mathbf{u}_2^{\text{LLFT}}, t) \end{bmatrix}; \\
 \begin{bmatrix} \mathbf{x}_1^+ \\ \mathbf{x}_2^+ \end{bmatrix} &= \begin{bmatrix} \text{if}(\text{jump}_1, G_1(\mathbf{x}_1, \mathbf{u}_1^{\text{LLFT}}, t), \mathbf{x}_1) \\ \text{if}(\text{jump}_2, G_2(\mathbf{x}_2, \mathbf{u}_2^{\text{LLFT}}, t), \mathbf{x}_2) \end{bmatrix}; \\
 \mathcal{C}(\mathbf{x}, \mathbf{u}, t) &= \mathcal{C}_1(\mathbf{x}_1, \mathbf{u}_1^{\text{LLFT}}, t) \cap \mathcal{C}_2(\mathbf{x}_2, \mathbf{u}_2^{\text{LLFT}}, t); \\
 \mathcal{D}(\mathbf{x}, \mathbf{u}, t) &= \mathcal{D}_1(\mathbf{x}_1, \mathbf{u}_1^{\text{LLFT}}, t) \cup \mathcal{D}_2(\mathbf{x}_2, \mathbf{u}_2^{\text{LLFT}}, t); \\
 \mathbf{y} &= \begin{bmatrix} h_1(\mathbf{x}_1, \mathbf{u}_1^{\text{LLFT}}, t) \\ h_2(\mathbf{x}_2, \mathbf{u}_2^{\text{LLFT}}, t) \end{bmatrix},
 \end{aligned}
 \tag{7}$$

with the predefined notations:

$$\mathbf{u} = \begin{bmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{u}_{11} \\ \mathbf{u}_{12}^{\text{ref}} \\ \mathbf{u}_{21}^{\text{ref}} \\ \mathbf{u}_{22} \end{bmatrix}, \quad \mathbf{u}_1^{\text{LLFT}} = \begin{bmatrix} \mathbf{u}_{11} \\ \mathbf{u}_{12}^{\text{ref}} + \mathbf{y}_{21}^- \end{bmatrix} \equiv \begin{bmatrix} \mathbf{u}_{11} \\ \mathbf{u}_{12} \end{bmatrix}, \quad \mathbf{u}_2^{\text{LLFT}} = \begin{bmatrix} \mathbf{u}_{21}^{\text{ref}} + \mathbf{y}_{12}^- \\ \mathbf{u}_{22} \end{bmatrix} \equiv \begin{bmatrix} \mathbf{u}_{21} \\ \mathbf{u}_{22} \end{bmatrix}. \tag{8}$$

The common convention in the literature is to consider the last NCON values from the input vector \mathbf{u}_1 , i.e., \mathbf{u}_{12} , as control input signals, while the last NMEAS values from the output vector \mathbf{y}_1 , i.e., \mathbf{y}_{12} , as measurements signals. Only the vector \mathbf{u} will be an exogenous signal, as the feedback components \mathbf{y}_{12}^- and \mathbf{y}_{21}^- are local and private feedback components computed implicitly at the previous time step, dictated by the selected ode solver. The exogenous signals \mathbf{u}_{11} and \mathbf{u}_{22} are seen as disturbance signals, while the signals \mathbf{u}_{12}^{ref} and \mathbf{u}_{21}^{ref} are seen as reference signals.

Given two initial subsystems Sys1 and Sys2 with dimensions (m_1, n_1, p_1) and (m_2, n_2, p_2) , respectively, the resulting LLFT connection system will have dimensions $(m = m_1 + m_2, n = n_1 + n_2, p = p_1 + p_2)$. The subsystem Sys1 is usually seen as the controlled plant, while Sys2 is seen as the controller. In order to assure compatibility between the two, several assertions must be made: $NMEAS = \text{length}(\mathbf{y}_{12}) = \text{length}(\mathbf{u}_{21})$ and $NCON = \text{length}(\mathbf{y}_{21}) = \text{length}(\mathbf{u}_{12})$.

The state, output, and hybrid domain equations for nonlinear and hybrid system upper linear fractional transformation (ULFT) connection, with the notations used in Figure 3, bottom row, implemented in classes ULFTConnectionSystem and HybridULFTConnectionSystem, are as follows:

$$\left. \begin{aligned} \begin{bmatrix} \dot{\mathbf{x}}_1 \\ \dot{\mathbf{x}}_2 \end{bmatrix} &= \begin{bmatrix} F_1(\mathbf{x}_1, \mathbf{u}_1^{ULFT}, t) \\ F_2(\mathbf{x}_2, \mathbf{u}_2^{ULFT}, t) \end{bmatrix}; \\ \mathbf{y} &= \begin{bmatrix} h_1(\mathbf{x}_1, \mathbf{u}_1^{ULFT}, t) \\ h_2(\mathbf{x}_2, \mathbf{u}_2^{ULFT}, t) \end{bmatrix}. \end{aligned} \right\} \begin{aligned} \begin{bmatrix} \dot{\mathbf{x}}_1 \\ \dot{\mathbf{x}}_2 \end{bmatrix} &= \begin{bmatrix} F_1(\mathbf{x}_1, \mathbf{u}_1^{ULFT}, t) \\ F_2(\mathbf{x}_2, \mathbf{u}_2^{ULFT}, t) \end{bmatrix}; \\ \begin{bmatrix} \mathbf{x}_1^+ \\ \mathbf{x}_2^+ \end{bmatrix} &= \begin{bmatrix} \text{if}(\text{jump}_1, G_1(\mathbf{x}_1, \mathbf{u}_1^{ULFT}, t), \mathbf{x}_1) \\ \text{if}(\text{jump}_2, G_2(\mathbf{x}_2, \mathbf{u}_2^{ULFT}, t), \mathbf{x}_2) \end{bmatrix}; \\ \mathcal{C}(\mathbf{x}, \mathbf{u}, t) &= \mathcal{C}_1(\mathbf{x}_1, \mathbf{u}_1^{ULFT}, t) \cap \mathcal{C}_2(\mathbf{x}_2, \mathbf{u}_2^{ULFT}, t); \\ \mathcal{D}(\mathbf{x}, \mathbf{u}, t) &= \mathcal{D}_1(\mathbf{x}_1, \mathbf{u}_1^{ULFT}, t) \cup \mathcal{D}_2(\mathbf{x}_2, \mathbf{u}_2^{ULFT}, t); \\ \mathbf{y} &= \begin{bmatrix} h_1(\mathbf{x}_1, \mathbf{u}_1^{ULFT}, t) \\ h_2(\mathbf{x}_2, \mathbf{u}_2^{ULFT}, t) \end{bmatrix}, \end{aligned} \tag{9}$$

with the predefined notations:

$$\mathbf{u} = \mathbf{u}_{12}, \quad \mathbf{u}_1^{ULFT} = \begin{bmatrix} \mathbf{y}_2^- \\ \mathbf{u}_{12} \end{bmatrix}, \quad \mathbf{u}_2^{ULFT} = \mathbf{y}_{11}^-. \tag{10}$$

The common convention in the literature is to consider the first NU values from the input vector of the plant subsystem, i.e., \mathbf{u}_{11} , as input uncertainty signals, while the first NY values from the output vector of the plant, i.e., \mathbf{y}_{11} , as output uncertainty signals. Only the vector $\mathbf{u} \equiv \mathbf{u}_{12}$ will be an exogenous signal, as the feedback components \mathbf{y}_{11}^- and \mathbf{y}_2^- are local and private feedback components computed implicitly at the previous time step, dictated by the selected ode solver. The exogenous signal \mathbf{u}_{12} is seen as set of performance and control signals for the plant, without any reference signals recalled explicitly compared to the LLFT case.

Given two initial subsystems Sys1 and Sys2 with dimensions (m_1, n_1, p_1) and (m_2, n_2, p_2) , respectively, the resulting ULFT connection system will have dimensions $(m = m_1 + m_2, n = n_1 + n_2, p = p_1 + p_2)$. The subsystem Sys1 is usually seen as the augmented controlled plant, while Sys2 is seen as the unstructured uncertainty block. In order to assure compatibility between the two, several assertions must be made: $NY = \text{length}(\mathbf{y}_{11}) = \text{length}(\mathbf{u}_2)$ and $NU = \text{length}(\mathbf{y}_2) = \text{length}(\mathbf{u}_{11})$.

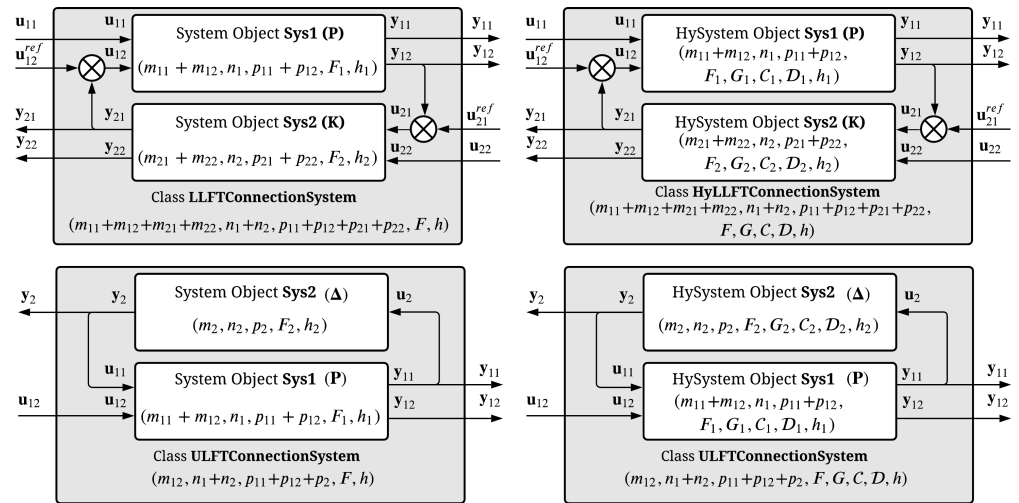


Figure 3. Upper (ULFT) and lower (LLFT) linear fractional transformation interconnections for general-purpose nonlinear and hybrid systems, with the ability to impose external reference signals.

2.4. Automatic Equilibrium Point Computation

Given a nonlinear system as in Equation (1), which may also include interconnections of systems, an operating point is desired with some of the input, state, and output variables imposed, such as a water level in a tank $y_h(t)$ controlled through two pumps $\mathbf{u}_{flow}(t)$, one with variable flow and one with a fixed flow, or a mechanical transportation system having a desired velocity $y_w(t)$ with respect to input forces and loads $\mathbf{u}(t)$. As such, a mechanism to automatically compute a partially imposed equilibrium point for an entire family of uncertain plants, relative to one which is considered nominal at the design phase, is proposed in this paragraph.

Starting from the system definition with dimensions m , n , and p , consider the sets of indexes, denoted by \mathcal{I} , and prescribed values, denoted by \mathcal{V} , for the input, state, and output variables, respectively:

$$\begin{cases} \mathcal{I}_u := \{i_1^u, i_2^u, \dots, i_{\bar{m}_u}^u\}, \mathcal{V}_u := \{\bar{\mathbf{u}}(i_1^u), \bar{\mathbf{u}}(i_2^u), \dots, \bar{\mathbf{u}}(i_{\bar{m}_u}^u)\}, & 0 \leq \bar{m}_u \leq m; \end{cases} \quad (11a)$$

$$\begin{cases} \mathcal{I}_x := \{i_1^x, i_2^x, \dots, i_{\bar{n}_x}^x\}, \mathcal{V}_x := \{\bar{\mathbf{x}}(i_1^x), \bar{\mathbf{x}}(i_2^x), \dots, \bar{\mathbf{x}}(i_{\bar{n}_x}^x)\}, & 0 \leq \bar{n}_x \leq n; \end{cases} \quad (11b)$$

$$\begin{cases} \mathcal{I}_y := \{i_1^y, i_2^y, \dots, i_{\bar{p}_y}^y\}, \mathcal{V}_y := \{\bar{\mathbf{y}}(i_1^y), \bar{\mathbf{y}}(i_2^y), \dots, \bar{\mathbf{y}}(i_{\bar{p}_y}^y)\}, & 0 \leq \bar{p}_y \leq p, \end{cases} \quad (11c)$$

along with their complementary sets of values for the indexes, denoted by \mathcal{UI} , and the values, denoted \mathcal{UV} , to be computed through optimization by solving a system of equations:

$$\begin{cases} \mathcal{UI}_u := \{i_1^u, i_2^u, \dots, i_{\tilde{m}_u}^u\}, \mathcal{UV}_u := \{\tilde{\mathbf{u}}(i_1^u), \tilde{\mathbf{u}}(i_2^u), \dots, \tilde{\mathbf{u}}(i_{\tilde{m}_u}^u)\}, & 0 \leq \tilde{m}_u \leq m; \end{cases} \quad (12a)$$

$$\begin{cases} \mathcal{UI}_x := \{i_1^x, i_2^x, \dots, i_{\tilde{n}_x}^x\}, \mathcal{UV}_x := \{\tilde{\mathbf{x}}(i_1^x), \tilde{\mathbf{x}}(i_2^x), \dots, \tilde{\mathbf{x}}(i_{\tilde{n}_x}^x)\}, & 0 \leq \tilde{n}_x \leq n; \end{cases} \quad (12b)$$

$$\begin{cases} \mathcal{UI}_y := \{i_1^y, i_2^y, \dots, i_{\tilde{p}_y}^y\}, \mathcal{UV}_y := \{\tilde{\mathbf{y}}(i_1^y), \tilde{\mathbf{y}}(i_2^y), \dots, \tilde{\mathbf{y}}(i_{\tilde{p}_y}^y)\}, & 0 \leq \tilde{p}_y \leq p, \end{cases} \quad (12c)$$

with

$$\begin{cases} \bar{m}_u + \tilde{m}_u = m, \mathcal{I}_u \cup \mathcal{UI}_u = \{1, 2, \dots, m\}, \mathcal{I}_u \cap \mathcal{UI}_u = \emptyset; \end{cases} \quad (13a)$$

$$\begin{cases} \bar{n}_x + \tilde{n}_x = n, \mathcal{I}_x \cup \mathcal{UI}_x = \{1, 2, \dots, n\}, \mathcal{I}_x \cap \mathcal{UI}_x = \emptyset; \end{cases} \quad (13b)$$

$$\begin{cases} \bar{p}_y + \tilde{p}_y = p, \mathcal{I}_y \cup \mathcal{UI}_y = \{1, 2, \dots, p\}, \mathcal{I}_y \cap \mathcal{UI}_y = \emptyset. \end{cases} \quad (13c)$$

a set of permutation matrices $P_u \in \mathbb{R}^{m \times m}$, $P_x \in \mathbb{R}^{n \times n}$, $P_y \in \mathbb{R}^{p \times p}$ are obtained after sorting the indexes such as the following system of vector-valued equations needs to be solved:

$$\begin{cases} \mathbf{0} = F\left(P_x \cdot \begin{bmatrix} \bar{\mathbf{x}} \\ \tilde{\mathbf{x}} \end{bmatrix}, P_u \cdot \begin{bmatrix} \bar{\mathbf{u}} \\ \tilde{\mathbf{u}} \end{bmatrix}, \tilde{t}\right); \\ P_y \cdot \begin{bmatrix} \bar{\mathbf{y}} \\ \tilde{\mathbf{y}} \end{bmatrix} = h\left(P_x \cdot \begin{bmatrix} \bar{\mathbf{x}} \\ \tilde{\mathbf{x}} \end{bmatrix}, P_u \cdot \begin{bmatrix} \bar{\mathbf{u}} \\ \tilde{\mathbf{u}} \end{bmatrix}, \tilde{t}\right). \end{cases} \tag{14a}$$

$$\tag{14b}$$

The system from Equation (14) becomes equivalent to directly solving a system of equations of the form $\mathbf{0} = \mathcal{F}(\mathbf{z})$ in the vector-valued unknown:

$$\mathbf{z} = [\tilde{\mathbf{x}}^T \quad \tilde{\mathbf{u}}^T \quad \tilde{\mathbf{y}}^T \quad \tilde{t}]^T \in \mathbb{R}^{\tilde{n}_x + \tilde{m}_u + \tilde{p}_y + 1}. \tag{15}$$

If the dynamical system is time-invariant or if the required time value is known a priori, then the time variable can be removed from the solver or it can be imposed to a certain value \bar{t} in the same manner as the for the other signals. Moreover, the method is flexible and allows imposing and solving only the subsystem (14a) if the output variables coincide with the states. The unknown variables from Equation (15) can be initialized to random values or a rough estimate for the entire family of uncertain plants can be obtained with the simulation to a step response of the nominal plant at the required amplitudes. All systems in the uncertain physical plant set will be found in the same mathematical vicinity. After solving the preferred algebraic system configuration from Equation (14), the desired equilibrium point $(\bar{\mathbf{u}}, \bar{\mathbf{x}}, \bar{\mathbf{y}}, \bar{t})$ can be reconstructed using the inverse permutation matrices P_u^{-1} , P_x^{-1} , P_y^{-1} and the notations from Equations (11) and (12). The method `findEqPoint` from class `System` forms and solves the system (14) and computes the desired equilibrium point for its predefined dynamical system based on the specifications from Equations (11) and (12) given in the structure `eq0pts`.

After acquiring the desired equilibrium point, the system linearization can be easily deduced through numeric differentiation methods. The most straightforward method is to compute the first-order Jacobian matrices of the functions F and h with respect to the state and input signals \mathbf{x} and \mathbf{u} , respectively:

$$A = \left. \frac{\delta F}{\delta \mathbf{x}} \right|_{(\mathbf{x}_0, \mathbf{u}_0, t_0)}; \quad B = \left. \frac{\delta F}{\delta \mathbf{u}} \right|_{(\mathbf{x}_0, \mathbf{u}_0, t_0)}; \quad C = \left. \frac{\delta h}{\delta \mathbf{x}} \right|_{(\mathbf{x}_0, \mathbf{u}_0, t_0)}; \quad D = \left. \frac{\delta h}{\delta \mathbf{u}} \right|_{(\mathbf{x}_0, \mathbf{u}_0, t_0)}. \tag{16}$$

The method `linearize`($\mathbf{x}_0, \mathbf{u}_0, t_0$) from class `System` of Section 1 computes the matrices from Equation (16) with a first-order derivative approximation and also the output equilibrium value $\mathbf{y}_0 = h(\mathbf{x}_0, \mathbf{u}_0, t_0)$:

$$A_{\overline{1},n,i} \approx \frac{F(\mathbf{x}_0 + \Delta \mathbf{x}_0^i, \mathbf{u}_0, t_0) - F(\mathbf{x}_0, \mathbf{u}_0, t_0)}{\Delta x}; \quad B_{\overline{1},m,j} \approx \frac{F(\mathbf{x}_0, \mathbf{u}_0 + \Delta \mathbf{u}_0^j, t_0) - F(\mathbf{x}_0, \mathbf{u}_0, t_0)}{\Delta u}, \tag{17}$$

following that the output matrices C and D to be computed in a similar manner by replacing F with h in the above formulas. The shorthand notations are $\Delta \mathbf{x}_0^i = [0 \ \dots \ 0 \ \Delta x \ 0 \ \dots \ 0]^T$ and $\Delta \mathbf{u}_0^j = [0 \ \dots \ 0 \ \Delta u \ 0 \ \dots \ 0]^T$ for the disturbance vectors corresponding to the state with the index $i \in \overline{1}, n$ or input with index $j \in \overline{1}, m$, respectively, while the optimal [18] unit perturbations are, using double precision, $\Delta x = \text{tol} \cdot (1 + \|\mathbf{x}_0\|)$ and $\Delta u = \text{tol} \cdot (1 + \|\mathbf{u}_0\|)$, $\text{tol} = 10^{-5}$. Obviously, when linearizing the system, the static amplification of the initial nonlinear system is not accounted in the procedure, but will not be relevant in the actual control design process and implementation due to the consideration of only Lipschitz function-based systems and, as such, it will be correctly compensated. The correct simulation of the linearized system near the operating point is done using the class `LTIEqSystem`.

2.5. Automatic Least Conservative Uncertainty Bound Computation

Figure 4 encompasses an overview of the toolbox classes described in Sections 2.5 and 2.6, along with showing their relationship with the classes from the previous sections. Based on any desirable combination of the above system classes, i.e., System, LTISystem, LTIEqSystem, HybridSystem, and their interconnections, we propose a new functionality to aid in uncertain system modeling, ready for use in augmenting the plant for μ synthesis, implemented in the classes UncertainPlantFactory, seen in Figure 1, along with UncertaintyBoundOptimizationProblem, seen in Figure 4.

The common uncertainty model types considered in practice are gathered in Table 1. Besides the definition of the uncertain plant $G(s)$ starting from a nominal model $G_n(s)$ in relation to the uncertainty block $\Delta(s)$, the mathematical expression of $\Delta(s)$ is necessary to experimentally deduce its frequency response. Left and right coprime factor uncertainties are described by two blocks: Δ_M and Δ_N , and one of them can be selected as a free term, i.e., one degree of freedom (1-DOF). The class UncertainPlantFactory provides an interface to define a nominal plant and a random plant from a prespecified set. Besides the methods getNominalPlant and getRandomPlant, both returning a System object, it implements each uncertainty type $\Delta(s)$ from Table 1, obtained through Monte Carlo simulation using the magnitude characteristic in the frequency domain.

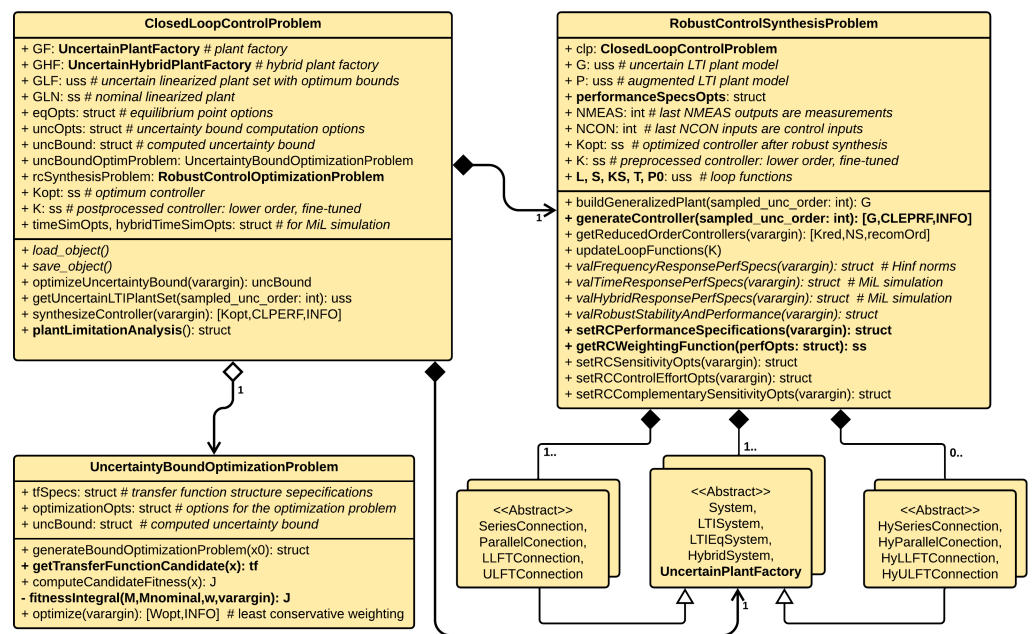


Figure 4. Closed loop control problem class which encompasses an uncertain plant set with operating point specification, options for automatically modeling the plant uncertainty, specifying robust control performances, synthesizing controller, and validating obtained results.

Table 1. Commonly used classes of perturbation and uncertainty models for multiple-input and multiple-output (MIMO) systems, implemented in class `UncertainPlantFactory`.

Uncertainty Type	Definition	Implementation
Additive	$G(s) = G_n(s) + \Delta(s)$	$\Delta(s) = G(s) - G_n(s)$
Inverse additive	$(G(s))^{-1} = (G_n(s))^{-1} + \Delta(s)$	$\Delta(s) = (G(s))^{-1} - (G_n(s))^{-1}$
Input multiplicative	$G(s) = G_n(s)[I + \Delta(s)]$	$\Delta(s) = (G_n(s))^{-1}(G(s) - G_n(s))$
Output multiplicative	$G(s) = [I + \Delta(s)]G_n(s)$	$\Delta(s) = (G(s) - G_n(s))(G_n(s))^{-1}$
Inverse input multiplicative	$(G(s))^{-1} = [I + \Delta(s)](G_n(s))^{-1}$	$\Delta(s) = (G(s))^{-1}(G_n(s)) - I$
Inverse output multiplicative	$(G(s))^{-1} = (G_n(s))^{-1}[I + \Delta(s)]$	$\Delta(s) = (G_n(s))(G(s))^{-1} - I$
Left coprime factor	$G(s) = (\tilde{M} + \Delta_{\tilde{M}})^{-1}(\tilde{N} + \Delta_{\tilde{N}})$	$\Delta_{\tilde{M}} = (\tilde{N} + \Delta_{\tilde{N}})(G(s))^{-1} - \tilde{M}$, 1-DOF
Right coprime factor	$G(s) = (N + \Delta_N)(M + \Delta_M)^{-1}$	$\Delta_M = (G(s))^{-1}(N + \Delta_N) - M$, 1-DOF

The frequency domain relevant for the studied plant is defined in logarithmic scale as

$$\Omega = \{\underline{\omega} = \omega_1 < \omega_2 < \dots < \omega_{N-1} < \omega_N = \bar{\omega}\}, \tag{18}$$

along with the magnitude characteristic values sampled at points from Ω . By convention, $\|\Delta(j\omega)\|_\infty \leq 1, \forall \omega \geq 0$; thus, the worst-case uncertainty deduced experimentally through Monte Carlo simulation is written as $\Delta(s)W_{exp}(s)$. As an example for the additive uncertainty type, the uncertain plant family will be $G(s) = G_n(s) + \Delta(s) \cdot W_{exp}(s)$, with $\|\Delta(j\omega)\|_\infty \leq 1, \forall \omega \geq 0$ and the toolbox returns the worst-case experimental magnitude characteristic of $W_{exp}(s)$, sampled at points from $j\Omega$ through the high-level method `getUncertaintyModelBoundary` from class `UncertainPlantFactory`, which wraps over low-level methods such as `getAdditiveUncBoundary`, `getInverseAdditiveUncBoundary` etc.

Due to the fact that the sampled points from the previous paragraph cannot be directly accounted for in robust control synthesis and, moreover, they may not represent an actual transfer function, it appears the need to compute a least conservative low-order transfer function to model the desired uncertainty family. This problem has been solved by employing a global optimization algorithm, such as PSO, described in [19]. The PSO algorithm has been considered in favor of other global optimization algorithms, such as GA, due to its inherent structure of addressing semi-continuous functions, such as our strongly nonlinear semi-continuous function described in Equation (21) in the variable from Equation (19).

A particle $\mathbf{x} := [x_1 \ x_2 \ \dots \ x_{n_4}]^T$ of the optimization problem is defined as the transfer function

$$W_x(s) = \frac{k}{s^p} \cdot \frac{\prod \left(\overset{\circ}{T}s + 1 \right) \prod \left(\frac{s^2}{\overset{\circ}{\omega}_n^2} + \frac{2\overset{\circ}{\zeta}}{\overset{\circ}{\omega}_n} s + 1 \right)}{\prod \left(\hat{T}s + 1 \right) \prod \left(\frac{s^2}{\hat{\omega}_n^2} + \frac{2\hat{\zeta}}{\hat{\omega}_n} s + 1 \right)} = \frac{x_1}{s^p} \cdot \frac{\prod_2^{n_1} (x_k s + 1) \prod_{n_1+1}^{n_2} \left(\frac{s^2}{x_k^2} + \frac{2x_{k+1}}{x_k} s + 1 \right)}{\prod_{n_2+1}^{n_3} (x_k s + 1) \prod_{n_3+1}^{n_4} \left(\frac{s^2}{x_k^2} + \frac{2x_{k+1}}{x_k} s + 1 \right)}, \tag{19}$$

with optimization variables

$$k > 0; \left\{ \overset{\circ}{T}_i \in \left[\frac{1}{\underline{\omega}}, \frac{1}{\bar{\omega}} \right] \right\}, i \in \overline{2 : n_1}; \left\{ \hat{T}_i \in \left[\frac{1}{\bar{\omega}}, \frac{1}{\underline{\omega}} \right] \right\}, i \in \overline{n_2 + 1 : n_3}; \tag{20a}$$

$$\left\{ \left(\overset{\circ}{\zeta}_i \in (0, 1), \overset{\circ}{\omega}_{n,i+1} \in [\underline{\omega}, \bar{\omega}] \right) \right\}, i \in \overline{n_1 + 1 : 2 : n_2}; \tag{20b}$$

$$\left\{ \left(\hat{\zeta}_i \in (0, 1), \hat{\omega}_{n,i+1} \in [\underline{\omega}, \bar{\omega}] \right) \right\}, i \in \overline{n_3 + 1 : 2 : n_4}. \tag{20c}$$

The cost functional found to provide good results for most initial particle swarms is

$$J_{\Omega, W_{exp}}(W_x) = \sum_{k=1}^N \left| |W_{exp}(j\omega_k)|^{dB} - |W_x(j\omega_k)|^{dB} \right| \cdot \varphi_N(k) + \sum_{|W_x(j\omega_k)| < |W_{exp}(j\omega_k)|} \lambda, \quad (21)$$

where $\varphi_N : \{1, 2, \dots, N\} \rightarrow \mathbb{R}^+$ from the first sum is a windowing function, based on the Gaussian window function, meant to amplify the penalization for low- and high-frequency components, while the second sum adds a high-cost term $\lambda = 10^9$ for each frequency point $j\omega_k$ for which the candidate function W_x is below the experimental reference uncertainty weight W_{exp} . The windowing function considered is defined by

$$\varphi_N(k) = \left(3 - 2 \cdot w_N \left(k - \frac{N+1}{2} \right) \right)^2, \quad (22)$$

with the Gaussian function $w_N : \mathbb{R} \rightarrow \mathbb{R}^+$ and value $\alpha = 2.5$:

$$w_N(x) = e^{-\frac{1}{2} \left(\alpha \frac{x}{(N-1)/2} \right)^2} = e^{-\frac{x^2}{2\sigma^2}} \Leftrightarrow \log w_N(x) = -\frac{1}{2} \left(\alpha \frac{x}{(N-1)/2} \right)^2 = -\frac{x^2}{2\sigma^2}. \quad (23)$$

The cost functional $J(W_x)$ is configurable, such that a different windowing function may be provided, or that the difference between W_{exp} and W_x in the integral may be considered in absolute magnitude values instead of decibels. An alternative formulation of this minimization problem would be to use a GA, where the optimization variable $\mathbf{x} = W_x(s)$ could mutate in order to obtain different transfer function structures: add or remove real poles and zeros or, also, complex conjugate pole and zero pairs.

This functionality is implemented in the class `UncertaintyBoundOptimizationProblem`, using the methods `getTransferFunctionCandidate` for Equation (19), `computeCandidateFitness` and `fitnessIntegral` for Equation (21) based on the magnitudes of $|W_{exp}(j\omega)|$ and $|W_x(j\omega)|$ for $\omega \in \Omega$ from Equation (18) and `optimize` to use the PSO algorithm to compute the best candidate transfer function $W_{x,optim}(s)$. The plot function has been overloaded to facilitate seeing the fitness function values in real-time and, moreover, the Bode plot for the best candidate $W_x(s)$ compared to the experimental data $W_{exp}(s)$.

2.6. Robust Synthesis and Closed Loop Validation

Classical solutions to the $\mathcal{H}_2/\mathcal{H}_\infty$ problems are presented in [20–24] and others. The control problem is typically formulated for the nominal plant using the generalized framework depicted in Figure 5a.

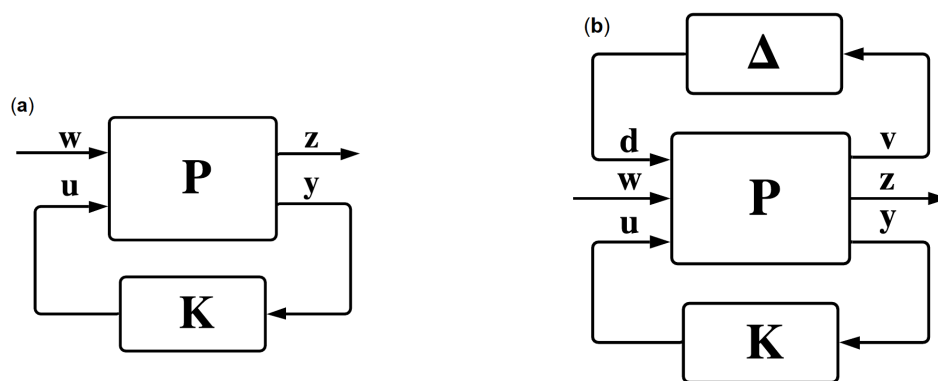


Figure 5. (a) Generalized plant framework; (b) generalized plant framework with uncertainties.

This generalized plant is obtained by augmenting the physical process model with a set of mathematical signals which aid the optimization procedure and has the following structure:

$$\begin{pmatrix} \mathbf{z}(t) \\ \mathbf{y}(t) \end{pmatrix} = \begin{pmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{pmatrix} \begin{pmatrix} \mathbf{w}(t) \\ \mathbf{u}(t) \end{pmatrix}, \quad \text{with} \quad \begin{pmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{pmatrix} = \left(\begin{array}{c|cc} A & B_1 & B_2 \\ \hline C_1 & D_{11} & D_{12} \\ C_2 & D_{21} & D_{22} \end{array} \right), \quad (24)$$

where $\mathbf{w} \in \mathbb{R}^{n_w}$ is the exogenous input vector, $\mathbf{u} \in \mathbb{R}^{n_u}$ is the control input vector, $\mathbf{z} \in \mathbb{R}^{n_z}$ is the error (output, performance) vector, and $\mathbf{y} \in \mathbb{R}^{n_y}$ is the measurement vector. The closed loop system is given by the LLFT interconnection of P and K :

$$P_{zw} = \text{LLFT}(P, K) = P_{11} + P_{12}K(I - P_{22}K)^{-1}P_{21}. \quad (25)$$

For the nominal plant, the target of the robust control problem is to minimize the \mathcal{H}_∞ norm using a stabilizing controller K , which can be written as

$$\min_{K \text{ stab.}} \|P_{zw}\|_\infty = \min_{K \text{ stab.}} \sup_{\omega \in \mathbb{R}^+} \bar{\sigma}(P_{zw}(j\omega)), \quad (26)$$

obtaining a (sub)optimal value γ by iteration, which minimizes the effects of the input vector $\mathbf{w}(t)$ as seen through the performance output vector $\mathbf{z}(t)$.

However, this problem ensures only nominal stability and nominal performance. However, the plant is a model of a physical process, having uncertainties. There are two types of uncertainties: unstructured, which illustrates neglected and unmodelled dynamics and which are represented by a full block $\Delta \in \mathbb{R}^{m \times m}$, and parametric, which are represented by δI , where δ is the maximum bound of the variable parameter. In a mixed-scenario, the following set is considered:

$$\Delta = \left\{ \Delta = \text{diag}(\delta_1 I_{n_1}, \dots, \delta_s I_{n_s}, \Delta_1, \dots, \Delta_f) \mid \delta_k \in \mathbb{R}, \Delta_j \in \mathbb{R}^{m_j \times m_j}, k = \overline{1, s}, j = \overline{1, f} \right\}. \quad (27)$$

In Figure 5b, the closed loop system containing a LLFT connection between plant P and controller K and an ULFT connection between plant P and uncertainty block Δ is presented. In this case, the generalized plant contains one extra input vector, i.e., disturbance inputs $\mathbf{d} \in \mathbb{R}^{n_d}$, and one extra output vector, i.e., disturbance outputs $\mathbf{v} \in \mathbb{R}^{n_v}$, giving the following structure:

$$P_\Delta(s) = \begin{pmatrix} P_{vd}(s) & P_{vw}(s) & P_{vu}(s) \\ P_{zd}(s) & P_{zw}(s) & P_{zu}(s) \\ P_{yd}(s) & P_{yw}(s) & P_{yu}(s) \end{pmatrix} \Leftrightarrow P_\Delta : \begin{pmatrix} \dot{\mathbf{x}}(t) \\ \mathbf{v}(t) \\ \mathbf{z}(t) \\ \mathbf{y}(t) \end{pmatrix} = \left(\begin{array}{c|ccc} A & B_d & B_w & B_u \\ \hline C_v & D_{vd} & D_{vw} & D_{vu} \\ C_z & D_{zd} & D_{zw} & D_{zu} \\ C_y & D_{yd} & D_{yw} & D_{yu} \end{array} \right) \begin{pmatrix} \mathbf{x}(t) \\ \mathbf{d}(t) \\ \mathbf{w}(t) \\ \mathbf{u}(t) \end{pmatrix}. \quad (28)$$

A mathematical tool used for studying the robustness is the *structured singular value*, defined for a square matrix $M \in \mathbb{C}^{N \times N}$ with respect to the set Δ as

$$\mu_\Delta(M) = \frac{1}{\min_{\Delta \in \Delta} \{ \bar{\sigma}(\Delta) \mid \det(I - M\Delta) = 0 \}}, \quad (29)$$

if there exists $\Delta \in \Delta$ such that the matrix $I - M\Delta$ is rank deficient; otherwise, it is 0. For the system presented in Figure 5b, the structured singular value of $\text{LLFT}(P, K)$, according to Δ , can be defined as

$$\mu_\Delta(\text{LLFT}(P, K)(s)) = \sup_{\omega \in \mathbb{R}^+} \mu_\Delta(\text{LLFT}(P, K)(j\omega)). \quad (30)$$

Besides the classical $\mathcal{H}_2/\mathcal{H}_\infty$ techniques, the μ synthesis framework manages to design a controller that meets the robust stability and robust performance specifications. The robust stability implies that a certain controller manages to stabilize all processes described by the upper linear fractional transformation between plant and uncertainty block, while the robust performance means that the controller is able to impose the desired closed loop performance in the worst-case scenario. Based on the main loop theorem, a controller K meets the robust stability and robust performance if and only if the structural singular value of the lower linear fractional transformation with respect to Δ is smaller than 1. Therefore, the minimization problem can be written as

$$\inf_{K \text{ stab.}} \sup_{\omega \in \mathbb{R}_+} \mu_\Delta(\text{LLFT}(P, K)(j\omega)), \tag{31}$$

which is not a convex problem. Additionally, the structural singular values are difficult to be explicitly computed. In order to solve this problem, the following upper bound is used [25]:

$$\mu_\Delta(\text{LLFT}(P, K)(j\omega)) \leq \inf_{D \in \mathcal{D}} \bar{\sigma}(D \cdot \text{LLFT}(P, K)(j\omega) \cdot D^{-1}), \tag{32}$$

where the set \mathcal{D} is defined in relation to the uncertainty set Δ as

$$\mathcal{D} = \left\{ \text{diag}\left(D_1, \dots, D_s, d_1 I_{m_1}, \dots, d_f I_{m_f}\right) \mid D_k = D_k^\top \in \mathbb{R}^{n_k \times n_k}, d_j > 0, k = \overline{1, s}, j = \overline{1, f} \right\}. \tag{33}$$

Now, using this bound, the solution of the initial non-convex problem can be practically approximated by solving the following quasi-convex problem:

$$\inf_{K \text{ stab.}} \sup_{\omega \in \mathbb{R}_+} \inf_{D \in \mathcal{D}} \bar{\sigma}\left(D(j\omega) \cdot \text{LLFT}(P, K)(j\omega) \cdot (D(j\omega))^{-1}\right). \tag{34}$$

Finally, if the system D is fixed, the problem (34) is nothing but a \mathcal{H}_∞ control problem, in this case called the K step. Furthermore, for a fixed controller K , the D scale step can be obtained by solving a Parrot problem for a desired set of frequencies $\Omega = \{\omega_1, \dots, \omega_N\}$ using a LMI and then obtain a minimum phase system after performing an identification step. Using these, an iterative algorithm, based on alternative D - K iterations, manages to solve the μ synthesis problem. This procedure starts with $D = I$ and successively applies a K step and a D scaling step until a stopping criterion is reached.

Of great use in the controller design phase are the sensitivity, complementary sensitivity, and control effort functions, respectively, defined by

$$S := (I + GK)^{-1}; \tag{35a}$$

$$T := GK(I + GK)^{-1}; \tag{35b}$$

$$R := K(I + GK)^{-1} = KS, \tag{35c}$$

where G is the open loop model. The great advantage of considering this approach is that it allows sculpting the relevant loop functions to impose steady-state and transitory regime performances, which are specified for different frequency ranges, using adequately selected weighting functions. Besides the minimization from Equation (34), different constraints can be added to the optimization problem to obtain a compromise between S , KS , and T at various frequencies:

$$\inf_{K \text{ stab.}} \sup_{\omega \in \mathbb{R}_+} \inf_{D \in \mathcal{D}} \bar{\sigma}\left(D(j\omega) \cdot \text{LLFT}(P, K)(j\omega) \cdot (D(j\omega))^{-1}\right), \tag{36a}$$

$$\text{such that } \left\| \begin{pmatrix} W_S S & W_T T & W_{KS} KS \end{pmatrix}^T \right\|_\infty < 1, \tag{36b}$$

also known as the mixed-sensitivity closed loop shaping μ synthesis method. The approach considered in this first iteration of the toolbox uses closed loop shaping with μ synthesis for the controller. Using the work in [26] as a starting point, different frequency response specifications, directly correlated with desired time-response performances, can be imposed in the weighting functions. The sensitivity weighting function W_S depends on four parameters as in

$$W_S(s) = \left(\frac{\frac{1}{M^{1/n}}s + \omega_B}{s + \omega_B A^{1/n}} \right)^n, \tag{37}$$

where ω_B represents the imposed bandwidth of the system; M imposes the \mathcal{H}_∞ norm of the sensitivity function, in order to limit the overshoot of the system; n imposes the slope of the sensitivity function for low frequencies; and A imposes the maximum allowed steady-state error.

On the other hand, the complementary sensitivity weighting function W_T can be generally defined by the following structure, in a *symmetrical* manner compared to W_S :

$$W_T(s) = \left(\frac{s + \omega_{BT}}{A_T^{1/n}s + \omega_{BT}M_T^{1/n}} \right)^n, \tag{38}$$

with ω_{BT} being the imposed bandwidth of the system; M_T imposes the \mathcal{H}_∞ norm of $T(s)$; n imposes the roll-off slope of the closed loop system, which should be directly coupled with sensor noise characteristics; and A_T imposes the least required attenuation for high frequencies. In practice, the complementary sensitivity bandwidth ω_{BT} can be adapted to the characteristics of the sensor in order to account for high-frequency noise.

Finally, the control effort weighting function with desired specifications $M_0 := |W_{KS}(0)|$, $M_\infty := |W_{KS}(\infty)|$ and $|W_{KS}(j \cdot \omega_d)| = M_d$, $M_0 < M_d < M_\infty$ can be synthesized by the following formula:

$$W_{KS}(s) = \frac{M_\infty s + M_0 \omega_d \sqrt{\frac{M_\infty^2 - M_d^2}{M_d^2 - M_0^2}}}{s + \omega_d \sqrt{\frac{M_\infty^2 - M_d^2}{M_d^2 - M_0^2}}}. \tag{39}$$

A higher-order counterpart can be generalized as for the previous cases, but was not found necessary for the proposed case studies and other tested benchmark plants.

Class `RobustControlSynthesisProblem` uses the nominal linearized plant around a required operating point using the system (14) and options (11)–(13), with a specified uncertainty type from Table 1, modeled through class `UncertaintyBoundOptimizationProblem`, allows imposing closed loop performance specifications with frequency weights (37)–(39), and synthesizes controller solutions for the problem (36) which cover the robust stability and robust performance problems. Additionally, the class also allows controller postprocessing, using order-reduction methods to compute easily implementable controllers. The aforementioned controller synthesis problem is illustrated in Figure 6, while the resulting LTI controller can be used in a MiL simulation context using the interface from class `LTI EqSystem`, encompassing system (3).

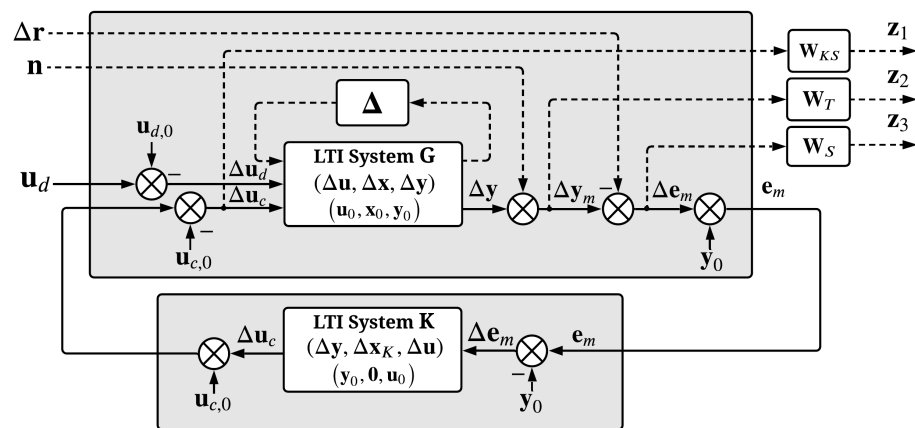


Figure 6. Robust controller synthesis diagram for the plant G , with the uncertainty set Δ determined a priori using the functionality from Section 2.5, linearized at the operating point $(\mathbf{u}_0, \mathbf{x}_0, \mathbf{y}_0)$; by convention, control inputs $\Delta \mathbf{u}_c$ of the linearized plant G are indexed after disturbance inputs $\Delta \mathbf{u}_d$.

Class `ClosedLoopControlProblem` gathers all data, options, computations, and results from the previously presented individual blocks. It is used to define the end-to-end control problem by aggregating the uncertain plant family with its corresponding least conservative bound optimization; the robust control synthesis procedure and metadata; and allows Model-in-the-Loop simulations with the nonlinear, linearized, and hybrid system models, with automatic validation of imposed performances for the nonlinear plant in the frequency and time-domain alike.

3. Results

To showcase the ease of use and functionalities of the proposed toolbox, a set of case studies will be illustrated for DC-to-DC power converter circuits in a unified manner to encompass modeling, control synthesis and performance validation, considering the topologies buck, boost, and single-ended primary-inductor converter (SEPIC). DC-to-DC converters have been considered as a case study due to their ubiquity in various practical domains and applications, as presented in [27], ranging from renewable energy, hybrid and electric vehicles, controlled power sources, and many more. They can be seen as a good benchmark for control systems, due to their switching behavior, nonlinear dynamics, and different tried and tested control methods, such as robust techniques in [28], Lyapunov methods in [29], passivity theory in [30], or sliding mode control as in [31].

This section will be split in a subsection which presents the converter mathematical models, a subsection with a suggested workflow for a general purpose control problem, followed by a subsection with numerical results and simulations for each of the studied converter topologies. For the SEPIC converter, having the most highly nonlinear behavior, thus being more difficult to control, we will illustrate and detail all plots generated by the toolbox, while for the buck and boost circuits, for brevity, we will show only the relevant figures and maintain the mathematical results and discussions.

3.1. Mathematical Modeling

The nonideal step-down (buck), step-up (boost), and single-ended primary-inductor converter (SEPIC) circuits are presented in Figure 7, where each component is described as follows.

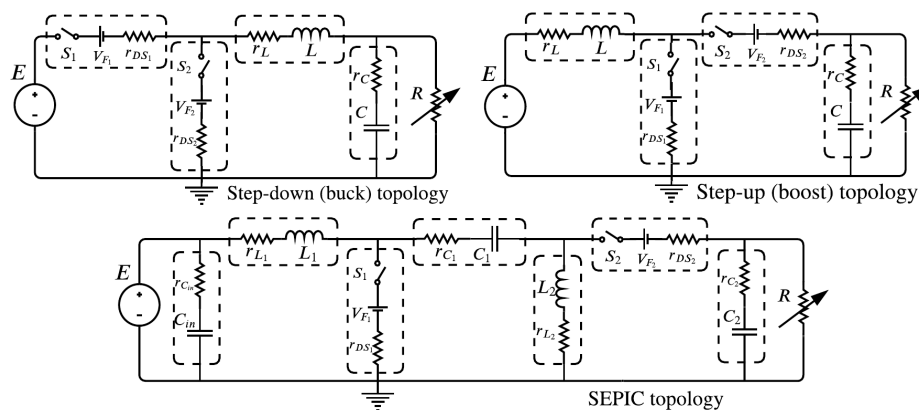


Figure 7. DC-to-DC power converter circuit topologies considered in the case studies: step-down (buck), step-up (boost), and single-ended primary-inductor (SEPIC).

- S_1 : switching device, usually a transistor, and S_2 : switching device, usually a diode or transistor;
- L, L_1, L_2 : converter inductors;
- C, C_{in}, C_1, C_2 : converter capacitors;
- R : (variable) output load resistance;
- E : external source voltage;
- r_L, r_{L1}, r_{L2} : resistances associated with the inductors;
- $r_C, r_{C_{in}}, r_{C1}, r_{C2}$: capacitors parasitic resistances;
- r_{DS1}, r_{DS2} : resistances associated with the ON state of the switching devices (usually drain source);
- V_{F1}, V_{F2} : constant voltage drops associated with the conducting phase of S_1 and S_2 ;
- $\mu \in [0, 1]$: normalized duty cycle applied to S_1 ; complementary to the PWM signal applied to S_2 .

Each converter has one control switching device S_1 , while the other one, S_2 , will be complementary to the former. Although, typically S_2 is a diode, it is preferable to use two encapsulated transistors for S_1 and S_2 . When working in continuous conduction mode (CCM), all converters will have an ON state, corresponding to S_1 being on and S_2 off, along with an OFF state, for its complementary behavior. The corresponding LTI models, for a constant load resistance R , for the ON and OFF states will be presented using the following structure with the external voltage seen as disturbance input $u(t) = E(t)$, the voltage drops V_{F1} and V_{F2} as constant DC inputs, states from the inductor currents and capacitor voltages, and the load resistor voltage as measured output:

$$\left(\begin{array}{c|c} A & \bar{B} \\ \hline C & \bar{D} \end{array} \right) = \left(\begin{array}{c|c|c} A & B & B_V \\ \hline C & D & D_V \end{array} \right), \tag{40}$$

where:

$$\begin{cases} \dot{\mathbf{x}} = A_{ON}\mathbf{x} + B_{ON}E + B_{V,ON}[V_{F1} V_{F2}]; \\ \mathbf{y} = C_{ON}\mathbf{x} + D_{ON}E + D_{V,ON}[V_{F1} V_{F2}], \end{cases} \quad \begin{cases} \dot{\mathbf{x}} = A_{OFF}\mathbf{x} + B_{OFF}E + B_{V,OFF}[V_{F1} V_{F2}]; \\ \mathbf{y} = C_{OFF}\mathbf{x} + D_{OFF}E + D_{V,OFF}[V_{F1} V_{F2}]. \end{cases} \tag{41}$$

The control variable is the duty cycle of the switching devices $\mu(t) \in [0, 1]$. Using a convex combination of the ON and OFF equation systems from Equation (41), an averaged state-space nonlinear model of the process is obtained close to the hybrid model’s behavior given a sufficiently high PWM frequency:

$$\dot{\mathbf{x}}(t) = \mu(t) \cdot \mathbf{x}_{ON}(t) + (1 - \mu(t)) \cdot \mathbf{x}_{OFF}(t) \equiv F(\mathbf{x}(t), [E(t), R(t), \mu(t)]^T, t). \tag{42}$$

As such, an affine nonlinear system, with respect to μ , with the state function F as above can be implemented by inheriting the class `System` from the toolbox. The disturbances which affect the system are the voltage source E and variable output load R , which are stochastic in nature, along with uncertainties of its components due to manufacturing tolerances, relevant on inductors and capacitors. As the toolbox easily allows using the output capacitor voltage or output load voltage with minor modifications, we used the resistor voltage as measurement variable due to its corresponding practical control use cases. By inheriting the class `UncertainPlantFactory`, a set of tolerances can be imposed on all relevant circuit parameters and, also, an LTI uncertain set can be automatically computed with the provided mechanisms. The equilibrium point will have only the steady-state values of \bar{E} , \bar{R} , and \bar{u}_R imposed, with the following structure:

$$(\bar{\mathbf{u}}, \bar{\mathbf{x}}, \bar{\mathbf{y}}) = ([\bar{E}, \bar{R}, \bar{\mu}], [\bar{i}_{L_1}, \bar{u}_{C_1}, \dots], \bar{u}_R). \tag{43}$$

After synthesizing a robust controller, model-in-the-loop simulations would be desired for the averaged state-space models and, also, for a hybrid model description of the converters. For the hybrid approach, the class `UncertainPlantFactory` is inherited again, this time with individual plants of type `HybridSystem`. For the general approach for CCM, based on the structure from Equation (4), the input vector is comprised of $\mathbf{u} = [E, R, \mu]^T$, the state vector is extended to $\mathbf{x} := [\mathbf{z}^T, q, \tau]^T$, with $\mathbf{z} = [i_{L_1}, u_{C_1}, \dots]^T$ being the physical continuous states; $q \in \{0, 1\}$ the discrete state number, i.e., *ON* and *OFF*; and $\tau \in [0, T_{PWM})$ the time values for a single PWM period. After each PWM period completion, the auxiliary time state τ is reinitialized to 0. The model description becomes, for the state-space description

$$\left\{ \begin{aligned} \begin{bmatrix} \dot{\mathbf{z}}(t) \\ \dot{q}(t) \\ \dot{\tau}(t) \end{bmatrix} &= \begin{bmatrix} (1-q) \cdot (A_{ON}\mathbf{z}(t) + \bar{B}_{ON}\mathbf{u}(t)) + q \cdot (A_{OFF}\mathbf{z}(t) + \bar{B}_{OFF}\mathbf{u}(t)) \\ 0 \\ 1 \end{bmatrix}, (\mathbf{x}, \mathbf{u}, t) \in \mathcal{C}; \tag{44a} \\ \begin{bmatrix} \mathbf{z}^+(t) \\ q^+(t) \\ \tau^+(t) \end{bmatrix} &= \begin{bmatrix} \mathbf{z}(t) \\ \begin{cases} 1, & \text{if } q == 0 \\ 0, & \text{if } q == 1 \end{cases} \\ \begin{cases} \tau, & \text{if } q == 0 \\ 0, & \text{if } q == 1 \end{cases} \end{bmatrix}, (\mathbf{x}, \mathbf{u}, t) \in \mathcal{D}, \tag{44b} \end{aligned} \right.$$

while the flow, jump, and output functions are

$$\begin{cases} \mathcal{C}(\mathbf{x}, \mathbf{u}, t) = \{((q == 0) \wedge (\tau \leq \mu(t) \cdot T_{PWM})) \vee ((q == 1) \wedge (\tau > \mu(t) \cdot T_{PWM}))\}; & \tag{45a} \\ \mathcal{D}(\mathbf{x}, \mathbf{u}, t) = \{((q == 0) \wedge (\tau > \mu(t) \cdot T_{PWM})) \vee ((q == 1) \wedge (\tau > T_{PWM}))\}; & \tag{45b} \\ \mathbf{y}(t) = h(\mathbf{x}(t), \mathbf{u}(t), t). & \tag{45c} \end{cases}$$

In order to encompass DCM regimes, the number of discrete states must be extended with new LTI blocks obtained by adding the mathematical constraint of canceling the diode S_2 voltage and, as such, the corresponding current signal for that branch, along with more sophisticated jump functions, flow and jump sets. For brevity, we will not insist on these extensions, although an example described for the boost converter can be found in [32].

3.2. Toolbox Workflow

A suggested end-to-end workflow for the toolbox can be summarized in the following steps, all of which should be run from intermediary methods of an instance of class `ClosedLoopControlProblem`:

- inherit class `System` to define the nonlinear model of the process as in Equation (1) and Figure 1;
- define equilibrium point specifications as in (11)–(13);

- inherit class `UncertainPlantFactory` and overload method `getRandomPlant`;
- using the desired operating point, linearize a set of systems using (16) and experimentally determine an uncertainty model W_{exp} based on Table 1;
- define uncertainty options, including transfer function structure for all particles, and execute the methods from class `UncertaintyBoundOptimizationProblem` in order to minimize the functional $J_{\Omega, W_{exp}}(W_x)$ from Equation (21), obtaining $W_{x,opt}$;
- run optimization to compute the uncertainty weight $W_{unc}(s)$ as in Table 1;
- define robust control specifications $W_S, W_T,$ and W_{KS} as in Equations (37)–(39);
- synthesize robust μ controller based on Figure 6;
- apply order-reducing methods on the resulting controller;
- validate frequency and time-response performance specifications using the nonlinear system at operating point through Model-in-the-Loop simulations;
- optionally, inherit the classes `HybridSystem` and `UncertainPlantFactory`, respectively, to validate time-response performance specifications using the corresponding hybrid plant model at operating point; for DC-to-DC converter control, the last step should be adapted for CCM or DCM operation.

3.3. Numerical Results

We will briefly present the obtained results for the three converters using the approaches established in Sections 3.1 and 3.2.

3.3.1. SEPIC Converter

The SEPIC converter state-space model for the ON state of switch S_1 , as structured in Equation (41), is

$$\left(\begin{array}{ccccc|ccc} -\frac{1}{r_{C_{in}}C_{in}} & 0 & 0 & 0 & 0 & \frac{1}{r_{C_{in}}C_{in}} & 0 & 0 \\ \frac{1}{L_1} & -\frac{r_{C_{in}}+r_{L_1}+r_{DS_1}}{L_1} & 0 & \frac{r_{DS_2}}{L_1} & 0 & 0 & -\frac{1}{L_1} & 0 \\ 0 & 0 & 0 & \frac{1}{C_1} & 0 & 0 & 0 & 0 \\ 0 & -\frac{r_{DS_1}}{L_2} & -\frac{1}{L_2} & -\frac{r_{DS_1}+r_{C_1}+r_{L_2}}{L_2} & 0 & 0 & \frac{1}{L_2} & 0 \\ 0 & 0 & 0 & 0 & -\frac{1}{(R+r_{C_2})C_2} & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & \frac{R}{R+r_{C_2}} & 0 & 0 & 0 \end{array} \right), \quad (46)$$

while for the OFF state of the switch S_1 is

$$\left(\begin{array}{ccccc|ccc} -\frac{1}{r_{C_{in}}C_{in}} & 0 & 0 & 0 & 0 & \frac{1}{r_{C_{in}}C_{in}} & 0 & 0 \\ \frac{1}{L_1} & -\frac{r_{aux}}{L_1} & -\frac{1}{L_1} & \frac{r_{DS_2}+r_{C_2}}{L_1} & -\frac{1}{L_1} & 0 & 0 & -\frac{1}{L_1} \\ 0 & \frac{1}{C_1} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{r_{DS_2}+r_{C_2}}{L_2} & 0 & -\frac{r_{DS_2}+r_{C_2}+r_{L_2}}{L_2} & \frac{1}{L_2} & 0 & 0 & \frac{1}{L_2} \\ 0 & \frac{R}{(R+r_{C_2})C_2} & 0 & -\frac{R}{(R+r_{C_2})C_2} & -\frac{1}{(R+r_{C_2})C_2} & 0 & 0 & 0 \\ \hline 0 & \frac{r_C R}{R+r_C} & 0 & -\frac{r_C R}{R+r_C} & \frac{R}{R+r_C} & 0 & 0 & 0 \end{array} \right), \quad (47)$$

with the auxiliary notation $r_{aux} = r_{L_1} + r_{C_1} + r_{DS_1} + r_{C_2} + r_{C_{in}}$.

The nominal SEPIC converter parameters and their tolerances are presented in Table 2.

Table 2. Single-ended primary-inductor converter (SEPIC) converter parameters, values, and corresponding tolerances.

Param.	Val.	Tol.	Param.	Val.	Tol.
L_1	2.57 [mH]	$\pm 20\%$	L_2	1.71 [mH]	$\pm 20\%$
r_{L_1}	130 [m Ω]	$\pm 10\%$	r_{L_2}	110 [m Ω]	$\pm 10\%$
r_{DS_1}	0.01 [Ω]	$\pm 10\%$	r_{DS_2}	80 [m Ω]	$\pm 10\%$
C_1	4.7 [μ F]	$\pm 20\%$	C_2	3.57 [μ F]	$\pm 20\%$
r_{C_1}	270 [m Ω]	$\pm 10\%$	r_{C_2}	350 [m Ω]	$\pm 10\%$
C_{in}	3.57 [μ F]	$\pm 20\%$	$r_{C_{in}}$	270 [m Ω]	$\pm 10\%$
V_{F_1}	0.2 [V]	$\pm 10\%$	V_{F_2}	0.62 [V]	$\pm 10\%$

The desired operating point specifications are output signal $y(t) \equiv u_R(t)$ is at 400 [V], with nominal voltage source and load inputs $u_1(t) = E = 300$ [V], $u_2(t) = R = 80$ [Ω]. The initial guesses for the state equilibrium values where $\tilde{x} = [30, 0.5, 30, -0.5, 30]$ and $u_3(t) = \mu = 0.57$ for the duty cycle control input. After computation, the actual equilibrium point is $(\mathbf{u}, \mathbf{x}, \mathbf{y}) = ([300, 80, 0.5788], [300, 6.8711, 297.722, -5, 400], 400)$.

An input multiplicative uncertainty model, i.e., $G(s) = G_n(s)[1 + \Delta(s)W_{unc}(s)]$, $\|\Delta\|_\infty \leq 1$, as in Table 1 has been automatically computed from input $u_3(t)$ to output $y_1(t)$, with tolerances ± 10 [V] and ± 5 [Ω] for inputs $u_1(t) = E$ and $u_2(t) = R$, based on 1000 Monte Carlo simulations. The relevant frequencies vary in the interval $[\underline{\omega} = 10^{-2}, \bar{\omega} = 10^8]$, with 300 equally distributed samples in log domain. A successful set of hyperparameters for the particle swarm optimization algorithm is comprised of a swarm size of 1000, initial swarm span of 10^4 , minimum neighbors fraction of 0.9, and inertia range of $[0.1, 1.1]$, for a transfer function structure as in Equation (19), with a complex pole pair and a complex zero pair, resulting in

$$W_{unc}(s) = \frac{0.67275(s^2 + 941.1s + 2.222 \times 10^5)}{s^2 + 147.3s + 5.422 \times 10^7}. \tag{48}$$

The linearized SEPIC plant family is comprised of fourth-order stable systems, in minimal form, with four zeros, three of which are of nonminimum phase. The nominal model is

$$G_n(s) = 4.1368 \frac{(s + 8.003e+5)(-s + 2.304e+4)(s^2 - 717.4s + 5.145e+7)}{(s^2 + 2673s + 3.749e+7)(s^2 + 1339s + 6.493e+7)}. \tag{49}$$

The entire uncertainty family has the same structure with poles, zeros, and equilibrium points in the vicinity of the nominal counterparts. The uncertain SEPIC plant family structure and behavior, along with the PSO conservative bound computation are illustrated in Figure 8. Figure 8-1 illustrates the pole-zero plot for the linearized uncertain SEPIC converter family, Figure 8-2 shows the best particle frequency-response fit $W_x(s)$ as in Equation (19) on the right y axis, and, also, the best functional fit on the left y axis, Figure 8-3 and 8-4 show the frequency response of the plant $G(s)$ family and uncertainty family $\Delta(s)W_{unc}(s)$, respectively, while Figure 8-5 illustrates the system states and outputs for a 2% step disturbance relative to the equilibrium input value $\mu_0 = 0.5788$.

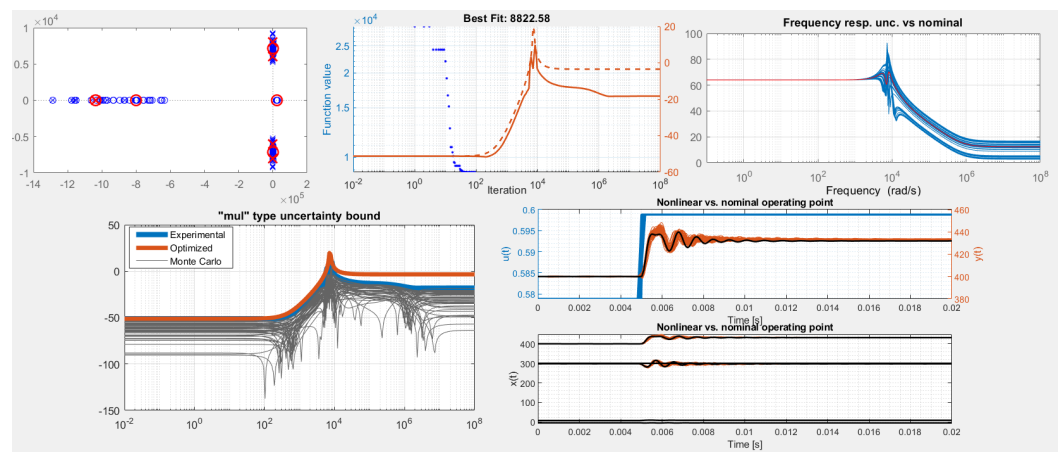


Figure 8. SEPIC multiplicative uncertainty set computation and open loop responses for the operating point with $E = 300[V]$, $R = 80[\Omega]$, $U_R = 400[V]$: step and frequency response, pole-zero placement; the step response is simulated for nonlinear plants sampled using `UncertainPlantFactory`.

For controller synthesis, the preferred loop shaping specifications were selected to highly penalize the control effort near the system resonance, as the obtained controllers would be difficult to implement in practice, needing high sampling frequencies, i.e., $f_e > 20$ [kHz]. As such, for the sensitivity function: $\omega_B = 200$ [rad/s], $A = 10^{-2}$, $M = 2$, $n = 1$; for the complementary sensitivity function: $\omega_{BT} = 2000$ [rad/s], $A_T = 10^{-4}$, $M_T = 2$, $n = 2$; and for the control effort $M_0 = 100$, $M_\infty = 10^5$, $|W_{KS}(j \cdot 200)| = 250$, resulting in

$$W_S(s) = \frac{0.5s + 200}{s + 2}, \quad W_T(s) = \frac{s^2 + 4000s + 4 \times 10^6}{1 \times 10^{-4}s^2 + 56.57s + 8 \times 10^6}, \quad W_{KS}(s) = \frac{10^5s + 8.729 \times 10^6}{s + 8.729 \times 10^4}. \quad (50)$$

From the μ synthesis procedure, a controller of order 21 is obtained. After order reduction, the smallest controller which manages to assure all imposed specifications for the plant family, with a peak value $\mu_\Delta(\text{LLFT}(P, K)) \leq 0.8361 < 1$, is given by the third-order system

$$K_{red}^{SEPIC} = \left(\begin{array}{ccc|c} -1.997 & 3.056 & 3.227 & -0.5018 \\ -3.057 & -2197 & -6118 & -0.3838 \\ 3.225 & 6118 & -1.016 \times 10^4 & 0.4055 \\ \hline -0.5018 & 0.3838 & 0.4055 & 0 \end{array} \right). \quad (51)$$

The controller design phase, order reduction, and frequency response closed loop performance of the reduced-order one for the uncertain plant family are illustrated in Figures 9 and 10. In this case, the control system has very large stability margins, with a phase margin of $\approx 82.1^\circ$ and gain margin in the interval [19.4, 20.3] [dB]. Additionally, as specified by the $n = 2$ and $A_T = 10^{-4}$ parameters of the complementary sensitivity weighting function, W_T , the closed loop control system mitigates sensor noise signals with a considered spectrum starting from $\omega_{BT} > 2000$ [rad/s], using an initial roll-off of -40 [dB/dec], followed by an attenuation of at least four orders of magnitude. In the actual MiL simulations, the attenuation does not stop at the prescribed value, as the system manages to maintain at least a -20 [dB/dec] roll-off.

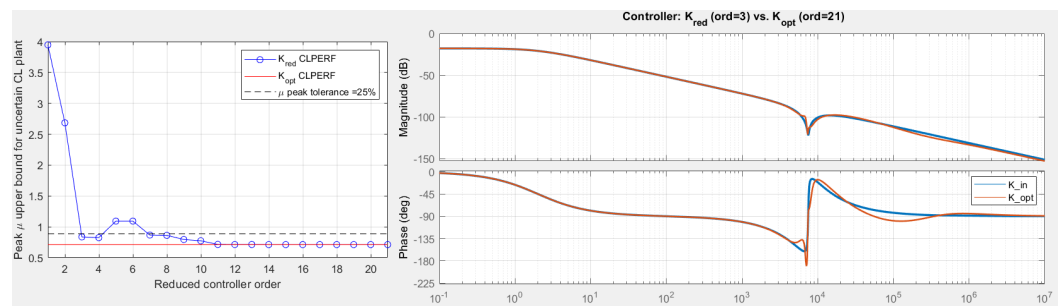


Figure 9. SEPIC-synthesized (order 21) vs. reduced (order 3) controllers; the peak μ value does not monotonically decrease with respect to controller order due to the influence of the ≈ 3750 [rad/s] notch over the converter resonance, which was not taken into account by the order reduction mechanism.

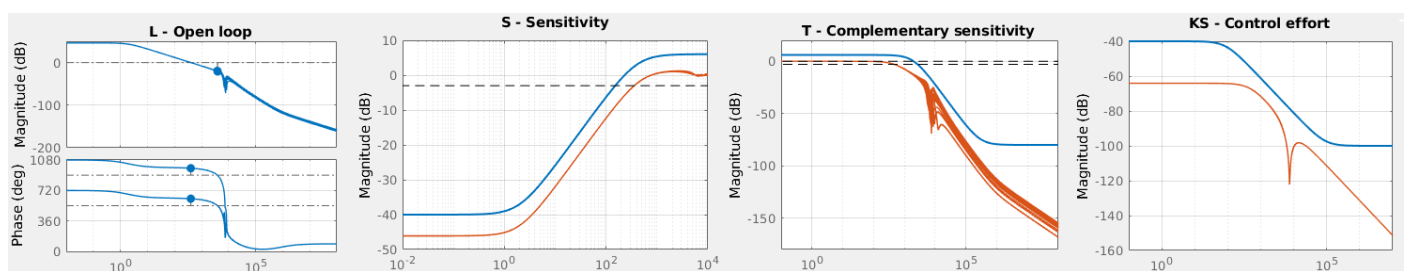


Figure 10. SEPIC open loop plant family with S , T , and KS functions using the reduced-order controller, which retains all imposed performance specifications for all test cases, provides high phase and gain margins, and guarantees closed loop response practically similar to that of a first-order low-pass filter; a relatively low bandwidth was imposed to compensate the SEPIC converter resonance and presence of multiple nonminimum phase zeros.

From Figure 11, the closed loop bandwidth can be observed as $\omega_B > 200$ [rad/s], equivalent to a rise time less than ≈ 5 [ms], a negligible steady-state error of $\approx 10^{-2} \times (y_{ss} - y_0) = 0.2$ [V], where y_{ss} represents the steady state value of the system, relative to the desired equilibrium value y_0 . Moreover, the system has no overshoot, and it behaves like a first-order low-pass filter by design. The nonlinear MiL simulation options are $N = 50$ random plants from the uncertainty set, solver is `ode15i`, due to the difficulty of simulating the closed loop plant otherwise (it is numerically unstable), with a step on the reference signal of 5% from its initial equilibrium value of approximately 400 [V] and a simulation time of 0.05 [s]. Almost the same conditions apply to the hybrid MiL simulation, with the solver switched to `ode113`, as `ode15i` is not supported here, and a PWM period selected randomly for each experiment from a nominal value $T_{PWM} = 17.5$ [μ s] with a $\pm 20\%$ fluctuation from one simulation to another. A comparison of the nonlinear and hybrid cases is presented also in Figure 11, where it can be seen that the transitory regime is practically identical, but for the hybrid case, a slightly lower command signal is necessary in steady state. Due to working with $u_R(t)$ instead of $u_C(t)$ only for the output signal, the current ripple is propagated into the measured voltage.

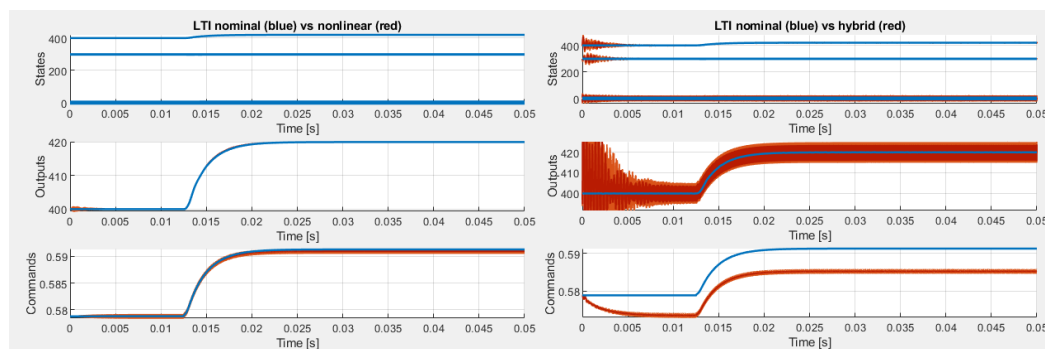


Figure 11. SEPIC-averaged state-space and hybrid model Monte Carlo closed loop simulations; due to a relatively low bandwidth imposed in order to obtain good stability margins and easily implementable controllers, all plants respond almost identically, although component tolerances reach values of $\pm 20\%$; the initial transients exist due to starting the simulations from the nominal equilibrium point only.

3.3.2. Buck Converter

The buck converter state-space models for the *ON* and *OFF* states of switch S_1 , respectively, as in Equation (41), are

$$\left(\begin{array}{cc|ccc} -\frac{r_p+r_{DS1}}{L} & -\frac{R}{(R+r_C)L} & \frac{1}{L} & -\frac{1}{L} & 0 \\ \frac{R}{(R+r_C)C} & -\frac{1}{(R+r_C)C} & 0 & 0 & 0 \\ \hline \frac{r_C R}{R+r_C} & \frac{R}{R+r_C} & 0 & 0 & 0 \end{array} \right); \left(\begin{array}{cc|ccc} -\frac{r_p-r_{DS2}}{L} & -\frac{R}{(R+r_C)L} & 0 & 0 & -\frac{1}{L} \\ \frac{R}{(R+r_C)C} & -\frac{1}{(R+r_C)C} & 0 & 0 & 0 \\ \hline \frac{r_C R}{R+r_C} & \frac{R}{R+r_C} & 0 & 0 & 0 \end{array} \right), \quad (52)$$

with the auxiliary notation $r_p = r_L + \frac{r_C R}{R+r_C} = r_L + r_C || R$.

The nominal buck converter parameters and their tolerances are presented in Table 3.

Table 3. Buck and boost converter parameters, values, and corresponding tolerances.

Param.	Val.	Tol.	Param.	Val.	Tol.
L	40 [μH]	$\pm 20\%$	r_L	10 [$\text{m}\Omega$]	$\pm 10\%$
C	600 [μF]	$\pm 20\%$	r_C	0.2 [Ω]	$\pm 10\%$
r_{DS1}	0.01 [Ω]	$\pm 10\%$	r_{DS2}	0.01 [Ω]	$\pm 10\%$
V_{F1}	0.2 [V]	$\pm 10\%$	V_{F2}	0.2 [V]	$\pm 10\%$

The desired operating point specifications are as follows: the output signal $y(t) \equiv u_R(t)$ is at 5 [V], with nominal voltage source and load inputs $u_1(t) = E = 12$ [V], $u_2(t) = R = 15$ [Ω]. The initial guesses for the state equilibrium values where $\tilde{x} = [1.25, 5]$ and $u_3(t) = \mu = 0.5$ for the duty cycle control input. After computation, the actual equilibrium point is $(\mathbf{u}, \mathbf{x}, \mathbf{y}) = ([12, 15, 0.4335], [0.333, 4.999], 5)$.

An input multiplicative uncertainty model, i.e., $G(s) = G_n(s)[1 + \Delta(s)W_{unc}(s)]$, $\|\Delta\|_\infty \leq 1$, as in Table 1, has been automatically computed from input $u_3(t)$ to output $y_1(t)$, with tolerances ± 1 [V] and ± 1 [Ω] for inputs $u_1(t) = E$ and $u_2(t) = R$, respectively, based on 1000 Monte Carlo simulations. The relevant frequencies vary in the interval $[\omega = 10^1, \bar{\omega} = 10^7]$, with 200 equally distributed samples in log domain. A successful set of hyperparameters for the particle swarm optimization algorithm is comprised of a swarm size of 1000, initial swarm span of 10^4 , minimum neighbors fraction of 0.9, and inertia range of [0.1, 1.1] for a transfer function structure as in Equation (19), with a real pole and a real zero, resulting in

$$W_{unc}(s) = \frac{0.51758(s + 510.5)}{s + 2906}. \quad (53)$$

The linearized buck plant family is comprised of second-order stable systems, in minimal form, with one zero. The nominal model is

$$G_n(s) = \frac{59178(s + 8333)}{s^2 + 5261s + 4.114 \times 10^7}. \tag{54}$$

The entire uncertainty family has the same structure with poles, zeros, and equilibrium points in the vicinity of the nominal counterparts. The uncertain buck plant family structure and behavior, along with the PSO conservative bound computation are illustrated in Figure 12.

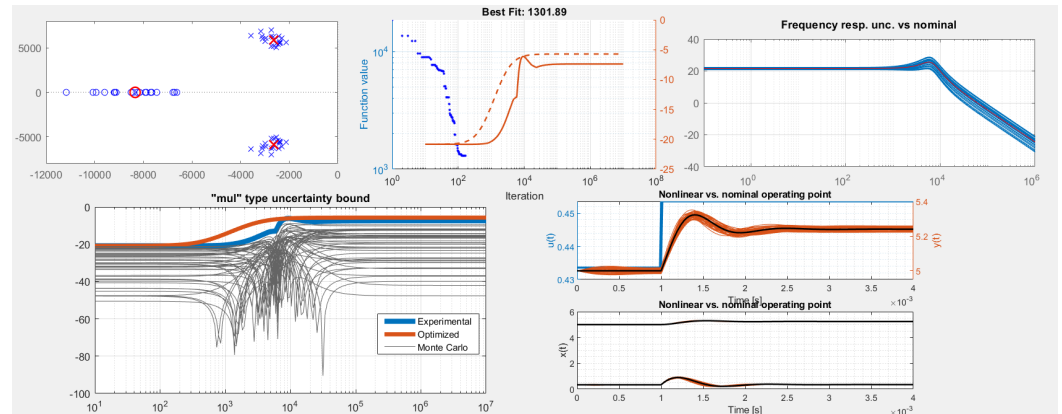


Figure 12. Buck multiplicative uncertainty set computation and open loop responses for the operating point with $E = 12[V]$, $R = 15[\Omega]$, $U_R = 5[V]$: step and frequency response, pole-zero placement; the step response is simulated for nonlinear plants sampled using `UncertainPlantFactory`.

For controller synthesis, the weighting function specifications were for the sensitivity function $\omega_B = 1200$ [rad/s], $A = 10^{-4}$, $M = 2$, $n = 1$; for the complementary sensitivity function $\omega_{BT} = 12 \times 10^3$ [rad/s], $A_T = 10^{-4}$, $M_T = 2$, $n = 2$; and for the control effort $M_0 = 0.1$, $M_\infty = 100$, $|W_{KS}(j \cdot 1200)| = 2$, resulting in

$$W_S(s) = \frac{0.5s + 1200}{s + 0.12}, W_T(s) = \frac{s^2 + 24000s + 1.44 \times 10^8}{1 \times 10^{-4}s^2 + 339.4s + 2.88 \times 10^8}, W_{KS}(s) = \frac{100s + 6006}{s + 6.006 \times 10^4}. \tag{55}$$

As specified by the $n = 2$ and $A_T = 10^{-4}$ parameters of the complementary sensitivity weighting function, W_T , the closed loop control system mitigates sensor noise signals with a considered spectrum starting from $\omega_{BT} > 12,000$ [rad/s], using an initial roll-off of -40 [dB/dec], followed by an attenuation of at least four orders of magnitude. From the μ synthesis, a controller of order 17 is obtained. After order reduction, the smallest controller which manages to assure all imposed specifications for the plant family, with a peak $\mu_\Delta(\text{LLFT}(P, K)) \leq 0.97 < 1$, is

$$K_{red}^{Buck} = \left(\begin{array}{ccc|c} -0.12 & -0.003479 & -0.4751 & 12.68 \\ -0.0001768 & -2.304 & -1.241 \times 10^4 & -0.1827 \\ -0.4611 & 1.241 \times 10^4 & -4.522 \times 10^4 & 25.09 \\ \hline 12.68 & 0.1838 & 25.09 & 0 \end{array} \right). \tag{56}$$

The nonlinear MiL simulation options are $N = 50$ random plants from the uncertainty set, with the `ode23t` solver, with a step on the reference signal of 5% from its initial equilibrium value of approximately 5 [V] and a simulation time of 0.02 [s]. The simulation conditions for the hybrid MiL case are identical, with an additional PWM period selected randomly for each experiment from a nominal value $T_{PWM} = 17.5$ [μ s] with a $\pm 20\%$ fluctuation from one simulation to another. A comparison of the nonlinear and hybrid cases is presented in Figure 13, where it can be seen that the transitory regime is practically

identical. Due to working with $u_R(t)$ instead of $u_C(t)$ only for the output signal, the current ripple is propagated into the measured voltage.

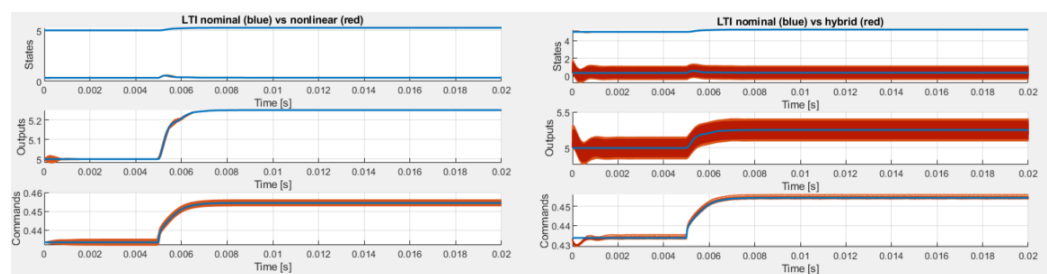


Figure 13. Buck averaged state-space and hybrid model Monte Carlo closed loop simulations with a closed loop bandwidth $\omega_B > 1200$ [rad/s], equivalent to a rise time of ≈ 0.83 [ms], negligible steady-state error, and no overshoot.

3.3.3. Boost Converter

The boost converter state-space models for the ON and OFF states of switch S_1 , respectively, as in Equation (41), are

$$\left(\begin{array}{cc|ccc} -\frac{r_L+r_{DS1}}{L} & 0 & \frac{1}{L} & -\frac{1}{L} & 0 \\ 0 & -\frac{1}{(R+r_C)C} & 0 & 0 & 0 \\ \hline 0 & \frac{R}{R+r_C} & 0 & 0 & 0 \end{array} \right); \left(\begin{array}{cc|ccc} -\frac{r_{aux}}{L} & -\frac{R}{(R+r_C)L} & \frac{1}{L} & 0 & -\frac{1}{L} \\ \frac{R}{(R+r_C)C} & -\frac{1}{(R+r_C)C} & 0 & 0 & 0 \\ \hline \frac{r_C R}{R+r_C} & \frac{R}{R+r_C} & 0 & 0 & 0 \end{array} \right), \quad (57)$$

with the auxiliary notation $r_{aux} = r_L + r_{DS2} + \frac{r_C R}{R+r_C} = r_L + r_{DS2} + r_C \parallel R$. The nominal boost converter parameters and their tolerances are presented in Table 3, as they correspond with the buck converter parameters.

The desired operating point specifications are as follows: the output signal $y(t) \equiv u_R(t)$ is at 24 [V], with nominal voltage source and load inputs $u_1(t) = E = 12$ [V], $u_2(t) = R = 15$ [Ω]. The initial guesses for the state equilibrium values where $\tilde{x} = [3, 24]$ and $u_3(t) = \mu = 0.5$ for the duty cycle control input. After computation, the actual equilibrium point is $(\mathbf{u}, \mathbf{x}, \mathbf{y}) = ([12, 15, 0.5179], [0.3189, 23.999], 24)$.

An input multiplicative uncertainty model, i.e., $G(s) = G_n(s)[1 + \Delta(s)W_{unc}(s)]$, $\|\Delta\|_\infty \leq 1$, as in Table 1, has been automatically computed from input $u_3(t)$ to output $y_1(t)$, with tolerances ± 1 [V] and ± 1 [Ω] for inputs $u_1(t) = E$ and $u_2(t) = R$, respectively, based on 1000 Monte Carlo simulations. The relevant frequencies vary in the interval $[\omega = 10^1, \bar{\omega} = 10^7]$, with 200 equally distributed samples in log domain. A successful set of hyperparameters for the particle swarm optimization algorithm is comprised of a swarm size of 1500, initial swarm span of 10^4 , minimum neighbors fraction of 0.9, and inertia range of [0.1, 1.1] for a transfer function structure as in Equation (19), with two real poles and two real zeros, resulting in

$$W_{unc}(s) = \frac{0.26592(s + 512.5)(s + 3.535 \times 10^4)}{(s + 4016)(s + 1.389 \times 10^4)}. \quad (58)$$

The linearized boost plant family is comprised of second-order stable systems, in minimal form, with two zeros, one being of nonminimum phase. The nominal model is

$$G_n(s) = \frac{0.65505(-s + 8.551 \times 10^4)(s + 8333)}{s^2 + 2988s + 9.746 \times 10^6}. \quad (59)$$

The entire uncertainty family has the same structure with poles, zeros, and equilibrium points in the vicinity of the nominal counterparts. The uncertain boost plant family structure and behavior, along with the PSO least conservative second-order bound computation are illustrated in Figure 14.

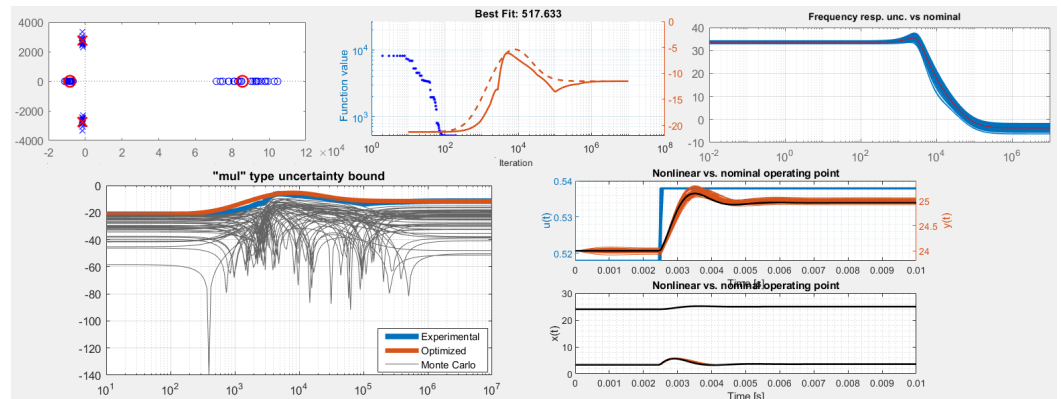


Figure 14. Boost multiplicative uncertainty set computation and open loop responses for the operating point with $E = 12[V]$, $R = 15[\Omega]$, $U_R = 24[V]$: step and frequency response, pole-zero placement; the step response is simulated for nonlinear plants sampled using `UncertainPlantFactory`.

For controller synthesis, the weighting function specifications were for the sensitivity function $\omega_B = 650$ [rad/s], $A = 10^{-4}$, $M = 2$, $n = 1$; for the complementary sensitivity function $\omega_{BT} = 3250$ [rad/s], $A_T = 10^{-4}$, $M_T = 2$, $n = 1$; and for the control effort $M_0 = 0.1$, $M_\infty = 100$, $|W_{KS}(j \cdot 650)| = 2$, resulting in

$$W_S(s) = \frac{0.5s + 650}{s + 0.065}, \quad W_T(s) = \frac{s + 3250}{0.0001s + 6500}, \quad W_{KS}(s) = \frac{100s + 3253}{s + 3.252 \times 10^4}. \quad (60)$$

An intrinsic limitation for the boost converter is that the bandwidth must not be imposed more than half of the value of the right-half plane non-minimum phase zero of the process, i.e., $\omega_B \leq \frac{\omega_z}{2} \approx 43,000$ [rad/s]. For this problem, this is a sufficiently high margin, as we also imposed a limitation for the command signal through W_{KS} . As specified by the $n = 2$ and $A_T = 10^{-4}$ parameters of the complementary sensitivity weighting function, W_T , the closed loop control system mitigates sensor noise signals with a considered spectrum starting from $\omega_{BT} > 3250$ [rad/s], using an initial roll-off of -40 [dB/dec], followed by an attenuation of at least four orders of magnitude. From the μ synthesis, a controller of order 17 is obtained. After order reduction, the smallest controller which retains all imposed specifications for the plant family, with a peak value $\mu_\Delta(\text{LLFT}(P, K)) \leq 0.9547 < 1$, is

$$K_{red}^{Boost} = \left(\begin{array}{ccc|c} -0.065 & 0.8025 & -0.002966 & 5.007 \\ 0.7116 & -9.715 \times 10^4 & 1.066 \times 10^4 & -30.91 \\ 0.002575 & -1.065 \times 10^4 & -1.422 & -0.113 \\ \hline 5.007 & -30.91 & 0.1142 & 0 \end{array} \right). \quad (61)$$

With this regulator, the boost converter control system with parameters from Table 3 has very large stability margins, with phase margins between $[81, 101]$ $^\circ$ and gain margins in the interval $[40, 46]$ [dB]. The obtained sensitivity bandwidths vary between $[828, 2510]$ [rad/s], all of them better than the prespecified value of 800.

Nonlinear MiL simulation options are $N = 50$ random plants from the uncertainty set, with the `ode15i` solver, with a step on the reference signal of 5% from its initial equilibrium value of approximately 5 [V] and a simulation time of 0.02 [s]. The simulation conditions for the hybrid MiL case are almost identical, with the use of the `ode113` solver instead, set to a relative tolerance of 10^{-8} , and with an additional PWM period selected randomly for each experiment from a nominal value $T_{PWM} = 17.5$ [μ s] with a $\pm 20\%$ fluctuation from one simulation to another. A comparison of the nonlinear and hybrid cases is presented in Figure 15, where it can be seen that the transitory regime is practically identical. Due to working with $u_R(t)$ instead of $u_C(t)$ only for the output signal, the current ripple is propagated into the measured voltage.

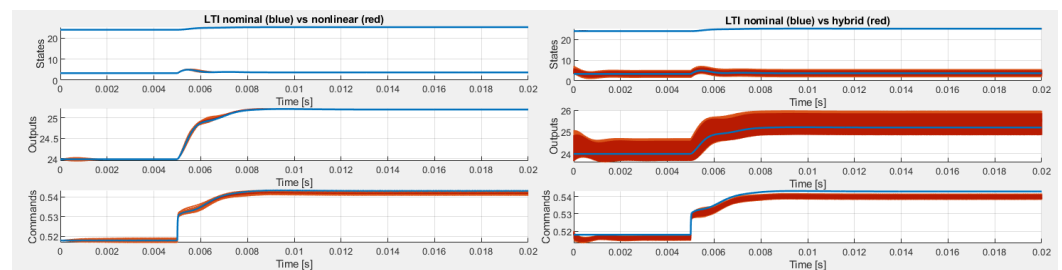


Figure 15. Boost averaged state-space and hybrid model Monte Carlo closed loop simulations with a closed loop bandwidth $\omega_B > 650$ [rad/s], equivalent to a rise time less than ≈ 1.53 [ms], negligible steady-state error and no overshoot.

4. Conclusions and Future Work

There are several aspects to be discussed with regards to the presented toolbox and, additionally, including future work to be implemented in upcoming iterations.

The main reason for which the hybrid system framework is considered for the proposed toolbox is that the continuous-time plant needs to be regulated with a numerical controller. The hybrid system context may include LTI or nonlinear systems, switching systems (which are hybrid by nature), and singular systems, represented using differential-algebraic equations (DAEs). As such, for a continuous-time plant, a continuous-time controller is designed using the robust control framework. The controller is required to be numerically implementable, therefore two interfaces are necessary, which lead to a hybrid system. The numerical implementation must be easily obtained and automatically validated using rapid control prototyping (RCP) techniques. However, the properties of the closed loop system need to be reanalyzed after the discretization of the controller. Differential-algebraic equations represent a useful framework for modeling dynamical systems in engineering with a network-based structure of components. They are used in various industry fields such as mechanics (e.g., multiple-link mobile manipulator model), chemical engineering (modeling of chemical reactions), electrical engineering, cyber-physical systems, etc. All categories of processes taken into consideration in the hybrid framework are described using an approximate model that incorporates their relevant behavior. However, these types of systems have model and structure uncertainties. Moreover, the nonlinear systems that are linearized around an equilibrium point introduce such uncertainties as well. Therefore, to consider these uncertainties for the controller design, the robust control framework is mandatory. The first main applicability of RCP was to derive the necessary C/C++ source code with drivers for a given target microprocessor, and to simulate in reproducible conditions the behavior of a complex system. For the latter case, the most relevant simulation types, given in increasing order of complexity and closeness to reality, are Model-in-the-Loop (MiL), Software-in-the-Loop (SiL), and Hardware-in-the-Loop (HiL).

Many modeling software programs return circuits or mechanical systems already in DAE form, and it would be difficult, or sometimes impossible, to reformulate them in an ODE form without changing variables and losing their intended physical significance. In the context of the robust control framework used for DAEs, a significant work is the monograph in [33]. Although the robust control theory was well formulated in the previous years, this is still an open domain for research and publication, as surveyed and described in [34]. The difficulty of using methods which work correctly for multiple operating points is mitigated by using adaptive methods, such as gain scheduling for tracking problems. Furthermore, other relevant problems are automatic C/C++ code generation for controller implementation and the commutation between the prescribed operating points. Considering use cases for modeling, simulation, computer-aided design, and RCP, a relevant set of examples in the domain of power electronics, hybrid vehicles, and renewable energy systems is given in [35,36], where, although the main limitation is that analysis, control, and implementation aspects must be performed individually for the presented applications, they can be generally included under the same software framework, with

the important exception of the robust control design and verification, along with the quantization sensitivity analysis in a unitary manner.

The main hindrance is that the majority of applications involve DAE systems of index greater than 1, meaning they necessitate more than one derivation step in order to formulate the problem, so there is a great need of tools that can also deal with high order DAEs, i.e., index 2 or 3 [37–39]. Another reason for the proposal of considering DAEs model is that they are also explicitly related to control issues, regarding both physical and operational constraints. Two illustrative examples are the case of improper systems, such as an ideal PID controller, and the elements that realize the decoupling in the MIMO systems case and when the plant has impulsive dynamics. The robust control techniques' drawback is that the closed loop $\mathcal{H}_2/\mathcal{H}_\infty$ norm must be minimized using the prescribed weighting functions that penalize the exogenous outputs and these weighting functions need to be found ad hoc, which sometimes lead to some intermediary bad controllers and work overhead. Furthermore, after numerical implementation, the discrete controller loses a part of the imposed performances due to an inadequate sampling period or badly selected quantization levels. Although there are solutions in the literature, there is no unified approach to solve all these mentioned problems. As an extension to the framework and mindset given by the two previously mentioned RCP use cases, the current project proposed a highly automated toolbox for robust control design, which, in the current state-of-the-art is a highly iterative design process, when taking into consideration plant uncertainties, although the mathematical background for solving the optimization problems is well established. It proposes to eliminate design overhead when considering and modifying a specification set, manually redesigning the weighting functions, the optimization procedure, discretization of the regulators, quantization analysis, and closed loop analysis for the linearized and initial hybrid plant. Furthermore, in unison with high-performance numerical toolboxes, a justified report should automatically result after its use and explicitly state when unrealistic design specifications were considered.

Author Contributions: Conceptualization, M.Ş. and V.M.; methodology, M.Ş.; software, M.Ş.; validation, V.M. and P.D.; formal analysis, M.Ş. and P.D.; investigation, M.Ş. and M.S.; resources, M.Ş.; data curation, D.M.; writing—original draft preparation, M.Ş., M.S., and V.M.; writing—review and editing, V.M., D.M., and M.S.; visualization, M.Ş. and D.M.; supervision, P.D.; project administration, M.Ş.; funding acquisition, D.M. All authors have read and agreed to the published version of the manuscript.

Funding: This paper was supported by the Project “Entrepreneurial competences and excellence research in doctoral and postdoctoral programs—ANTREDOC”, project co-funded by the European Social Fund.

Conflicts of Interest: The authors declare no conflicts of interest.

Sample Availability: Relevant source code implementation, including classes, objects, scripts, and data to reproduce the results presented in this paper are available from the GitHub repository: [roconsys-toolbox-public](https://github.com/roconsys-toolbox-public).

Abbreviations

The following abbreviations are used in this manuscript:

CACSD	Computer-Aided Control System Design
CCM	Continuous Conduction Mode
DAE	Differential-Algebraic Equation
DC	Direct Current
DCM	Discontinuous Conduction Mode
DOF	Degree of Freedom
HyEQ	Hybrid Equations Toolbox
LLFT	Lower Linear Fractional Transformation

LMI	Linear Matrix Inequality
LTI	Linear Time-Invariant
MiL	Model-in-the-Loop
ODE	Ordinary Differential Equation
PSO	Particle Swarm Optimization
RCP	Rapid Control Prototyping
SEPIC	Single-ended primary-inductor converter
ULFT	Upper Linear Fractional Transformation

References

- Balas, G.; Chiang, R.; Packard, A.; Safonov, M. *Robust Control Toolbox, Reference for MATLAB*; The MathWorks, Inc.: Natick, MA, USA, 2020.
- Global Optimization Toolbox, User's Guide for MATLAB*; The MathWorks, Inc.: Natick, MA, USA, 2020.
- Feyel, P.; Duc, G.; Sandou, G. Optimal tuning of \mathcal{H}_∞ fixed-structure robust controller against multiple high-level requirements using evolutionary computation. *Int. J. Robust Nonlinear Control* **2019**, *29*, 949–972. [[CrossRef](#)]
- Blackwell, C.; Sastry, M.K.S. Multivar - A MATLAB based MIMO Control System Design Application. In Proceedings of the 8th International Conference on Computational Intelligence and Communication Networks, Tehri, India, 23–25 December 2016.
- Keller, P. Robust and optimal \mathcal{H}_∞ control in LabVIEW. *IEEE Conf. Control. Technol. Appl. (CCTA)* **2017**. [[CrossRef](#)]
- Jacobs, L.; Verbandt, M.; De Preter, A.; Anthonis, J.; Swevers, J.; Pipeleers, G. *A Toolbox for Robust Control Design: An Illustrative Case Study*; IEEE: Tokyo, Japan, 2018.
- Sadeghpour, M.; de Oliveira, V.; Karimi, A. A Toolbox for Robust PID Controller Tuning Using Convex Optimization. *IFAC Proc. Vol.* **2012**, *45*, 158–163. [[CrossRef](#)]
- Ljung, L. *System Identification Toolbox, Reference for MATLAB*; The MathWorks, Inc.: Natick, MA, USA, 2020.
- Karimi, A.S. Frequency-Domain Robust Control Toolbox. In Proceedings of the 52nd IEEE Conference on Decision and Control, Firenze, Italy, 10–13 December 2013. [[CrossRef](#)]
- Sadeghzadeh, A.; Karimi, A.S. Fixed-structure \mathcal{H}_2 controller design for polytopic systems via LMIs. *Optim. Control Appl. Meth.* **2014**. [[CrossRef](#)]
- Verbandt, M.; Swevers, J.; Pipeleers, G. An LTI control toolbox—Simplifying optimal feedback controller design. In Proceedings of the 2016 European Control Conference (ECC), Aalborg, Denmark, 29 June–1 July 2016. [[CrossRef](#)]
- Șuşcă, M. Solving Algebraic Riccati Equations Using Proper Deflating Subspaces for $\mathcal{H}_2/\mathcal{H}_\infty$ Synthesis. Master's Thesis, Technical University of Cluj-Napoca, Cluj-Napoca, Romania, 2019. [[CrossRef](#)]
- Șuşcă, M.; Mihaly, V.; Stănescu, M.; Dobra, P. Iterative Refinement Procedure for Solutions to Algebraic Riccati Equations. In Proceedings of the 2020 IEEE International Conference on Automation, Quality and Testing, Robotics (AQTR), Cluj-Napoca, Romania, 21–23 May 2020. [[CrossRef](#)]
- Mihaly, V. General Purpose Linear Matrix Inequality Solver With Applications in Robust and Nonlinear Control. Master's Thesis, Technical University of Cluj-Napoca, Cluj-Napoca, Romania, 2020.
- Gumussoy, S.; Henrion, D.; Millstone, M.; Overton, M.L. Multiobjective Robust Control with HIFOO 2.0. In Proceedings of the 6th IFAC Symposium on Robust Control Design, Haifa, Israel, 16–18 June 2009.
- Sanfelice, R.G.; Copp, D.A.; Nanez, P. *Hybrid Equations (HyEQ) Toolbox v2.04—A Toolbox for Simulating Hybrid Systems in MATLAB/Simulink®*; MathWorks®: Natick, MA, USA, 2017.
- Goebel, R.; Sanfelice, R.G.; Teel A.R. *Hybrid Dynamical Systems: Modeling, Stability, and Robustness*; Princeton University Press: Princeton, NJ, USA, 2012.
- Griewank, A.; Walther, A. *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation*, 2nd ed.; SIAM: Philadelphia, PA, USA, 2008.
- Kennedy, J.; Eberhart, R. Particle swarm optimization. In Proceedings of the ICNN'95—International Conference on Neural Networks, Perth, WA, Australia, 27 November–1 December 1995; Volume 4, pp. 1942–1948. [[CrossRef](#)]
- Doyle, J.C.; Glover K.; Khargonekar, P.P.; Francis, B.A. State-Space Solutions to Standard \mathcal{H}_2 and \mathcal{H}_∞ Control Problems. *IEEE Trans. Autom. Control.* **1989**, *34*, 831–847. [[CrossRef](#)]
- Fortuna, L.; Frasca, M. *Optimal and Robust Control—Advanced Topics with MATLAB*; CRC Press: Boca Raton, FL, USA, 2012.
- Gahinet, P.; Apkarian P. A Linear Matrix Inequality Approach to \mathcal{H}_∞ Control. *Int. J. Robust Nonlinear Control.* **1994**, *4*, 421–448. [[CrossRef](#)]
- Ionescu, V.; Oară, C.; Weiss, M. *Generalized Riccati Theory and Robust Control: A Popov Function Approach*; John Wiley & Sons Ltd.: Chichester, UK, 1999.
- Zhou, K.; Doyle, J.C.; Glover, K. *Robust and Optimal Control*; Prentice Hall: Englewood Cliffs, NJ, USA, 1996.
- Packard, A.; Doyle, J.; Balas, G. Linear, Multivariable Robust Control With a μ Perspective. *J. Dyn. Syst. Meas. Control.* **1993**, *115*, 426–438. [[CrossRef](#)]
- Skogestad, S.; Postlethwaite, I. *Multivariable Feedback Control: Analysis and Design*, 2nd ed.; John Wiley & Sons Ltd.: Chichester, UK, 2005.

27. Raghavendra, K.V.G.; Zeb, K.; Muthusamy, A.; Krishna, T.N.V.; Kumar, S.V.S.; Kim, D.-H.; Kim, M.-S.; Cho, H.-G.; Kim, H.-J. A Comprehensive Review of DC–DC Converter Topologies and Modulation Strategies with Recent Advances in Solar Photovoltaic Systems. *Electronics* **2020**, *9*, 31. [[CrossRef](#)]
28. Chang, E.-C.; Cheng, C.-A.; Wu, R.-C. Robust Optimal Tracking Control of a Full-Bridge DC-AC. *Converter. Appl. Sci.* **2021**, *11*, 1211. [[CrossRef](#)]
29. Garcia, G.; Lopez Santos, O. A Unified Approach for the Control of Power Electronics Converters. Part I—Stabilization and Regulation. *Appl. Sci.* **2021**, *11*, 631. [[CrossRef](#)]
30. Mihaly, V.; Şuşcă, M.; Dobra, P. Passivity-Based Controller for Nonideal DC-to-DC Boost Converter. In Proceedings of the 2019 22nd International Conference on Control Systems and Computer Science (CSCS), Bucharest, Romania, 28–30 May 2019; pp. 30–35. [[CrossRef](#)]
31. Repecho, V.; Biel, D.; Olm, J.M.; Fossas, E. Robust sliding mode control of a DC/DC Boost converter with switching frequency regulation. *J. Frankl. Inst.* **2018**, *355*, 5367–5383. [[CrossRef](#)]
32. Lunze, I.; Lagarrigue, F.L. *Handbook of Hybrid Systems Control: Theory, Tools, Applications*; Cambridge University Press: Cambridge, UK, 2009.
33. Feng, Y.; Yagoubi M. *Robust Control of Linear Descriptor Systems*; Studies in Systems, Decision and Control 102; Springer: Singapore, 2017.
34. Liu, K.-Z.; Yao, Y. *ROBUST CONTROL—Theory and Applications*; John Wiley & Sons (Asia) Pte Ltd.: Singapore, 2016.
35. Zamboni, W.; Petrone, G. (Eds.) *ELECTRIMACS 2019, Selected Papers—Volume 1*; Springer International Publishing: Cham, Switzerland, 2020.
36. Zamboni, W.; Petrone, G. (Eds.) *ELECTRIMACS 2019, Selected Papers—Volume 2*; Springer International Publishing: Cham, Switzerland, 2020.
37. Ascher, U.M.; Petzold, L.R. *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations*; SIAM: Philadelphia, PA, USA, 1998.
38. Brenan, K.E.; Campbell, S.L.; Petzold L.R. *Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations*; SIAM Classics in Applied Mathematics: Philadelphia, PA, USA, 1996.
39. Milano, F.; Dassios, I.; Liu, M.; Tzounas, G. *Eigenvalue Problems in Power Systems*; CRC Press: Boca Raton, FL, USA, 2021.