Robust Simulation of Hybrid Mechanistic and Machine Learning Models

October 15, 2020

Matthew D. Stuber, PhD
Assistant Professor, University of Connecticut

2020 INCOSE New England Fall Workshop

Outline

1. Introduction
2. Robust Simulation of Mechanistic Models
3. Concepts of Hybrid Models
4. Robust Simulation of Hybrid Models
5. Conclusion

Robust Simulation

- A “Robust System” mitigates the effects of uncertainty to ensure performance/safety constraints are satisfied.

Robust Simulation

- A "Robust System" mitigates the effects of uncertainty to ensure performance/safety constraints are satisfied.
- "Robust Simulation" refers to the ability to rigorously account for the impacts of uncertainty via a model-based (i.e., simulation) approach
  - Conclude whether or not a system can meet the desired performance/safety constraints in the face of uncertainty using mathematical models

Robust Simulation: Another Perspective

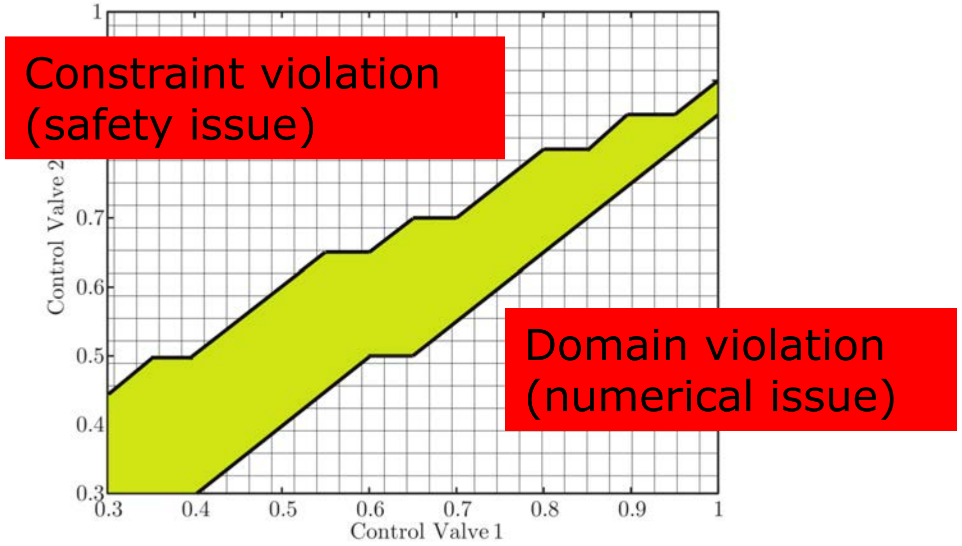
- "Robust Simulation" could also be viewed through the modeler's lens

Robust Simulation: Another Perspective

- "Robust Simulation" could also be viewed through the modeler's lens
  - Modeling and simulation of systems often requires changing parameter values and/or model libraries and the solver then fails to converge

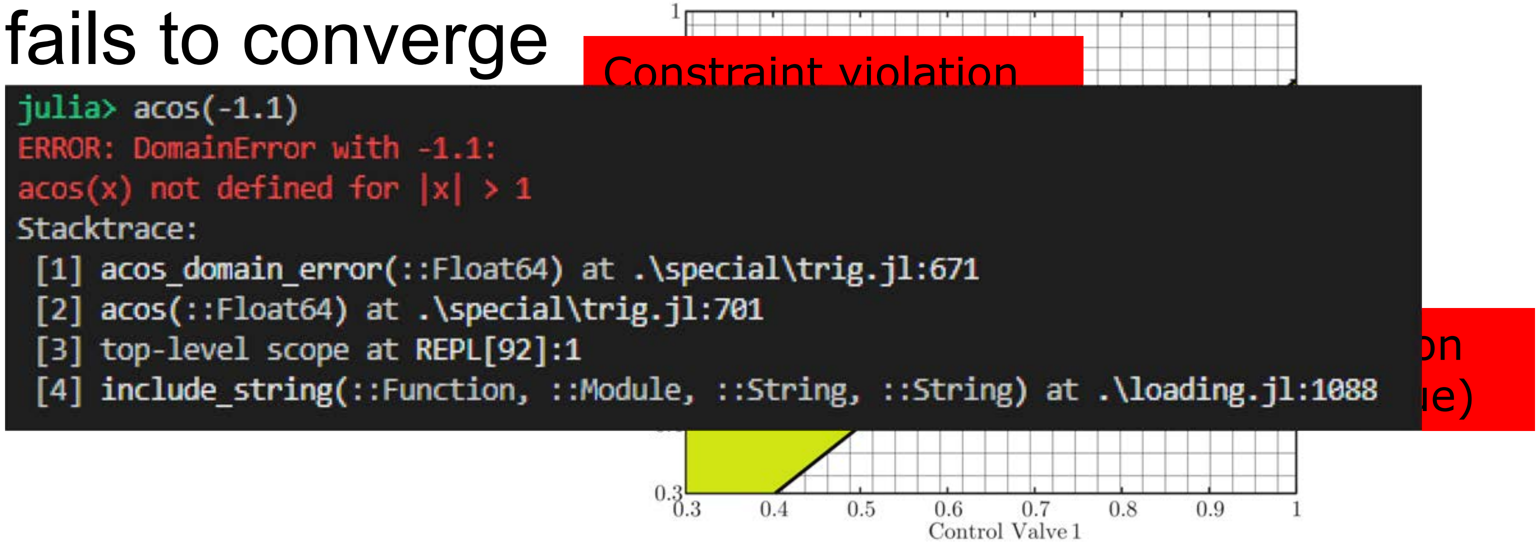
Robust Simulation: Another Perspective

- "Robust Simulation" could also be viewed through the modeler's lens
  - Modeling and simulation of systems often requires changing parameter values and/or model libraries and the solver then fails to converge



Robust Simulation: Another Perspective

- “Robust Simulation” could also be viewed through the modeler’s lens
  - Modeling and simulation of systems often requires changing parameter values and/or model libraries and the solver then fails to converge



Robust Simulation: Another Perspective

- Numerical infeasibility encountered in algebraic systems

DOI: [10.1002/aic.14447](https://doi.org/10.1002/aic.14447)

AIChE

Worst-Case Design of Subsea Production Facilities Using Semi-Infinite Programming

Matthew D. Stuber, Achim Wechsung, Arul Sundaramoorthy, and Paul I. Barton
Process Systems Engineering Laboratory, Dept. of Chemical Engineering, Massachusetts Institute of Technology, Cambridge, MA 02139

DOI 10.1002/aic.14447
Published online April 3, 2014 in Wiley Online Library (wileyonlinelibrary.com)

The problem of designing novel process systems for deployment in extreme and hostile environments is addressed. Specifically, the process system of interest is a subsea production facility for ultra deepwater oil and gas production. The costs associated with operational failures in deepwater environments are prohibitively high and, therefore, warrant the application of worst-case design strategies. That is, prior to the construction and deployment of a process, a certificate of robust feasibility is obtained for the proposed design. The concept of worst-case design is addressed by formulating the design feasibility problem as a semi-infinite optimization problem with implicit functions embedded. A basic model of a subsea production facility is presented for a case study of rigorous performance and safety verification. Relying on recent advances in global optimization of implicit functions and semi-infinite programming, the design feasibility problem is solved, demonstrating that this approach is effective in addressing the problem of worst-case design of novel process systems. © 2014 American Institute of Chemical Engineers *AIChE J*, 60: 2513–2524, 2014

Keywords: robust design, design under uncertainty, verification, global optimization, semi-infinite programming

Introduction

As oil and gas reserves continue to be depleted from traditional on-land and shallow-water fields, there has been a significant effort made toward production from increasingly more hostile environments such as those in the ultra deepwater—greater than 7500 ft depths—of the Gulf of Mexico. In 2004, a vast deposit of petroleum, known as the "lower tertiary trend," containing 3–15 billion barrels of petroleum, was discovered by Chevron geologists.[1] However, as was demonstrated by BP in 2010 when it suffered a catastrophic failure of its leased ultra deepwater drilling platform—in only about 5000 ft of water—resulting in 11 lives lost and an estimated $30 billion in expenses and five million barrels spilled[2,3] with significant ecological damage, pursuing oil reserves in deepwater environments comes with inherently high risk magnified by a lack of sufficient technology. In this environment, the costs associated with operational failures far outweigh the costs associated with "overdesigning" the process, and so the goal must be to avoid them altogether.

Industry engineers have suggested that the application of traditional floating platforms to ultra deepwater production is too risky. Instead, novel remote compact subsea production facilities are considered a key enabling technology for ultra deepwater oil and gas production. Due to imprecise data and incomplete knowledge of the extreme subsea conditions, among various other factors, it is apparent that uncertainty must be accounted for. Thus, the task of designing such a process system is far from trivial.

Correspondence concerning this article should be addressed to P. I. Barton at pib@mit.edu.

As field conditions are extreme, they are difficult and expensive to recreate in the laboratory, and as building physical pilot plant systems for testing at field conditions is implausible, model-based design must support and complement empirical studies. Furthermore, it is worth mentioning that even if building and deploying pilot systems were a cost-effective approach, they can only be tested under a finite number of conditions, and therefore, no rigorous guarantee of worst-case performance/safety can be verified.

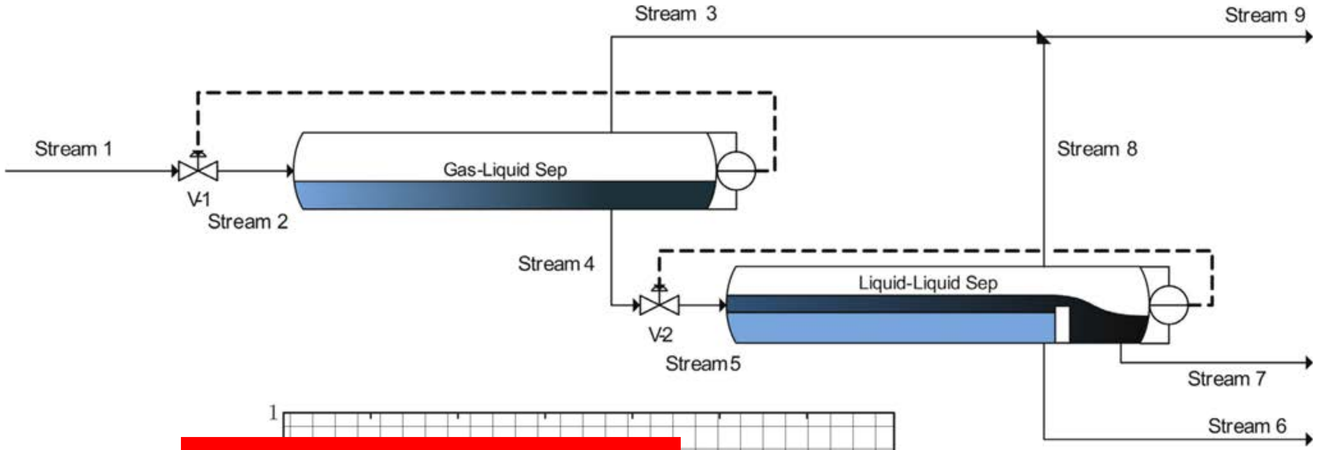
Stuber and Barton[4] previously stated that for these types of systems, the first question a design engineer must address is:

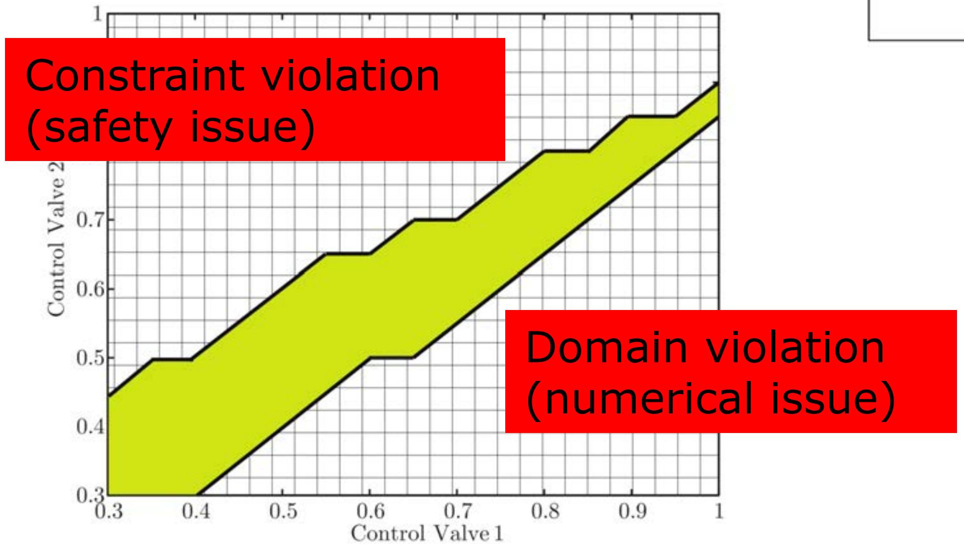
> Given a process model, and taking into account the uncertainty in the model and disturbances to the inputs of the system, do there exist control settings such that, at steady state, the physical system will always meet performance/safety specifications?

This question will be formulated mathematically later and its application to subsea production facilities will be the primary focus of this article. In the following section, the subsea process system model will be presented and the case study will be set up.

Model and Case Study

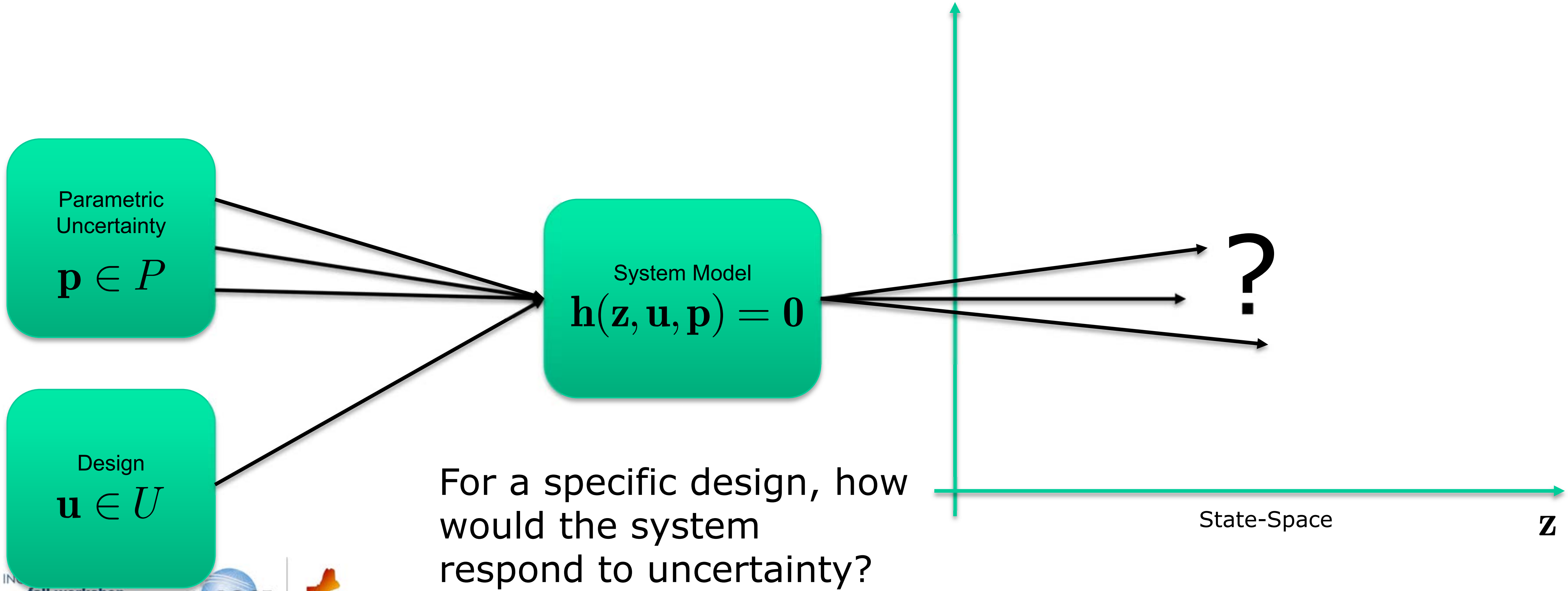
The subsea separator is considered to be at the heart of subsea production facilities because it is the key process system for performing upstream phase separation as material is being produced from the wellhead. In the steady-state model presented here, it is considered that a three-phase mixture of oil/water/gas is being sufficiently separated to allow for reinjection of the water back into the environment and the production of separate oil and gas





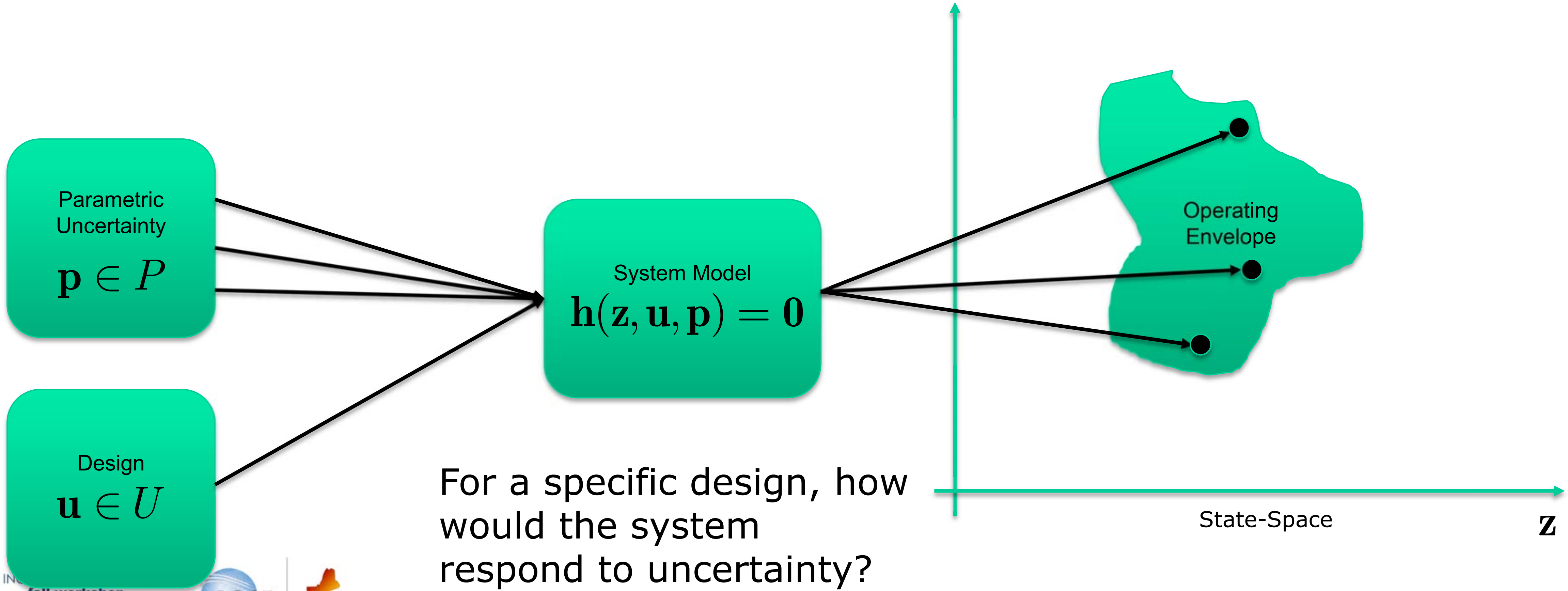
Robust Simulation of Mechanistic Models

Accounting for Uncertainty



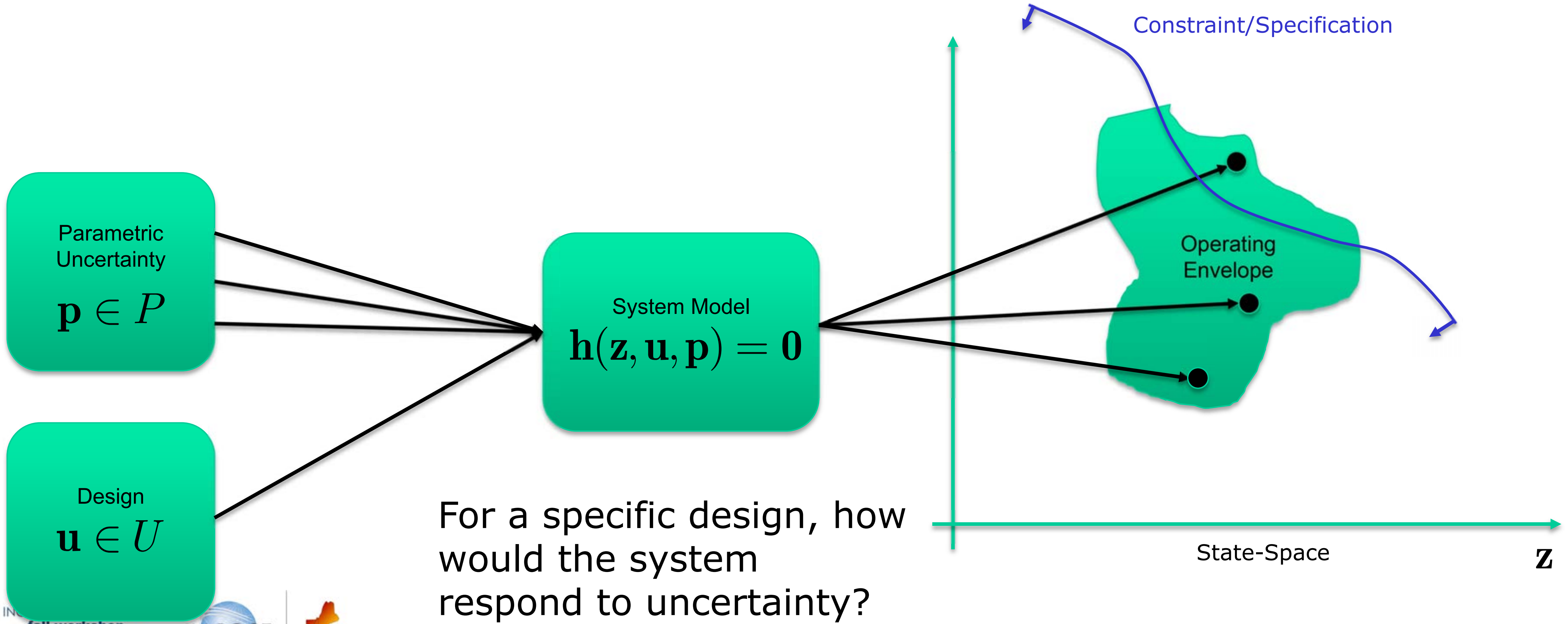
For a specific design, how would the system respond to uncertainty?

Accounting for Uncertainty



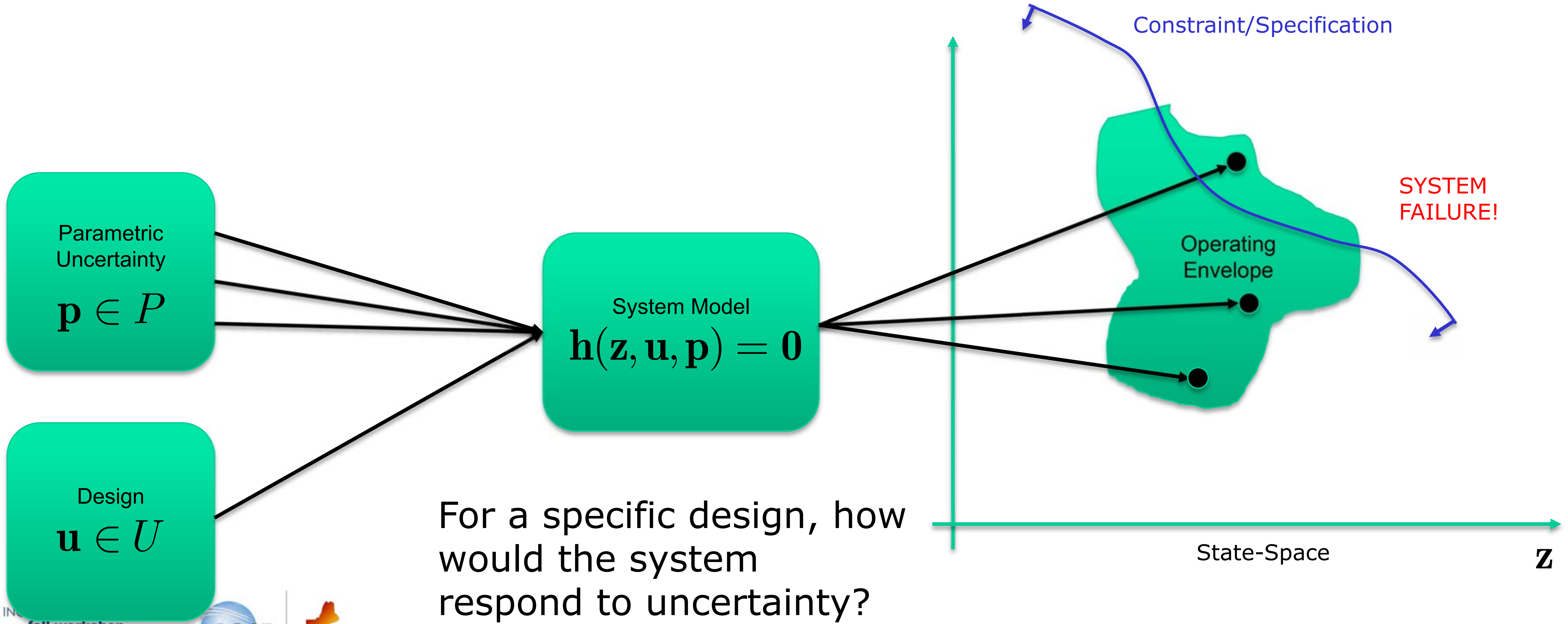
For a specific design, how would the system respond to uncertainty?

Accounting for Uncertainty



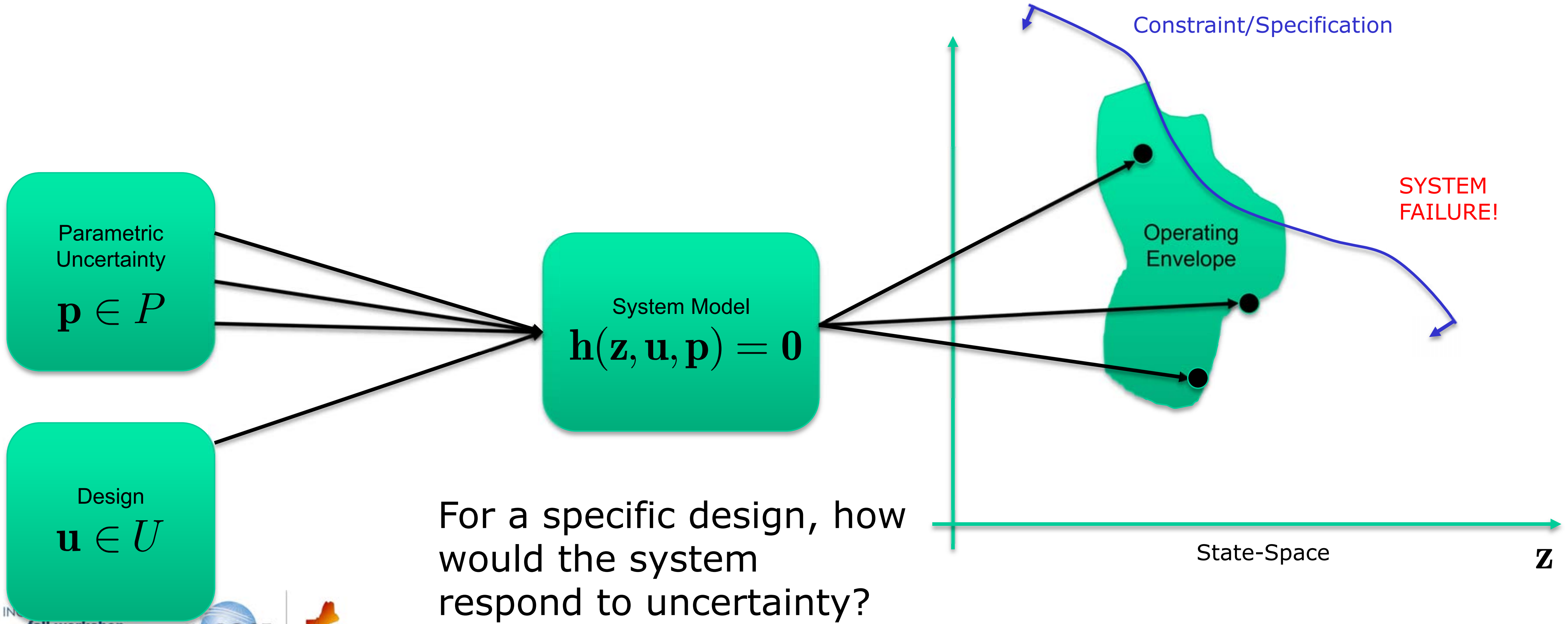
For a specific design, how would the system respond to uncertainty?

Accounting for Uncertainty



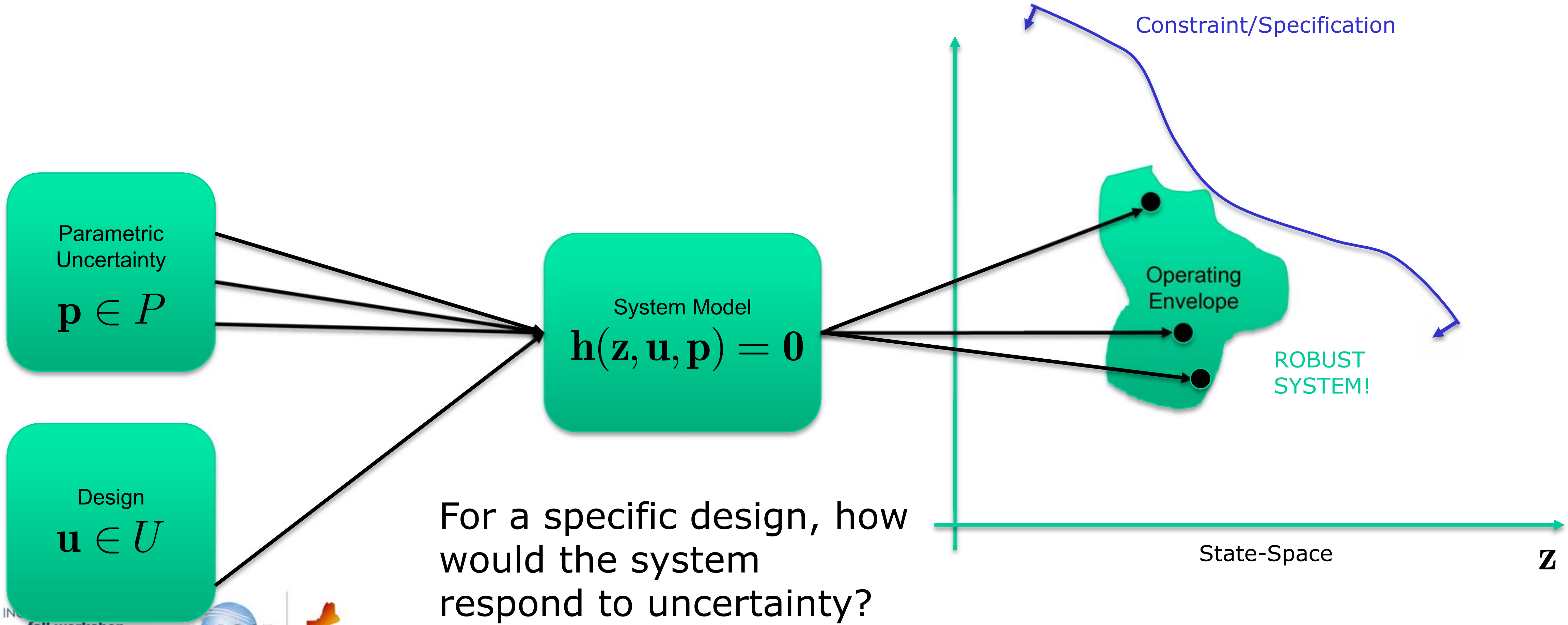
For a specific design, how would the system respond to uncertainty?

Accounting for Uncertainty



For a specific design, how would the system respond to uncertainty?

Accounting for Uncertainty



For a specific design, how would the system respond to uncertainty?

Accounting for Uncertainty

Research Challenge:
Verifying a system is not robust is as simple as finding a single realization of uncertainty that violates the constraint.

Verifying a system is robust requires simulating infinitely-many realizations of uncertainty and ensuring the system never violates the constraint.

Parametric
Uncertaint

$\mathbf{p} \in P$

Design
$\mathbf{u} \in U$

cation

ST
EM!

$\mathbf{z}$

respond to uncertainty?

Accounting for Uncertainty

- Steady-state vs. dynamical systems models

System Model

$$\mathbf{h}(\mathbf{z}, \mathbf{u}, \mathbf{p}) = \mathbf{0}$$

nonlinear algebraic system

System Model

$$\dot{\mathbf{x}}(\mathbf{u}, \mathbf{p}, t) = \mathbf{f}(\mathbf{x}(\mathbf{u}, \mathbf{p}, t), \mathbf{u}, \mathbf{p}, t)$$

nonlinear ODE system

Accounting for Uncertainty

- Steady-state vs. dynamical systems models

System Model

$$\mathbf{h}(\mathbf{z}, \mathbf{u}, \mathbf{p}) = \mathbf{0}$$

nonlinear algebraic system

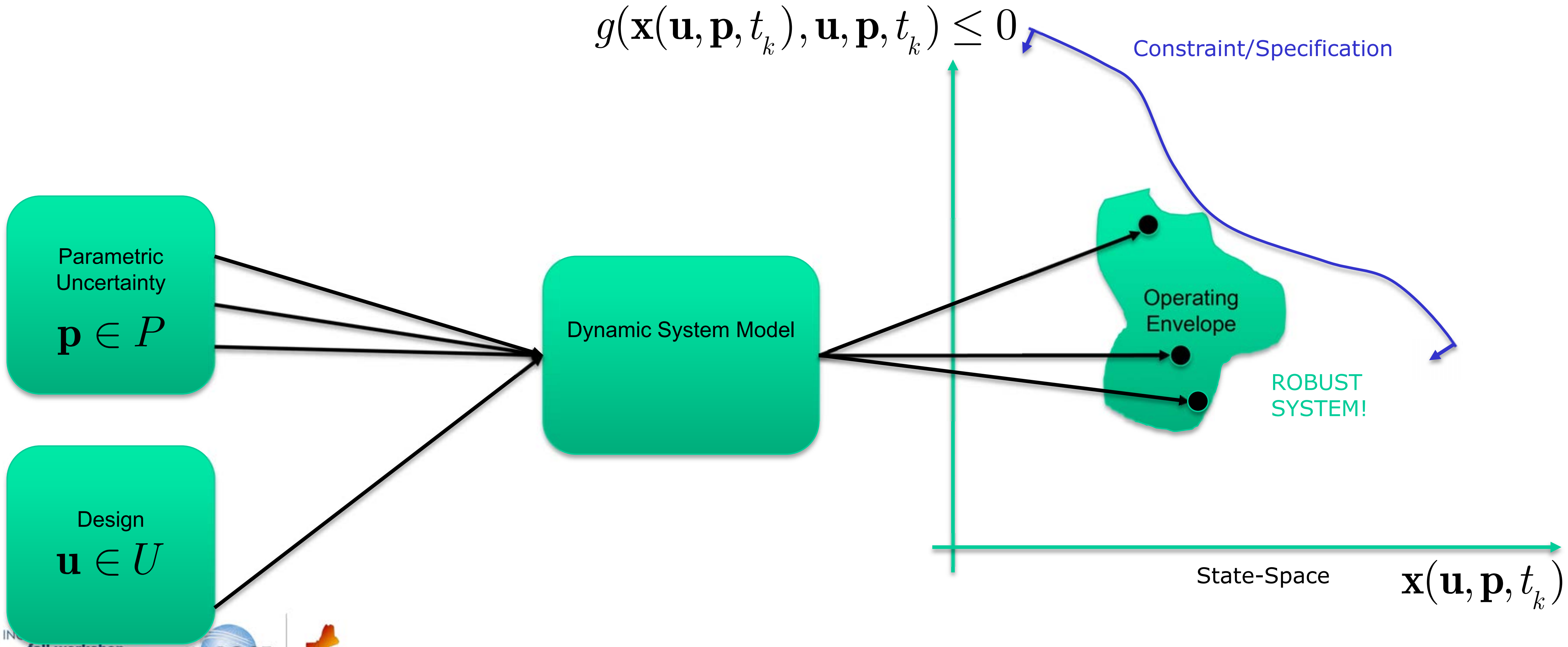
System Model

$$\dot{\mathbf{x}}(\mathbf{u}, \mathbf{p}, t) = \mathbf{f}(\mathbf{x}(\mathbf{u}, \mathbf{p}, t), \mathbf{u}, \mathbf{p}, t)$$

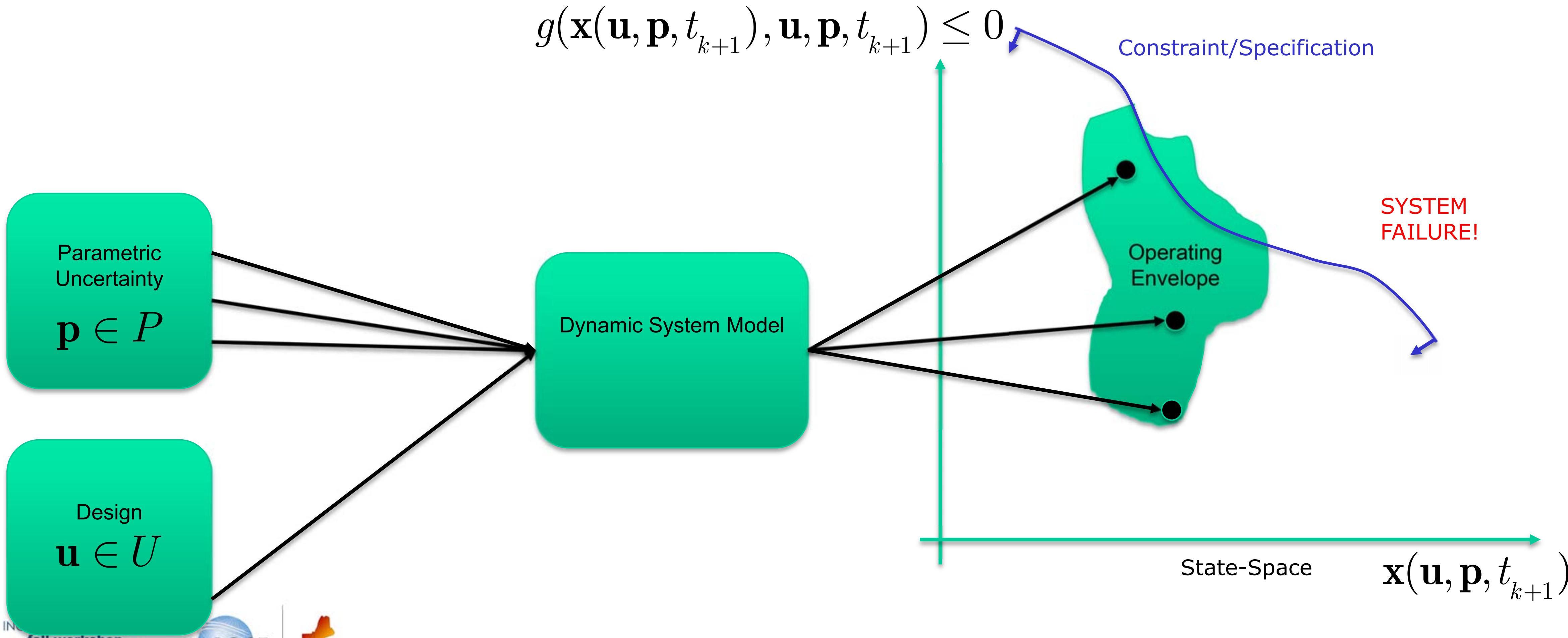
nonlinear ODE system

Now, we must account for the transient response to uncertainty in our design.

Accounting for Uncertainty



Accounting for Uncertainty



Mathematical Preliminaries

- From a design perspective, our objective is to verify performance/safety in the face of (the worst-case) uncertainty over the time horizon.

$$\gamma(\mathbf{u}) = \max_{\mathbf{p}\in P, t\in I} g(\mathbf{x}(\mathbf{u},\mathbf{p},t),\mathbf{u},\mathbf{p},t)$$
$$\text{s.t.}\ \ \dot{\mathbf{x}}(\mathbf{u},\mathbf{p},t) = \mathbf{f}(\mathbf{x}(\mathbf{u},\mathbf{p},t),\mathbf{u},\mathbf{p},t)$$
$$\mathbf{x}(\mathbf{u},\mathbf{p},0) = \mathbf{x}_0(\mathbf{u},\mathbf{p})$$

Mathematical Preliminaries

- From a design perspective, our objective is to verify performance/safety in the face of (the worst-case) uncertainty over the time horizon.

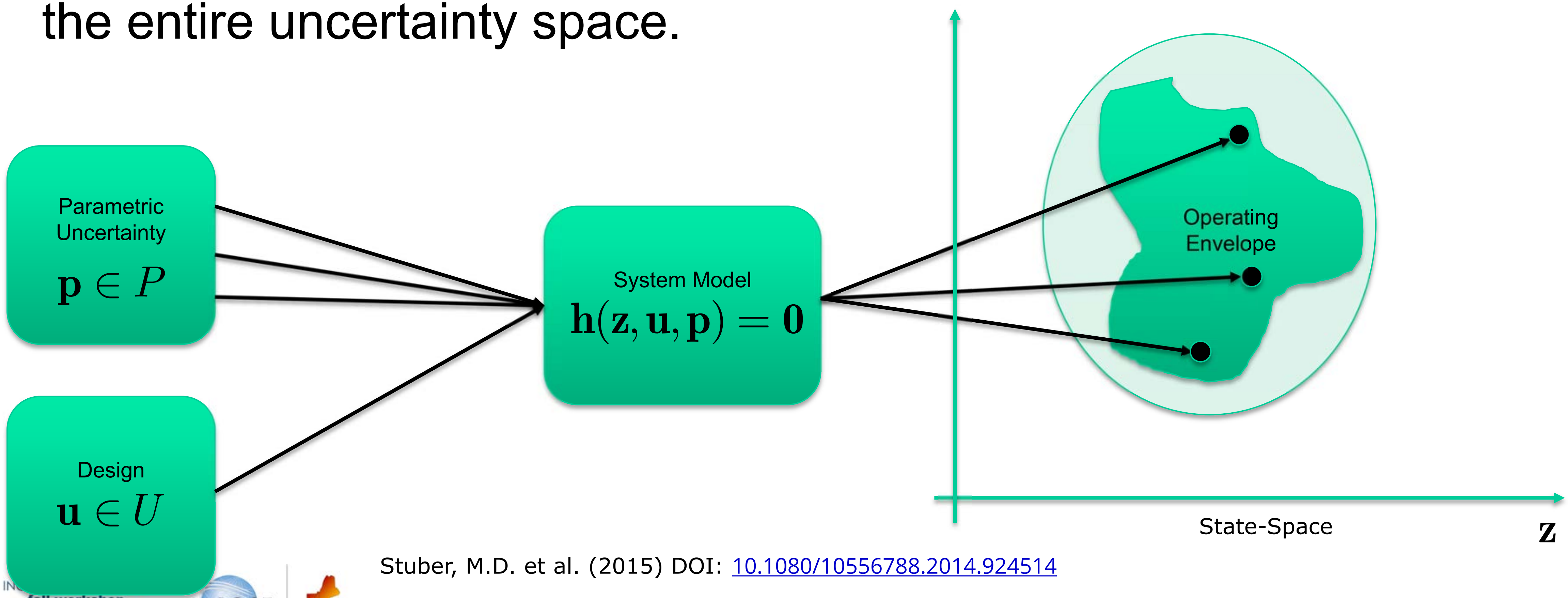
$$\gamma(\mathbf{u}) = \max_{\mathbf{p}\in P, t\in I} g(\mathbf{x}(\mathbf{u},\mathbf{p},t),\mathbf{u},\mathbf{p},t)$$
$$\text{s.t.}\ \ \dot{\mathbf{x}}(\mathbf{u},\mathbf{p},t) = \mathbf{f}(\mathbf{x}(\mathbf{u},\mathbf{p},t),\mathbf{u},\mathbf{p},t)$$
$$\mathbf{x}(\mathbf{u},\mathbf{p},0) = \mathbf{x}_0(\mathbf{u},\mathbf{p})$$

If $\gamma(\mathbf{u}) \leq 0$, we have verified the robustness of our design **u**.

"For a given design, the system does not violate performance/safety at any point in time, even in the face of the worst-case uncertainty"

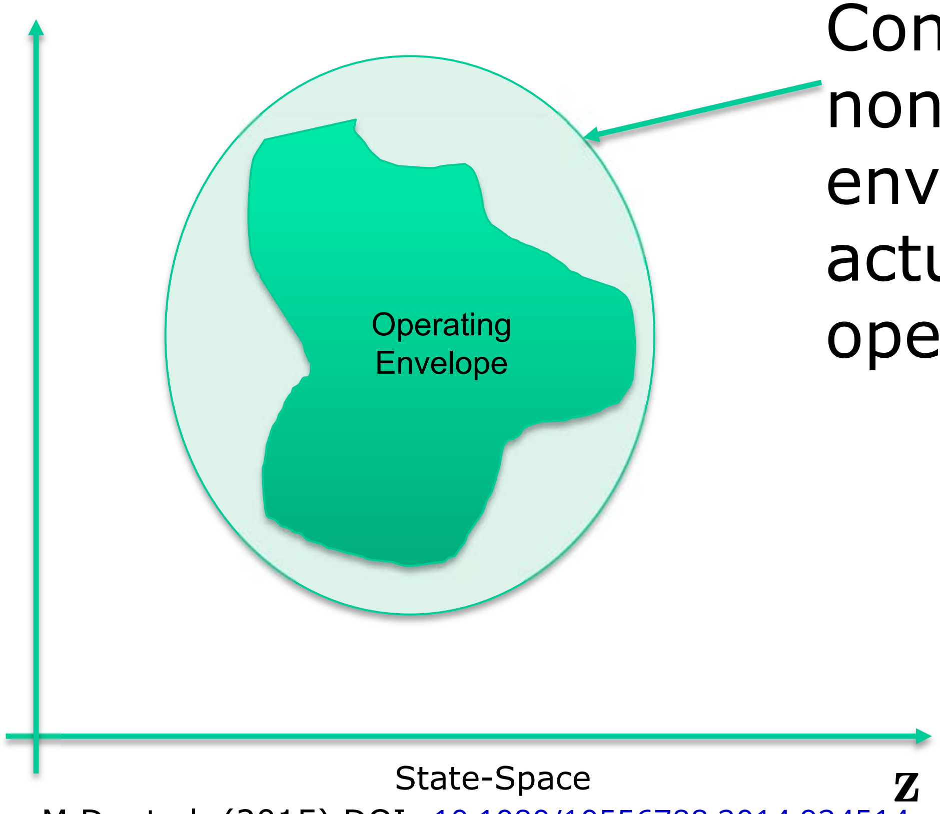
Robust Steady-State Simulation

- Previous developments: a set-valued mapping theory that enables the calculation of rigorous bounds on the states over the entire uncertainty space.



Stuber, M.D. et al. (2015) DOI: [10.1080/10556788.2014.924514](https://doi.org/10.1080/10556788.2014.924514)

Robust Steady-State Simulation



Convex relaxation of nonconvex operating envelope (without actually simulating the operating envelope)

Stuber, M.D. et al. (2015) DOI: [10.1080/10556788.2014.924514](https://doi.org/10.1080/10556788.2014.924514)

Robust Dynamic Simulation

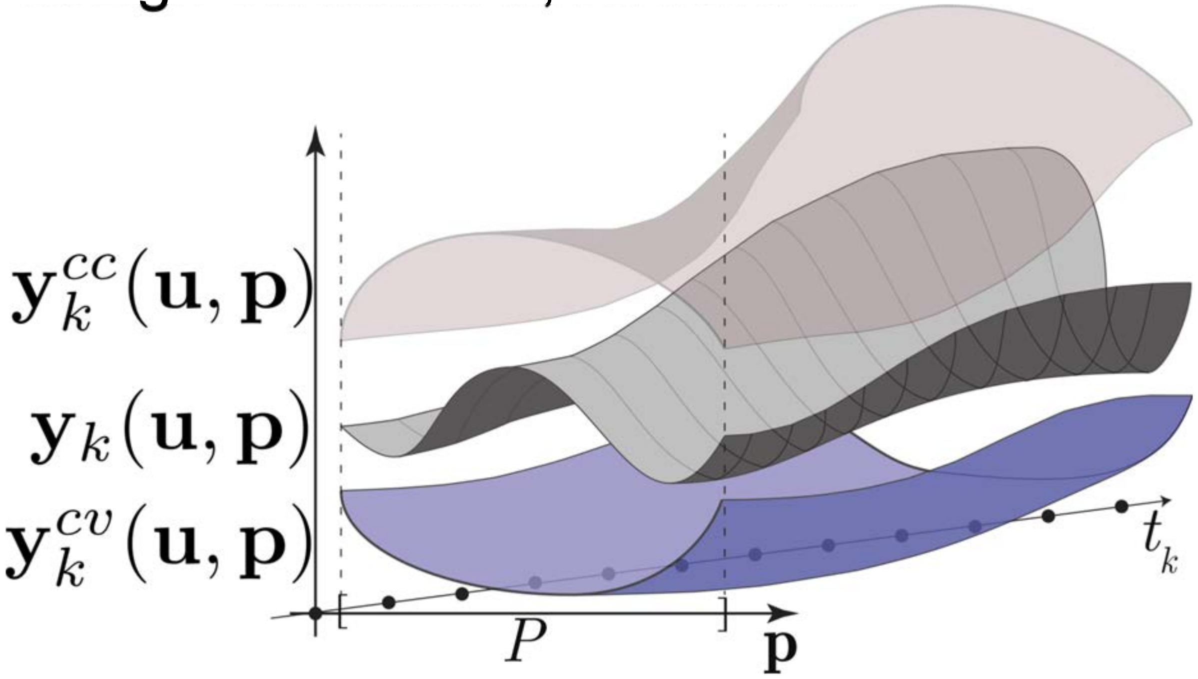
- Our dynamic model is reformulated in the discrete form as a nonlinear algebraic system:

$$\mathbf{h}(\mathbf{y},\mathbf{u},\mathbf{p}) = \begin{pmatrix} \mathbf{y}_0 - \mathbf{x}_0(\mathbf{u},\mathbf{p}) \\ \mathbf{y}_1 - \mathbf{y}_0 - h\mathbf{f}(\mathbf{y}_1,\mathbf{u},\mathbf{p},t_1) \\ \vdots \\ \mathbf{y}_K - \mathbf{y}_{K-1} - h\mathbf{f}(\mathbf{y}_K,\mathbf{u},\mathbf{p},t_K) \end{pmatrix} = \mathbf{0}$$

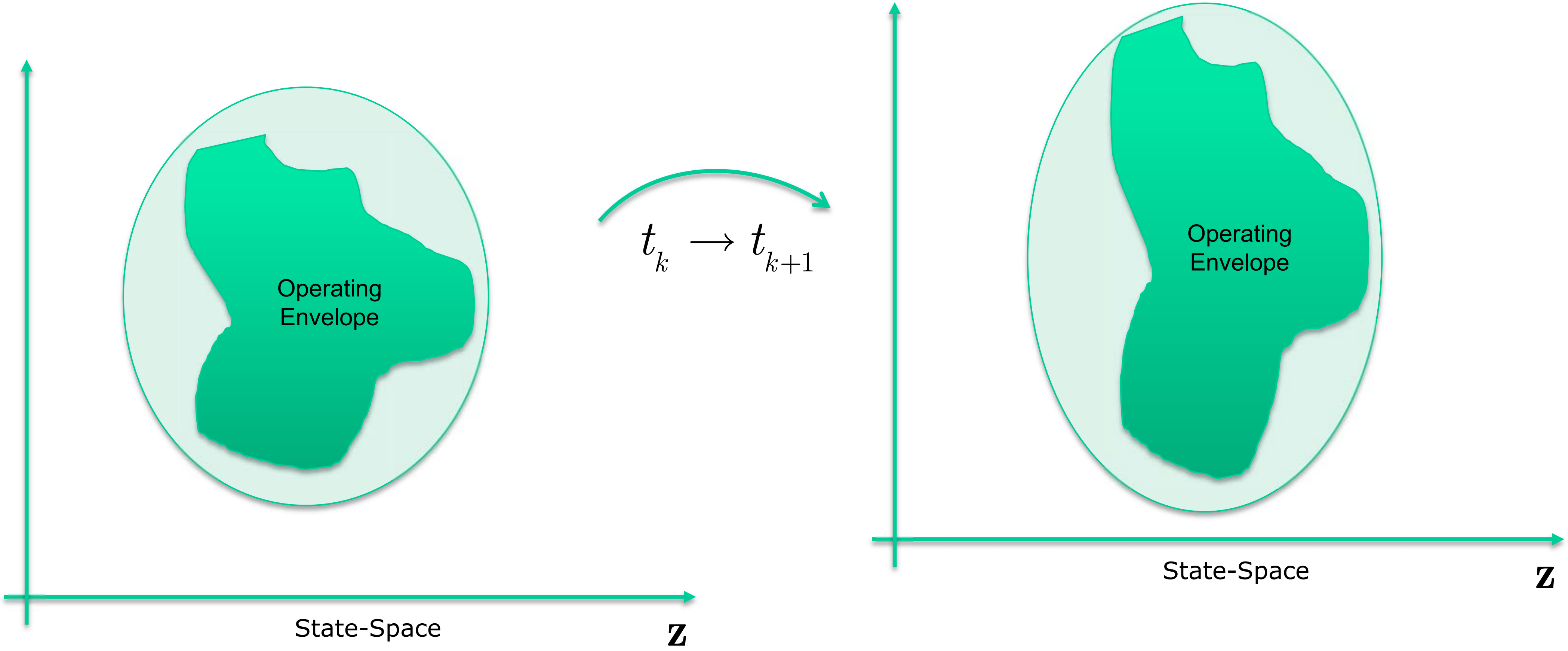
$$\mathbf{h} : \mathbb{R}^{n_x(K+1)} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_p} \to \mathbb{R}^{n_x(K+1)}$$

Robust Dynamic Simulation

Apply our theory for robust dynamic simulation to our system to calculate rigorous bounds on the state variables over the range of uncertainty variables **p** and design variables **u**, forward in time.

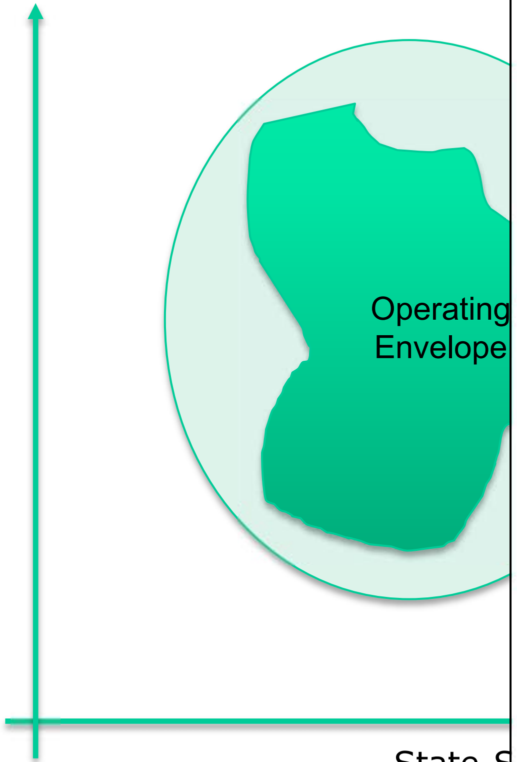


Robust Dynamic Simulation



DOI:10.1002/aic.16836

Robu ion



Received: 20 March 2019 | Revised: 8 October 2019 | Accepted: 12 October 2019
DOI: 10.1002/aic.16836

FUTURES ISSUE: PROCESS SYSTEMS ENGINEERING

AIChE JOURNAL

Global optimization of stiff dynamical systems

Matthew E. Wilhelm[1] | Anne V. Le[2] | Matthew D. Stuber[1]

[1]Process Systems and Operations Research Laboratory, Department of Chemical and Biomolecular Engineering, University of Connecticut, Storrs, Mansfield, Connecticut
[2]Department of Chemical Engineering, Texas A&M University, College Station, Texas

Correspondence
Matthew D. Stuber, PhD, 191 Auditorium Road, Unit 3222, Storrs, CT 06269-3222.
Email: stuber@alum.mit.edu

Present address
Anne V. Le, Department of Chemical Engineering, University of Massachusetts Amherst, Amherst, MA 01003.

Funding information
National Science Foundation, Grant/Award Numbers: 1560072, 1706343, 1932723; University of Connecticut

Abstract
We present a deterministic global optimization method for nonlinear programming formulations constrained by stiff systems of ordinary differential equation (ODE) initial value problems (IVPs). The examples arise from dynamic optimization problems exhibiting both fast and slow transient phenomena commonly encountered in model-based systems engineering applications. The proposed approach utilizes unconditionally stable implicit integration methods to reformulate the ODE-constrained problem into a nonconvex nonlinear program (NLP) with implicit functions embedded. This problem is then solved to global optimality in finite time using a spatial branch-and-bound framework utilizing convex/concave relaxations of implicit functions constructed by a method which fully exploits problem sparsity. The algorithms were implemented in the Julia programming language within the EAGO.jl package and demonstrated on five illustrative examples with varying complexity relevant in process systems engineering. The developed methods enable the guaranteed global solution of dynamic optimization problems with stiff ODE–IVPs embedded.

KEYWORDS
dynamic simulation, global optimization, implicit functions, stiff systems

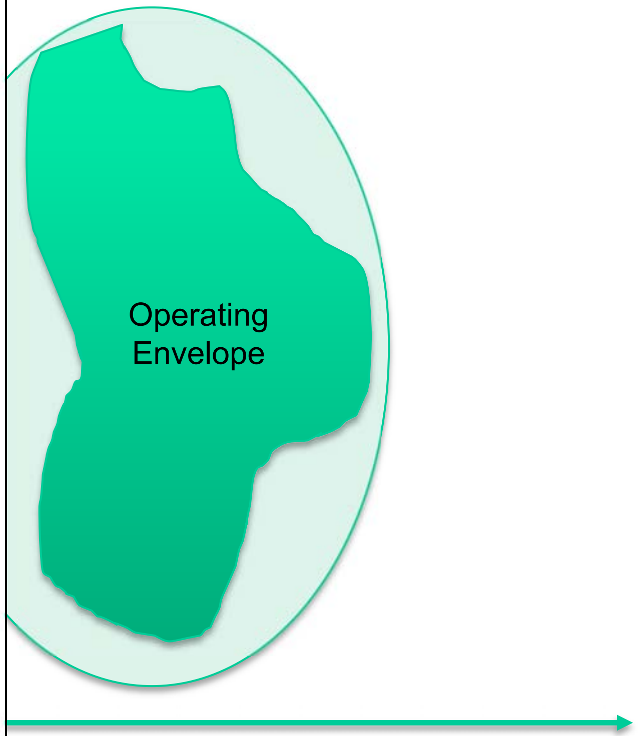
1 | INTRODUCTION

Dynamic optimization problems of the form:

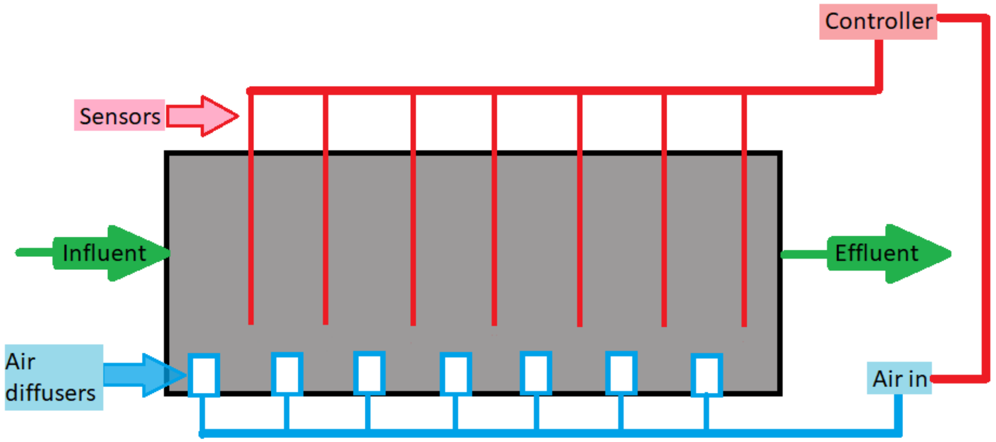
$$\begin{aligned} \phi^* = & \min_{\mathbf{p} \in P \subset \mathbb{R}^{n_p}} \phi(\mathbf{x}(\mathbf{p},t_f),\mathbf{p}) \\ \text{s.t. } & \dot{\mathbf{x}}(\mathbf{p},t) = \mathbf{f}(\mathbf{x}(\mathbf{p},t),\mathbf{p},t), \quad \forall t \in I = [t_0,t_f] \\ & \mathbf{x}(\mathbf{p},t_0) = \mathbf{x}_0(\mathbf{p}) \\ & \mathbf{g}(\mathbf{x}(\mathbf{p},t_f),\mathbf{p}) \leq \mathbf{0} \end{aligned} \tag{1}$$

are of extreme importance to process systems engineers and the broader model-based systems engineering community as they can be formulated for a variety of systems whose transient behavior is of particular interest, from optimal control to mechanistic model validation. The first major complicating detail of the optimization formulation (1) is that it is constrained by a system of ordinary differential equation–initial value problems (ODE–IVPs). Therefore, simply verifying a feasible point requires the solution of a system of ODE–IVPs. The second major complicating detail is that (1) is a nonconvex program, in general, and therefore verifying optimality requires deterministic global optimization. The focus of this paper is on solving (1) to guaranteed global optimality (or declaration of infeasibility). The methods developed in this work are of specific importance when the ODE–IVP system is stiff.

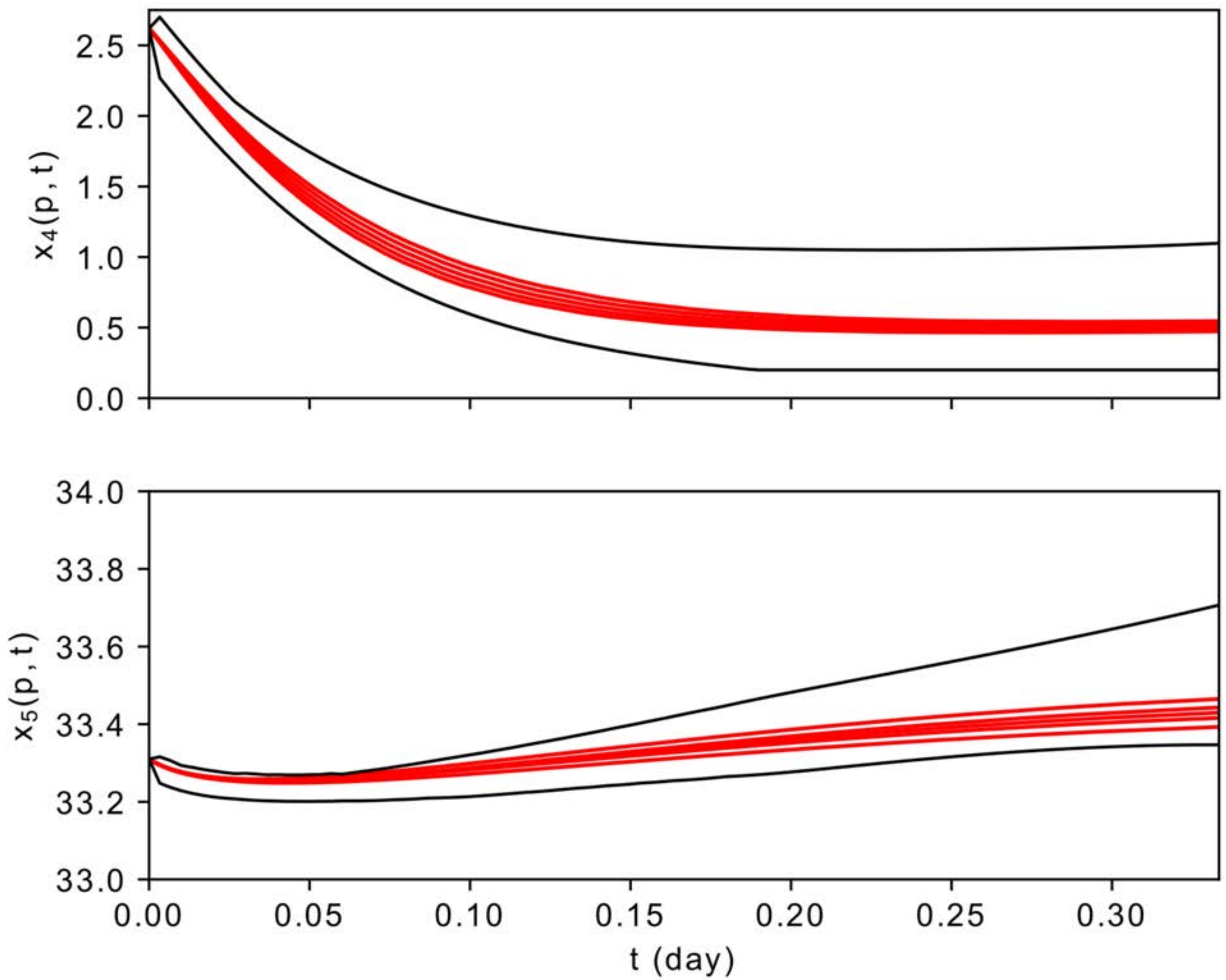
Methods for solving (1) rigorously to global optimality rely on the spatial branch-and-bound (B&B) framework[1,2] or some variant. The B&B algorithm requires the ability to calculate rigorous upper and lower bounds on the global optimal solution value. An upper bound can be calculated by simply evaluating $\phi(\mathbf{x}(\cdot, t_f), \cdot)$ at any feasible point. However, calculating rigorous lower bounds poses significant challenges as this step requires that rigorous and accurate global bounds are known or are readily calculable for all variables and functions of (1). For standard nonconvex nonlinear programs (NLPs) (i.e., without dynamical systems constraints), rigorous lower bounds on the optimal solution value are obtained by calculating convex and concave relaxations of the functions and solving a corresponding convex lower-bounding problem. Applying this approach to a dynamic optimization



Robust Dynamic Simulation



Control of a 9-species biological reaction for wastewater treatment.



Robust Dynamic Simulation

Next steps: verification and validation of robustness

$$
\begin{aligned}
&\min_{\mathbf{u}\in U} \gamma \\
&\text{s.t.}\ \ \gamma \geq \max_{\mathbf{p}\in P, t\in I} g(\mathbf{x}(\mathbf{u},\mathbf{p},t),\mathbf{u},\mathbf{p},t) \\
&\qquad\qquad \text{s.t.}\ \ \dot{\mathbf{x}}(\mathbf{u},\mathbf{p},t) = \mathbf{f}(\mathbf{x}(\mathbf{u},\mathbf{p},t),\mathbf{u},\mathbf{p},t),\ t\in I \\
&\qquad\qquad\qquad \mathbf{x}(\mathbf{u},\mathbf{p},0) = \mathbf{x}_0(\mathbf{u},\mathbf{p})
\end{aligned}
$$

Robust Design

Robust Dynamic Simulation

Next steps: verification and validation of robustness

$$\begin{aligned}
&\min_{\mathbf{u}\in U} \gamma \\
&\text{s.t.} \quad \gamma \geq \max_{\mathbf{p}\in P, t\in I} g(\mathbf{x}(\mathbf{u},\mathbf{p},t),\mathbf{u},\mathbf{p},t) \\
&\qquad\qquad \text{s.t.} \quad \dot{\mathbf{x}}(\mathbf{u},\mathbf{p},t) = \mathbf{f}(\mathbf{x}(\mathbf{u},\mathbf{p},t),\mathbf{u},\mathbf{p},t),\ t\in I \\
&\qquad\qquad\qquad \mathbf{x}(\mathbf{u},\mathbf{p},0) = \mathbf{x}_0(\mathbf{u},\mathbf{p})
\end{aligned}$$

Robust Design

$$\begin{aligned}
&\max_{\mathbf{p}\in P} \gamma \\
&\text{s.t.} \quad \gamma \geq \min_{\mathbf{u}\in U} g(\mathbf{x}(\mathbf{u},\mathbf{p},t_f),\mathbf{u},\mathbf{p},t_f) \\
&\qquad\qquad \text{s.t.} \quad \dot{\mathbf{x}}(\mathbf{u},\mathbf{p},t) = \mathbf{f}(\mathbf{x}(\mathbf{u},\mathbf{p},t),\mathbf{u},\mathbf{p},t),\ t\in I \\
&\qquad\qquad\qquad \mathbf{x}(\mathbf{u},\mathbf{p},0) = \mathbf{x}_0(\mathbf{u},\mathbf{p})
\end{aligned}$$

Robust Operation

Robust Dynamic Simulation

Next steps: verification and validation of robustness

Find the best design **u** and seek the worst-case realization of uncertainty **p** to see if the system violates the performance/safety specifications.

$$\begin{aligned}
&\max_{\mathbf{p}\in P} \gamma \\
&\text{s.t.}\;\; \gamma \geq \min_{\mathbf{u}\in U} g(\mathbf{x}(\mathbf{u},\mathbf{p},t_f),\mathbf{u},\mathbf{p},t_f) \\
&\qquad\qquad \text{s.t.}\;\; \dot{\mathbf{x}}(\mathbf{u},\mathbf{p},t) = \mathbf{f}(\mathbf{x}(\mathbf{u},\mathbf{p},t),\mathbf{u},\mathbf{p},t),\, t\in I \\
&\qquad\qquad\qquad \mathbf{x}(\mathbf{u},\mathbf{p},0) = \mathbf{x}_0(\mathbf{u},\mathbf{p})
\end{aligned}$$

Robust Design

Robust Operation

Robust Dynamic Simulation

Next steps: verification and validation of robustness

Find the best design **u** and seek the worst-case realization of uncertainty **p** to see if the system violates the performance/safety specifications.

Robust Design

Find the worst-case realization of uncertainty **p** and seek a recourse (control) **u** and to see if the system violates the performance/safety specifications.

Robust Operation

Robust Dynamic Simulation

Next steps: verification and validation of robustness

$$
\begin{aligned}
&\min_{\mathbf{u}\in U} \gamma \\
&\text{s.t. } \gamma \geq \max_{\mathbf{p}\in P, t\in I} g(\mathbf{x}(\mathbf{u},\mathbf{p},t),\mathbf{u},\mathbf{p},t) \\
&\qquad \text{s.t. } \dot{\mathbf{x}}(\mathbf{u},\mathbf{p},t) = \mathbf{f}(\mathbf{x}(\mathbf{u},\mathbf{p},t),\mathbf{u},\mathbf{p},t), t \in I \\
&\qquad\qquad \mathbf{x}(\mathbf{u},\mathbf{p},0) = \mathbf{x}_0(\mathbf{u},\mathbf{p})
\end{aligned}
$$

Robust Design

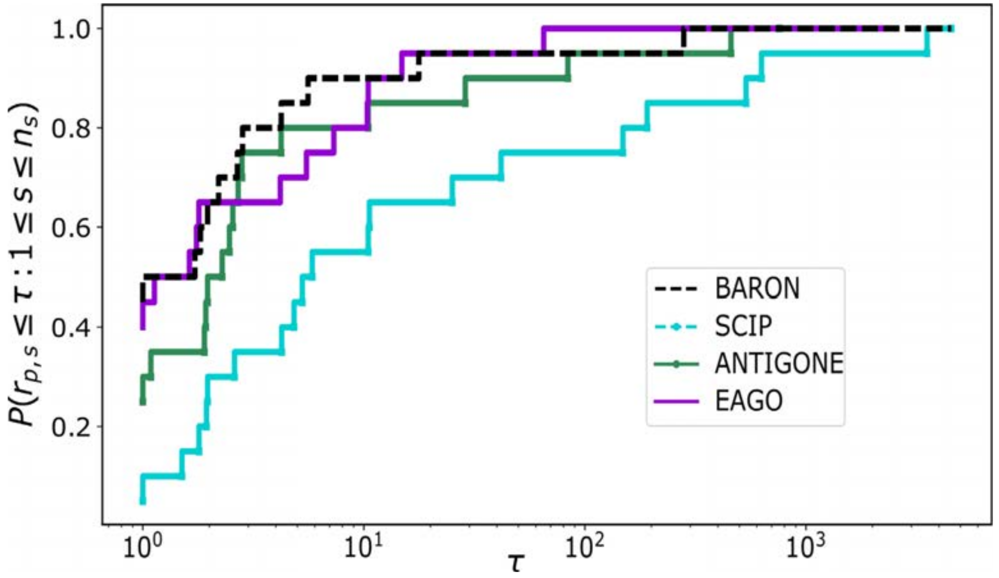
$$
\begin{aligned}
&\max_{\mathbf{p}\in P} \gamma \\
&\text{s.t. } \gamma \geq \min_{\mathbf{u}\in U} g(\mathbf{x}(\mathbf{u},\mathbf{p},t_f),\mathbf{u},\mathbf{p},t_f) \\
&\qquad \text{s.t. } \dot{\mathbf{x}}(\mathbf{u},\mathbf{p},t) = \mathbf{f}(\mathbf{x}(\mathbf{u},\mathbf{p},t),\mathbf{u},\mathbf{p},t), t \in I \\
&\qquad\qquad \mathbf{x}(\mathbf{u},\mathbf{p},0) = \mathbf{x}_0(\mathbf{u},\mathbf{p})
\end{aligned}
$$

Robust Operation

Software Tools

Software Tools

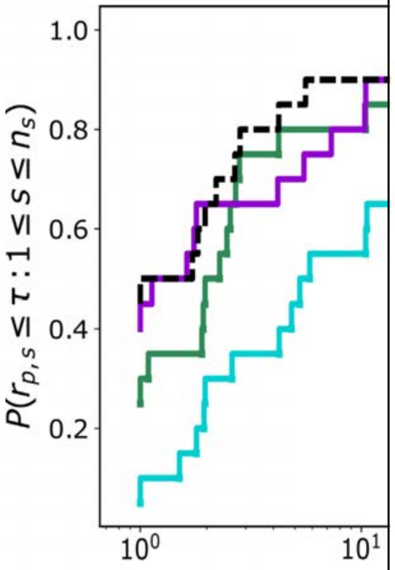
- EAGO.jl: **E**asy **A**dvanced **G**lobal **O**ptimization in Julia.
  - Open-source, competitive with state-of-the-art commercial solvers but much more flexible to account for complicated user-defined functions (UDFs)



DOI:[10.1080/10556788.2020.1786566](https://doi.org/10.1080/10556788.2020.1786566)

Softw

- EAGO.jl: **E**asy **A** ization in Julia.
  - Open-source, co -art commercial solvers but much for complicated user-defined fun



OPTIMIZATION METHODS & SOFTWARE
https://doi.org/10.1080/10556788.2020.1786566

Taylor & Francis
Taylor & Francis Group

Check for updates

EAGO.jl: easy advanced global optimization in Julia

M. E. Wilhelm and M. D. Stuber

Process Systems and Operations Research Laboratory, Department of Chemical and Biomolecular Engineering, University of Connecticut, Storrs, CT, USA

ABSTRACT

An extensible open-source deterministic global optimizer (EAGO) programmed entirely in the Julia language is presented. EAGO was developed to serve the need for supporting higher-complexity user-defined functions (e.g. functions defined implicitly via algorithms) within optimization models. EAGO embeds a first-of-its-kind implementation of McCormick arithmetic in an Evaluator structure allowing for the construction of convex/concave relaxations using a combination of source code transformation, multiple dispatch, and context-specific approaches. Utilities are included to parse user-defined functions into a directed acyclic graph representation and perform symbolic transformations enabling dramatically improved solution speed. EAGO is compatible with a wide variety of local optimizers, the most exhaustive library of transcendental functions, and allows for easy accessibility through the JuMP modelling language. Together with Julia's minimalist syntax and competitive speed, these powerful features make EAGO a versatile research platform enabling easy construction of novel meta-solvers, incorporation and utilization of new relaxations, and extension to advanced problem formulations encountered in engineering and operations research (e.g. multilevel problems, user-defined functions). The applicability and flexibility of this novel software is demonstrated on a diverse set of examples. Lastly, EAGO is demonstrated to perform comparably to state-of-the-art commercial optimizers on a benchmarking test set.

ARTICLE HISTORY
Received 15 January 2020
Accepted 15 June 2020

KEYWORDS
Deterministic global optimization; nonconvex programming; McCormick relaxations; optimization software; branch-and-bound; Julia

2010 MATHEMATICS SUBJECT CLASSIFICATIONS
90C26; 90C34; 90C57; 90C90

1. Introduction and motivation

Mathematical optimization problems are ubiquitous in scientific and technical fields. Applications range from aerospace and chemical process systems to finance. However, even relatively simple physical processes such as mixing, may introduce significant nonconvexity into problem formulations [60]. As such, nonconvex programs often represent the most faithful representations of the system of interest. Multiple approaches have been developed to address these cases. Heuristics such as evolutionary algorithms, may approximate good solutions for select problems. However, heuristics may fail to guarantee that even a feasible

CONTACT M. D. Stuber stuber@alum.mit.edu Process Systems and Operations Research Laboratory, Department of Chemical and Biomolecular Engineering, University of Connecticut, 191 Auditorium Rd, Unit 3222, Storrs, CT 06269-3222, USA

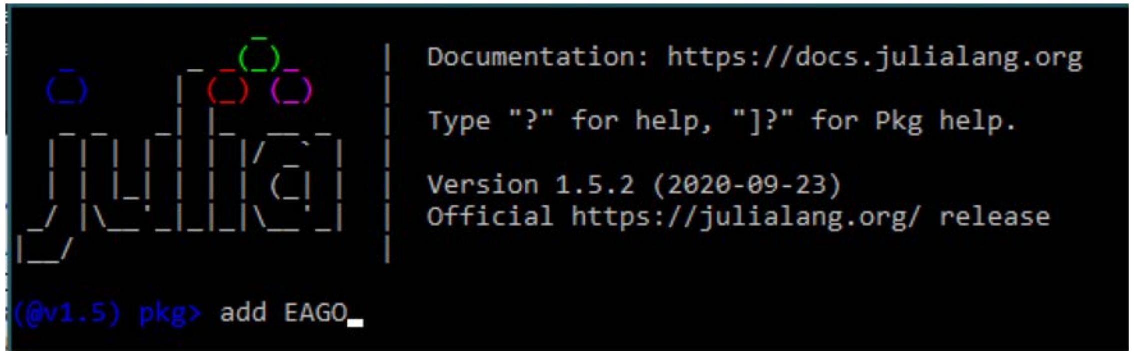
Supplemental data for this article can be accessed here. https://doi.org/10.1080/10556788.2020.1786566

DOI:10.1080/10556788.2020.1786566

Software Tools

- EAGO.jl: **E**asy **A**dvanced **G**lobal **O**ptimization in Julia.

  Software access: registered Julia package

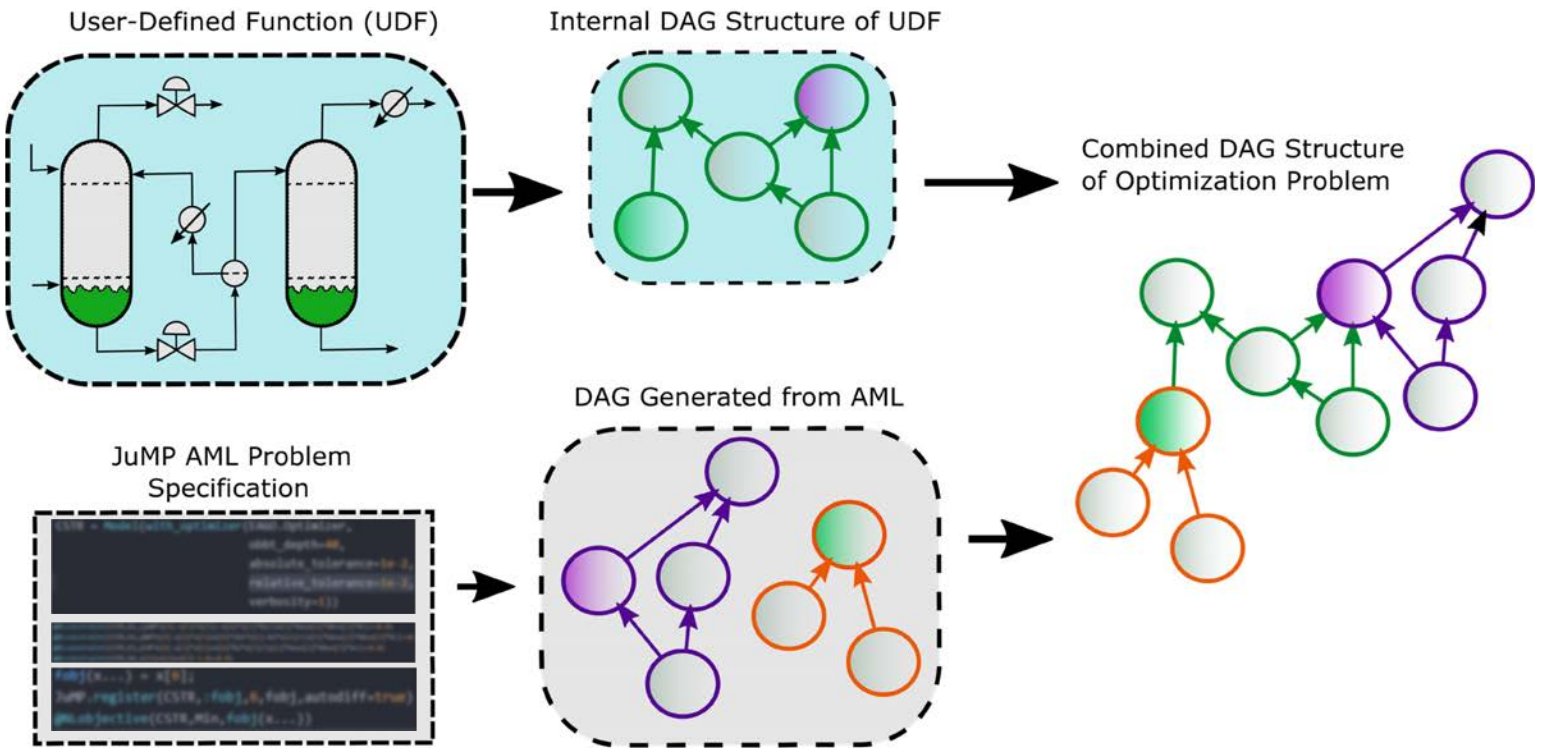


  Github: https://github.com/PSORLab/EAGO.jl

DOI:10.1080/10556788.2020.1786566

Software Tools

- EAGO.jl: **E**asy **A**dvanced **G**lobal **O**ptimization in Julia.

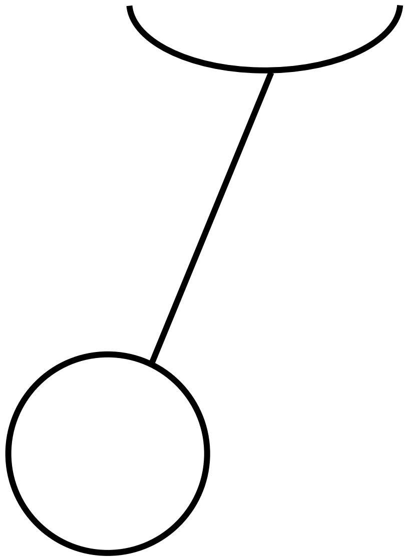


DOI:10.1080/10556788.2020.1786566

Concepts of Hybrid Models

Dynamic NN Prediction Errors

- Consider the anharmonic oscillator (e.g., pendulum)

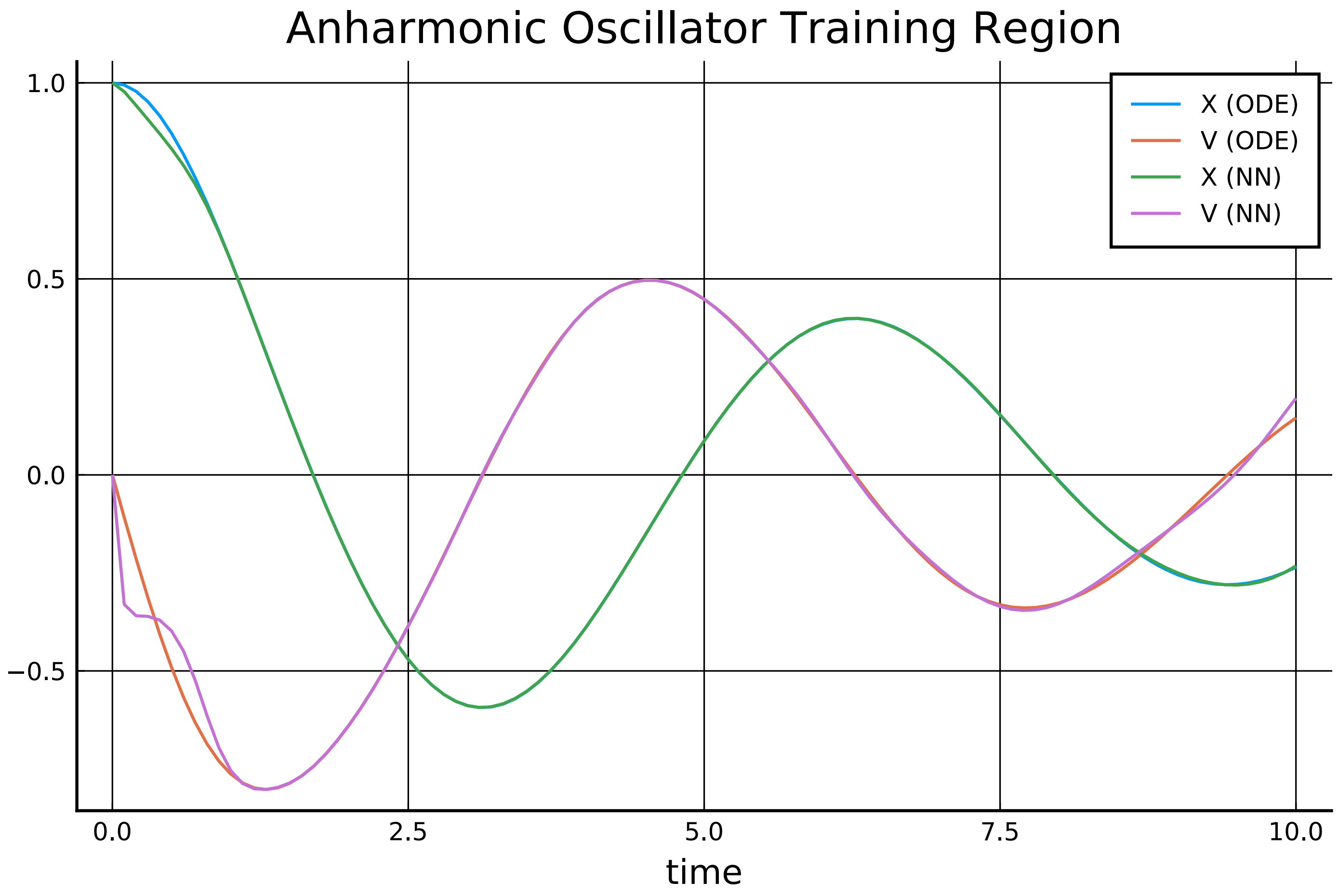


$$\ddot{x}(t) = -kx(t) - \alpha x(t)^3 - \beta \dot{x}(t) - \gamma \dot{x}(t)^3$$
$$x(0) = 1, \dot{x}(0) = 0$$

(Example from DiffEqFlux.jl)

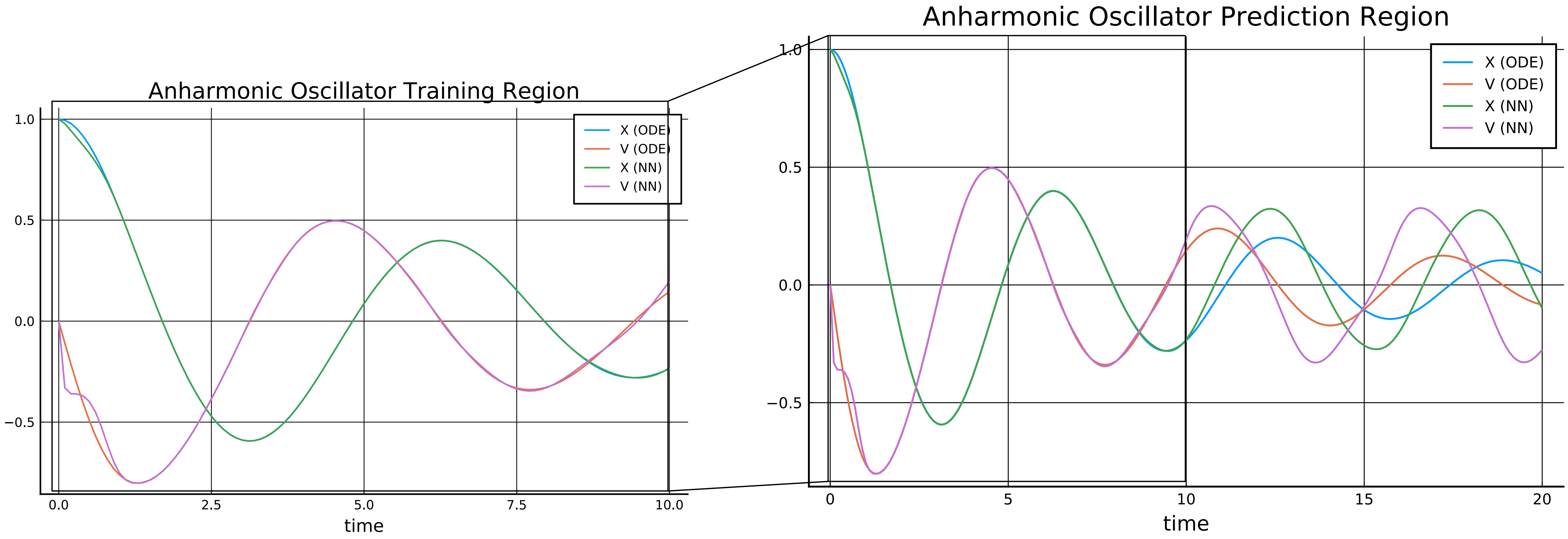
Dynamic NN Prediction Errors

Tensor product layer with 10^{th}-Order Legendre Basis

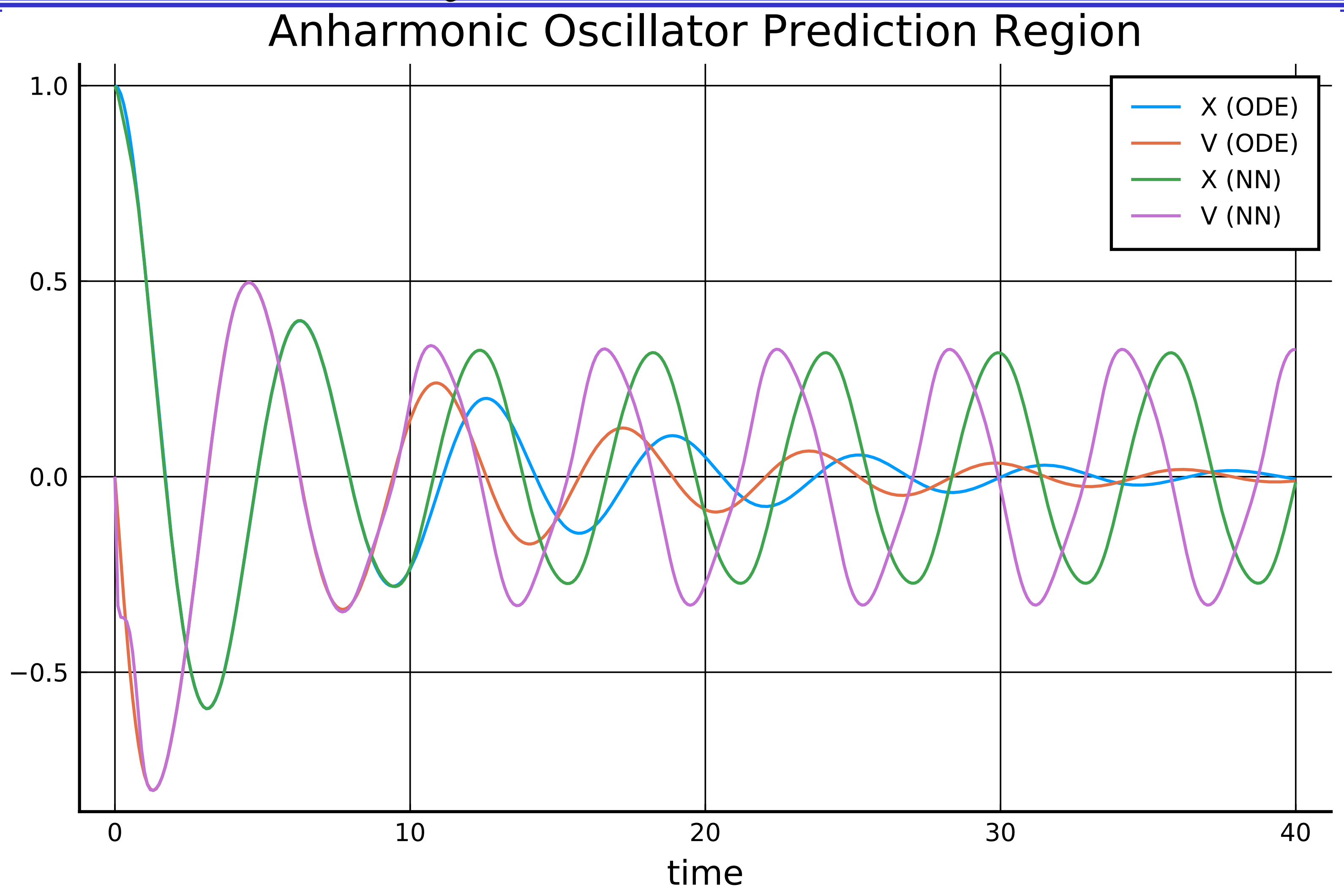


(Example from DiffEqFlux.jl)

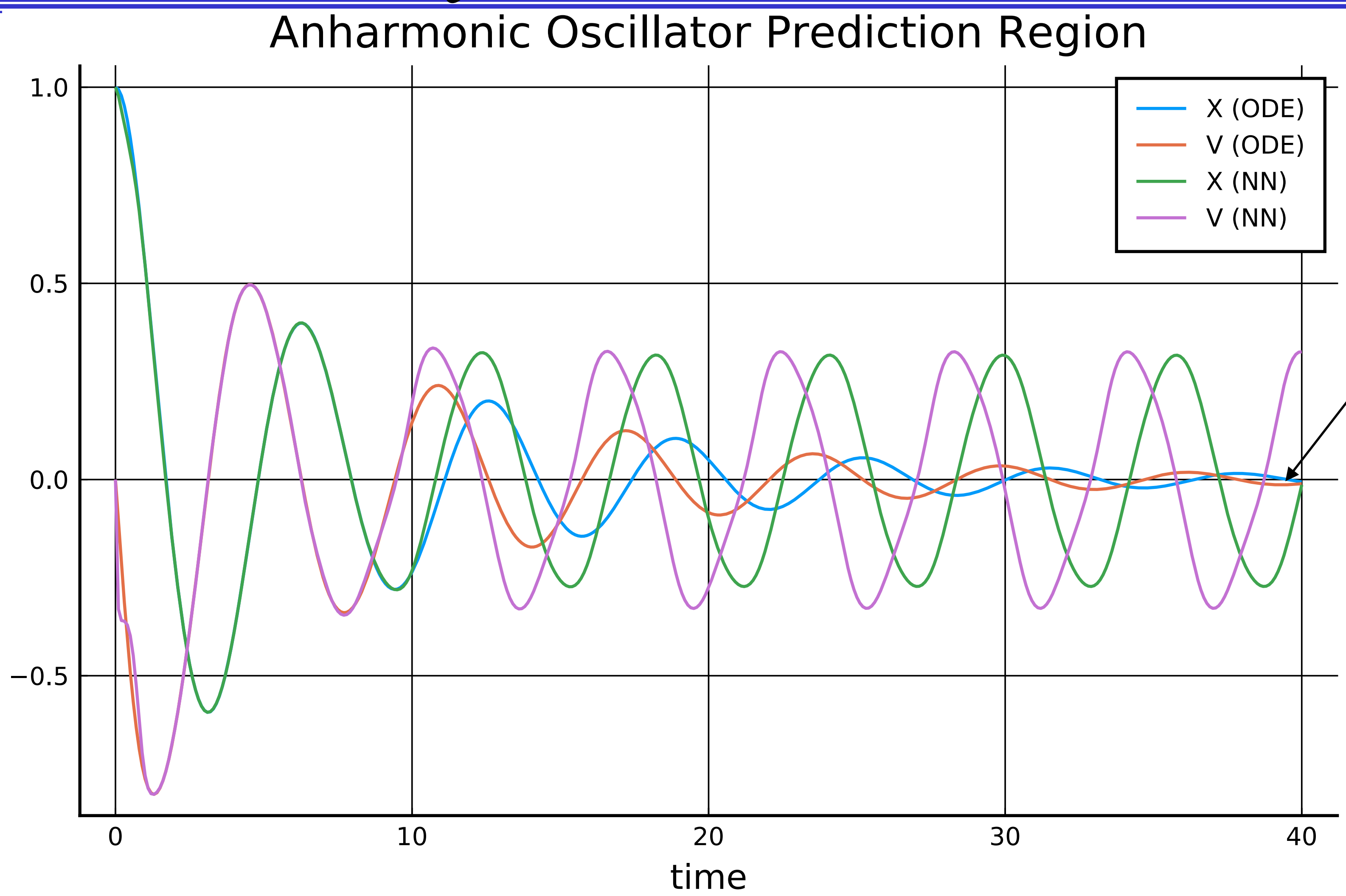
Dynamic NN Prediction Errors



Dynamic NN Prediction Errors

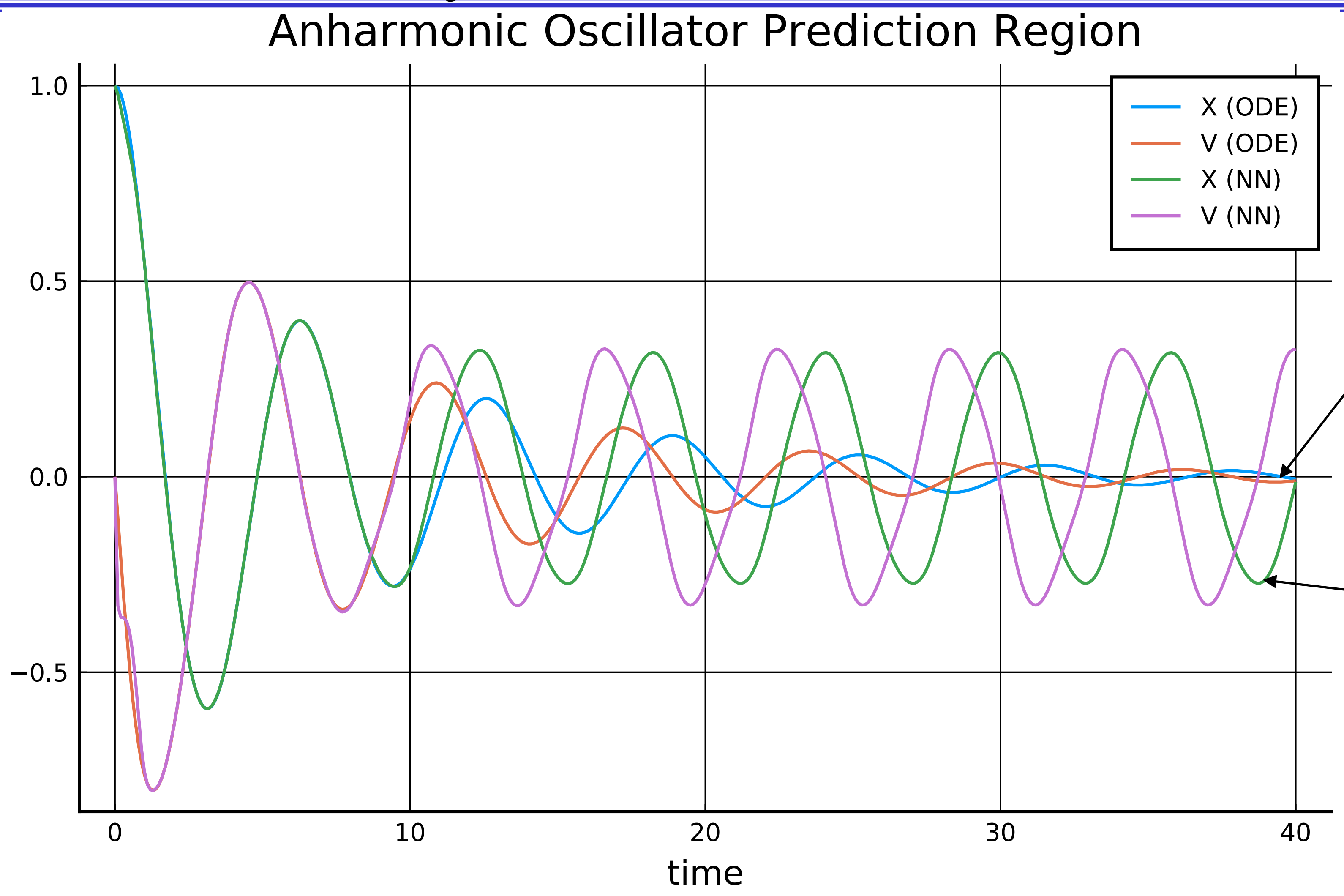


Dynamic NN Prediction Errors



Actual trajectories exhibit rapid decay

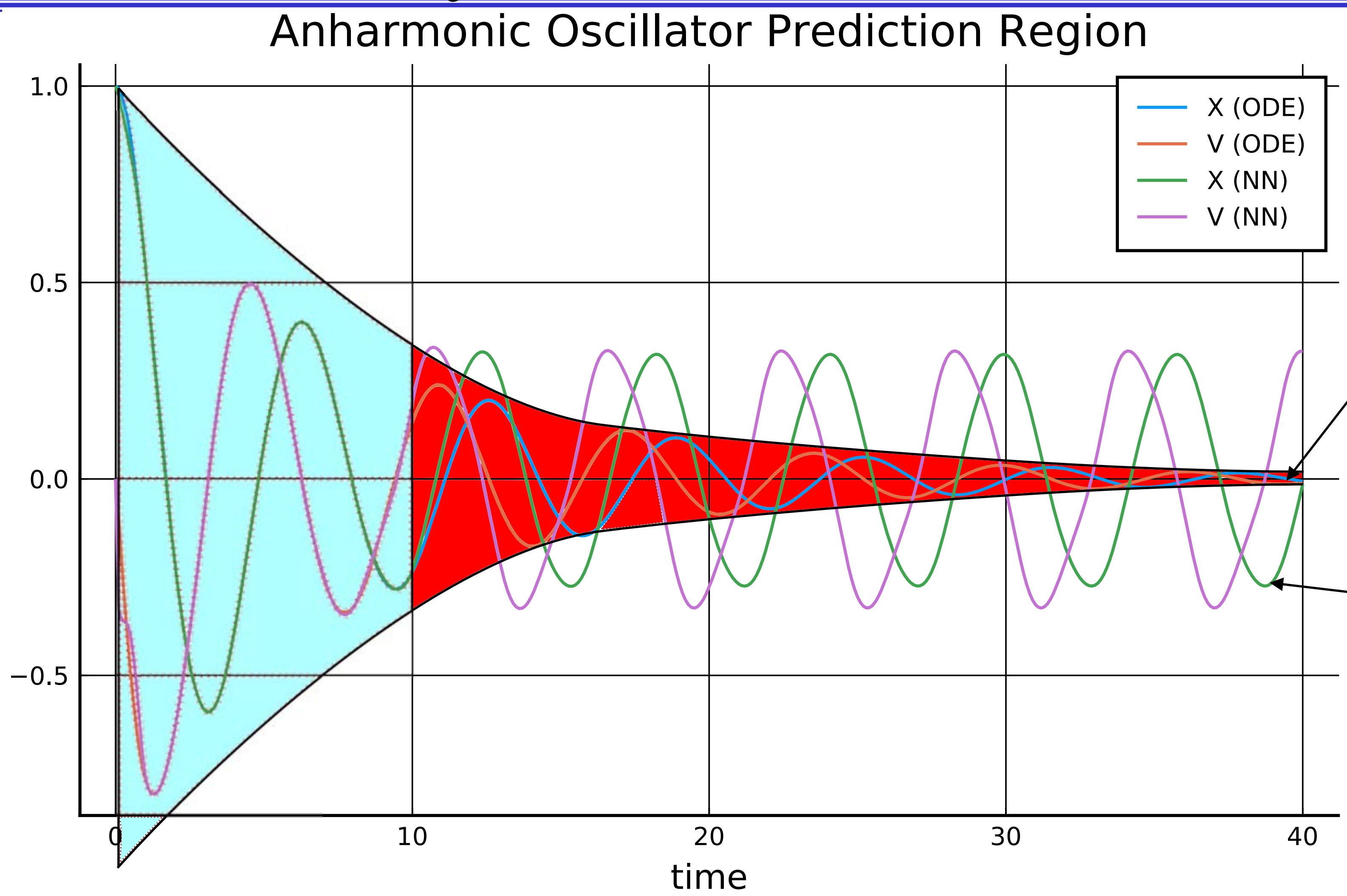
Dynamic NN Prediction Errors



Actual trajectories exhibit rapid decay

ML model fails to capture this behavior

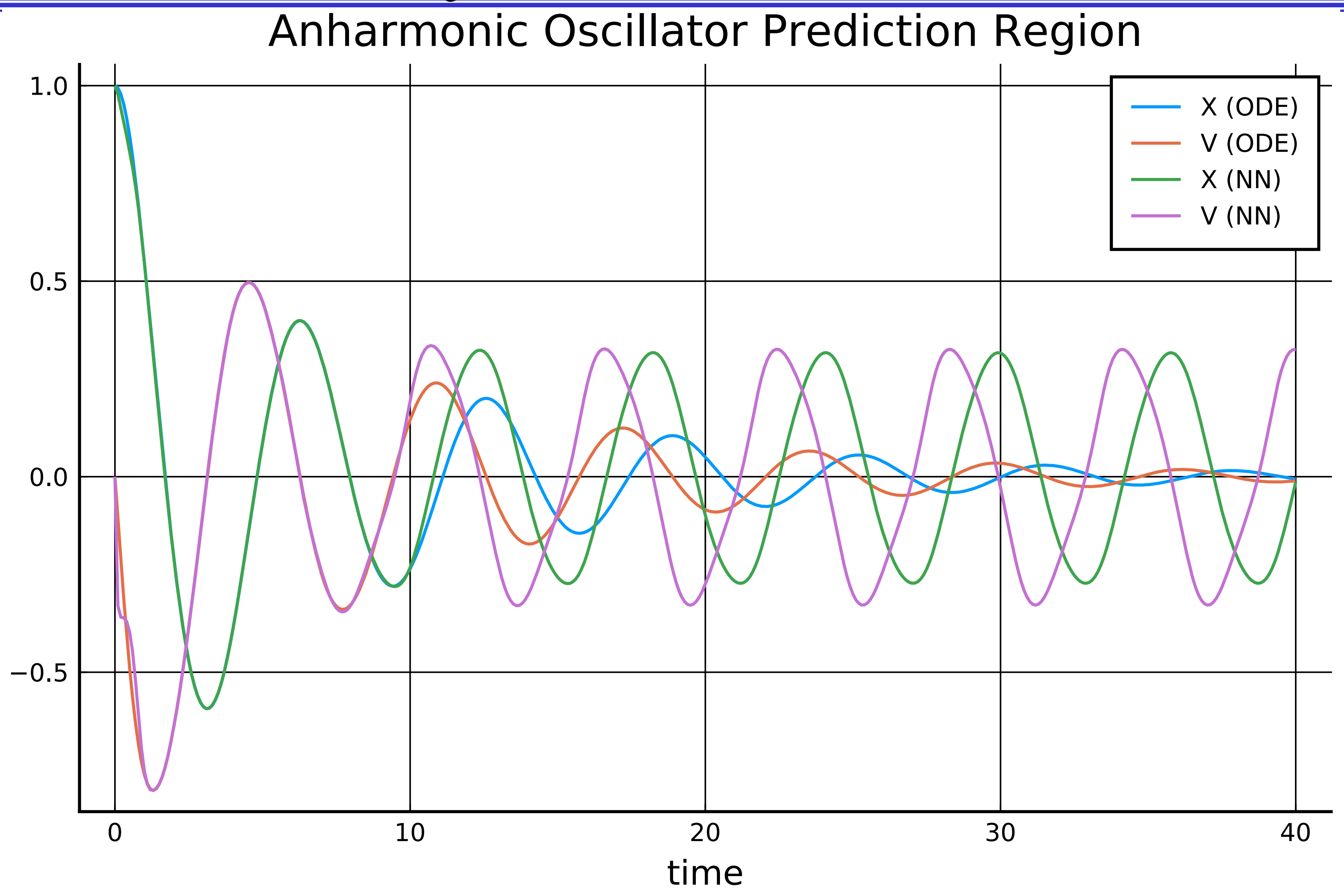
Dynamic NN Prediction Errors



Actual trajectories exhibit rapid decay

ML model fails to capture this behavior

Dynamic NN Prediction Errors



Outstanding ML Challenges

- Absence of theory
- Absence of causal models (correlation not causation)
- Sensitivity to imperfect data
- Computational expense (training)

Begoli, E., Bhattacharya, T., Kusnezov, D. (2019) DOI: 10.1038/s42256-018-0004-1

Outstanding ML Challenges (SE Perspective)

- Lack requirements specification
- Lack design specification
- Lack interpretability (causal relationships)
- Lack robustness

Kuwajima, H., Yasuoka, H., Nakae, T., (2020) DOI: 10.1007/s10994-020-05872-w

Outstanding ML Challenges (SE Perspective)

- Lack requirements specification
- Lack design specification
- Lack interpretability
- Lack robustness

"Greatest impact on conventional system quality models"

Kuwajima, H., Yasuoka, H., Nakae, T., (2020) DOI: 10.1007/s10994-020-05872-w

Outstanding ML Challenges (SE Perspective)

- Lack requirements specification
- Lack
- Lack
- Lack robustness

Research Challenge:
Can we exploit machine learning approaches for safety-critical systems?

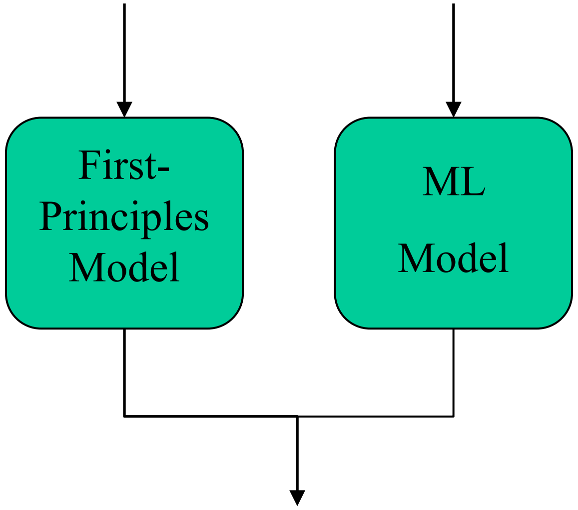
Kuwajima, H., Yasuoka, H., Nakae, T., (2020) DOI: [10.1007/s10994-020-05872-w](10.1007/s10994-020-05872-w)

Hybrid Mechanistic ML Models

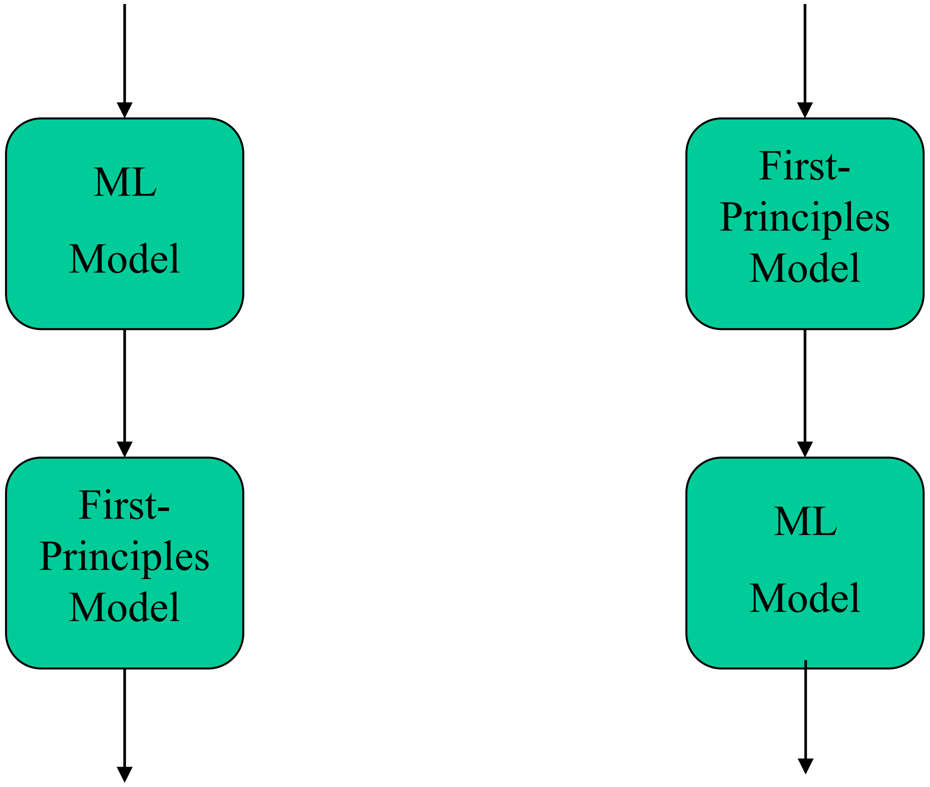
- Not a “new” idea (emergence in 1992)
- Combine aspects of machine learning and mechanistic modeling
- Black-Box → Gray-Box

Hybrid Mechanistic ML Models

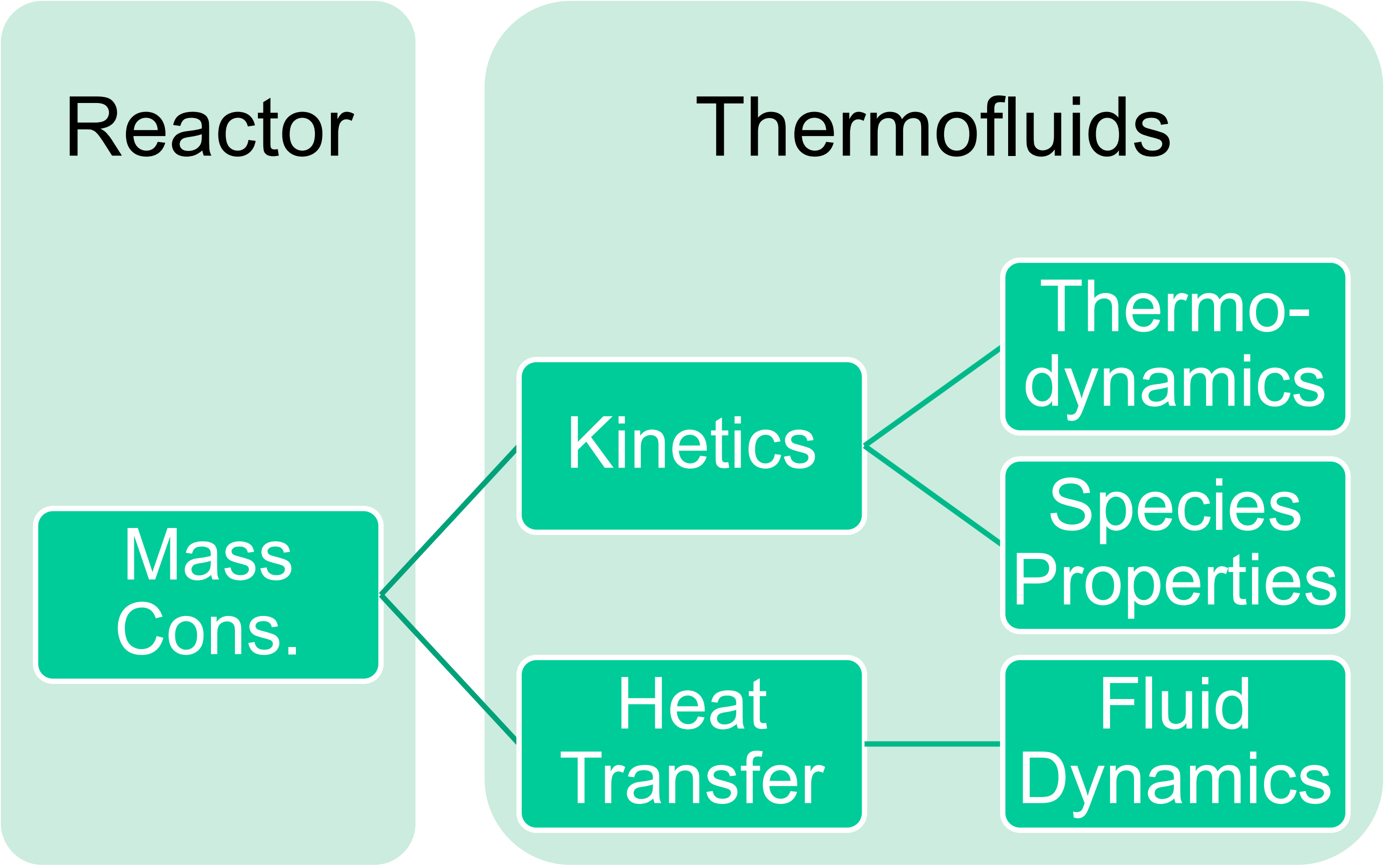
Parallel



Series



Hybrid Mechanistic ML Models



Hybrid Mechanistic ML Models

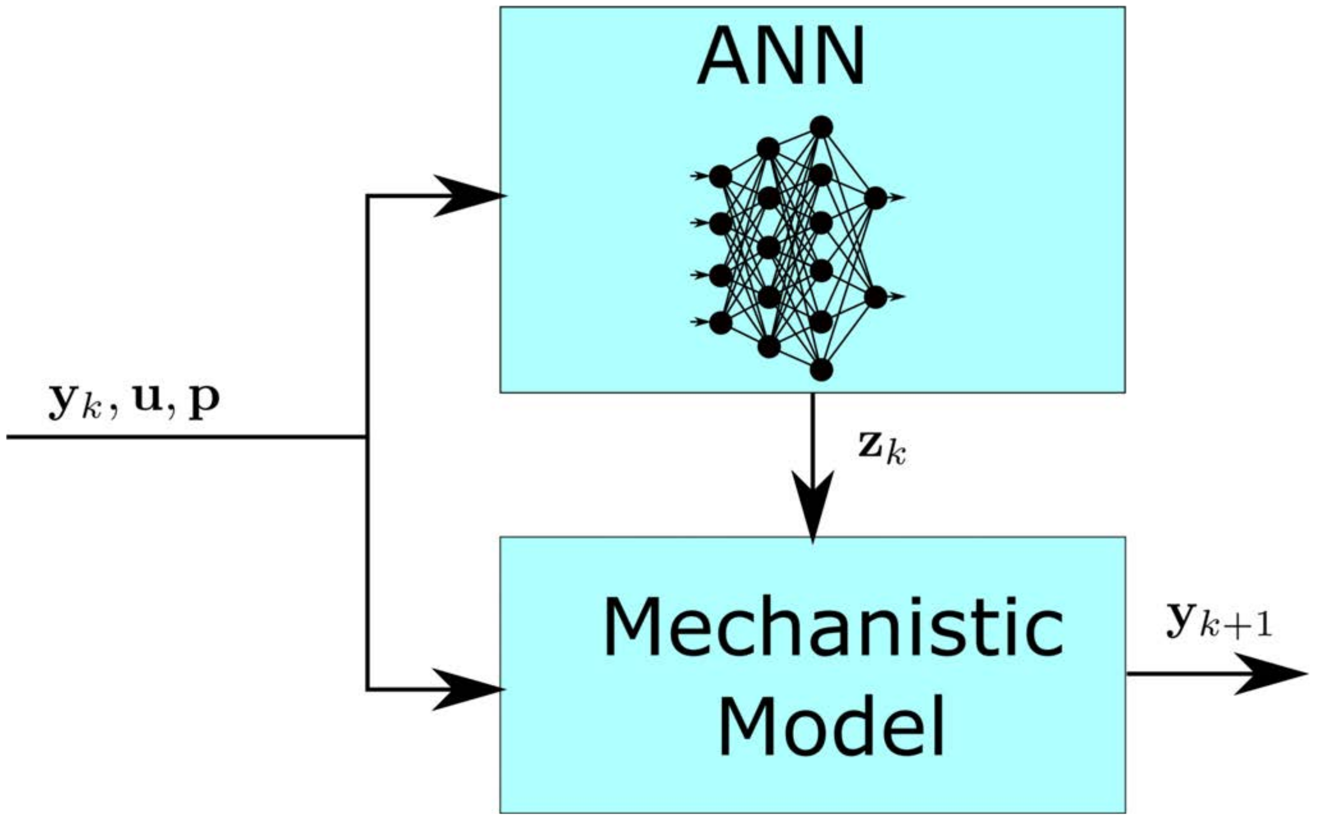
Reactor

Mass Cons.

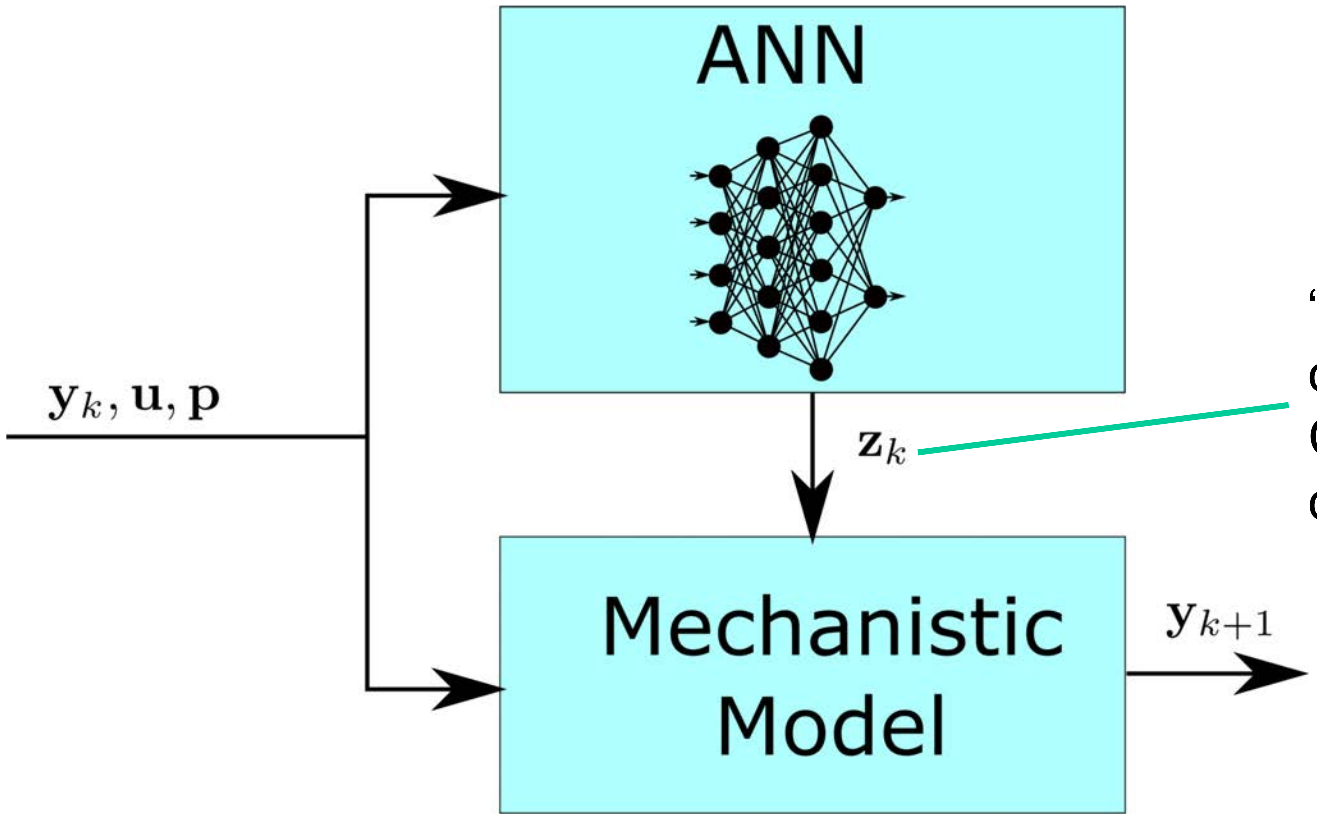
Machine Learning Model

- Empirical model
- Artificial neural net
- Dyn. neural net
- Support vect. machine

Hybrid Mechanistic ML Models



Hybrid Mechanistic ML Models



"Intermediate" states or calculations. E.g., Complex nonlinear dynamics.

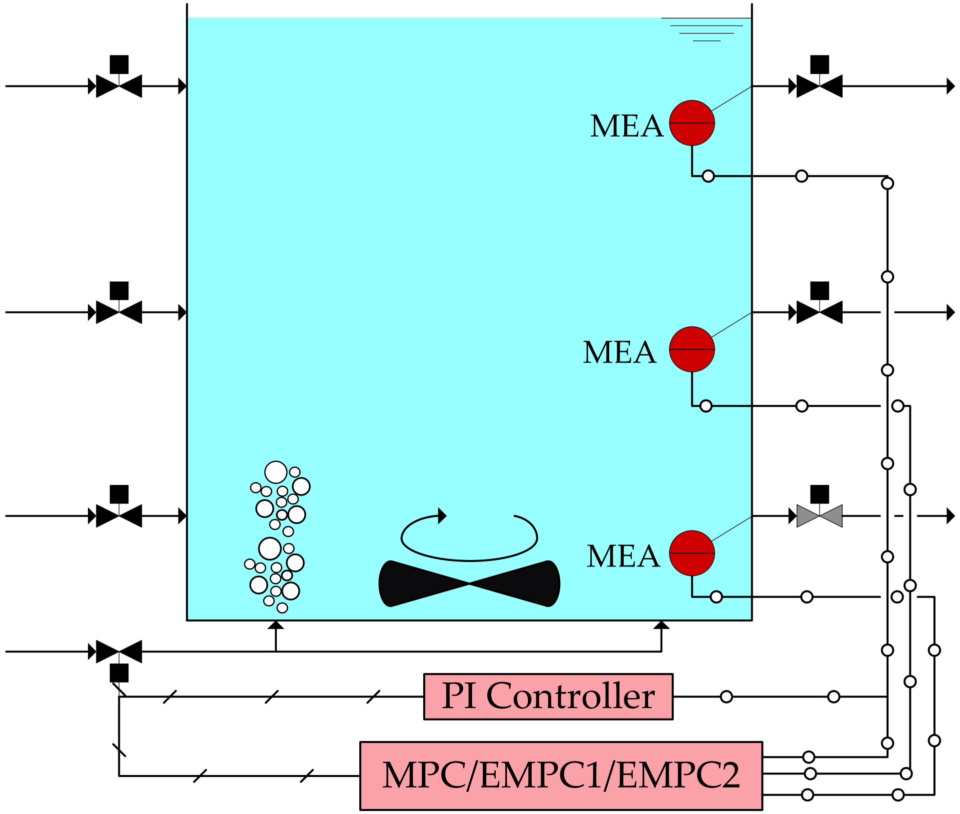
Hybrid Mechanistic ML Models

Benefits over pure data-driven models:

- Requires less data
- Have system insight
- Better controller performance
- Better performance with nonlinear dynamics
- Better performance in extrapolation
  - More useful for optimization applications

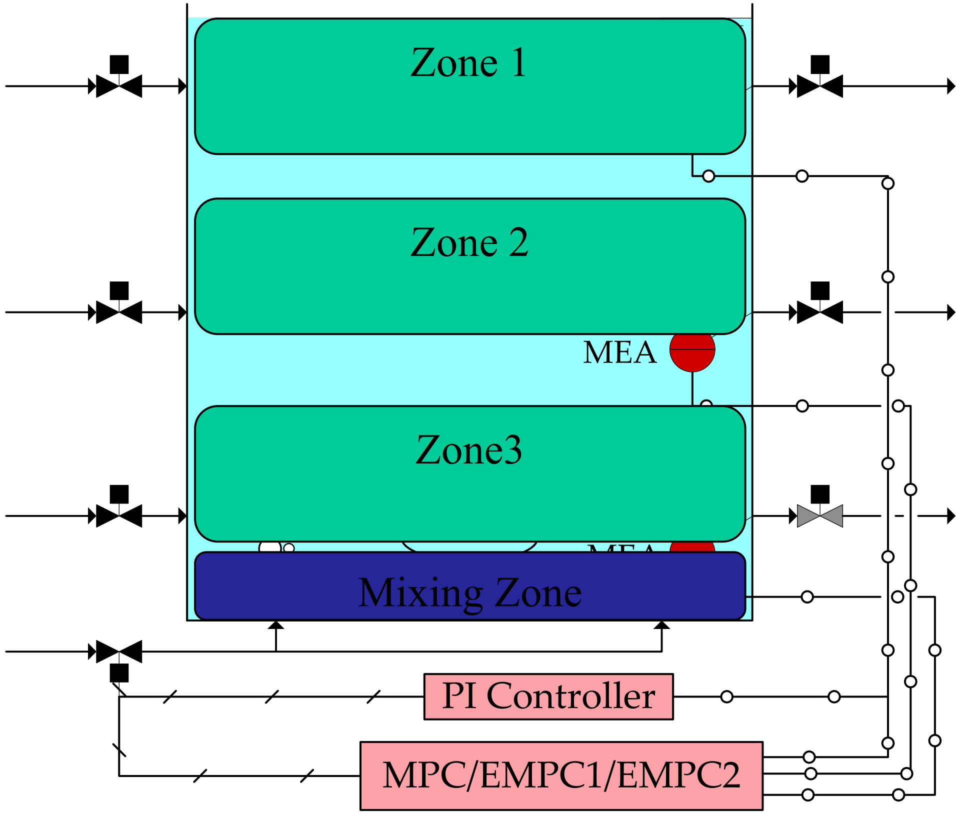
Robust Simulation of Hybrid Models

Physics-Informed Data-Driven Models



- Want to control wastewater treatment processes to optimize energy consumption and meet discharge requirements

Physics-Informed Data-Driven Models



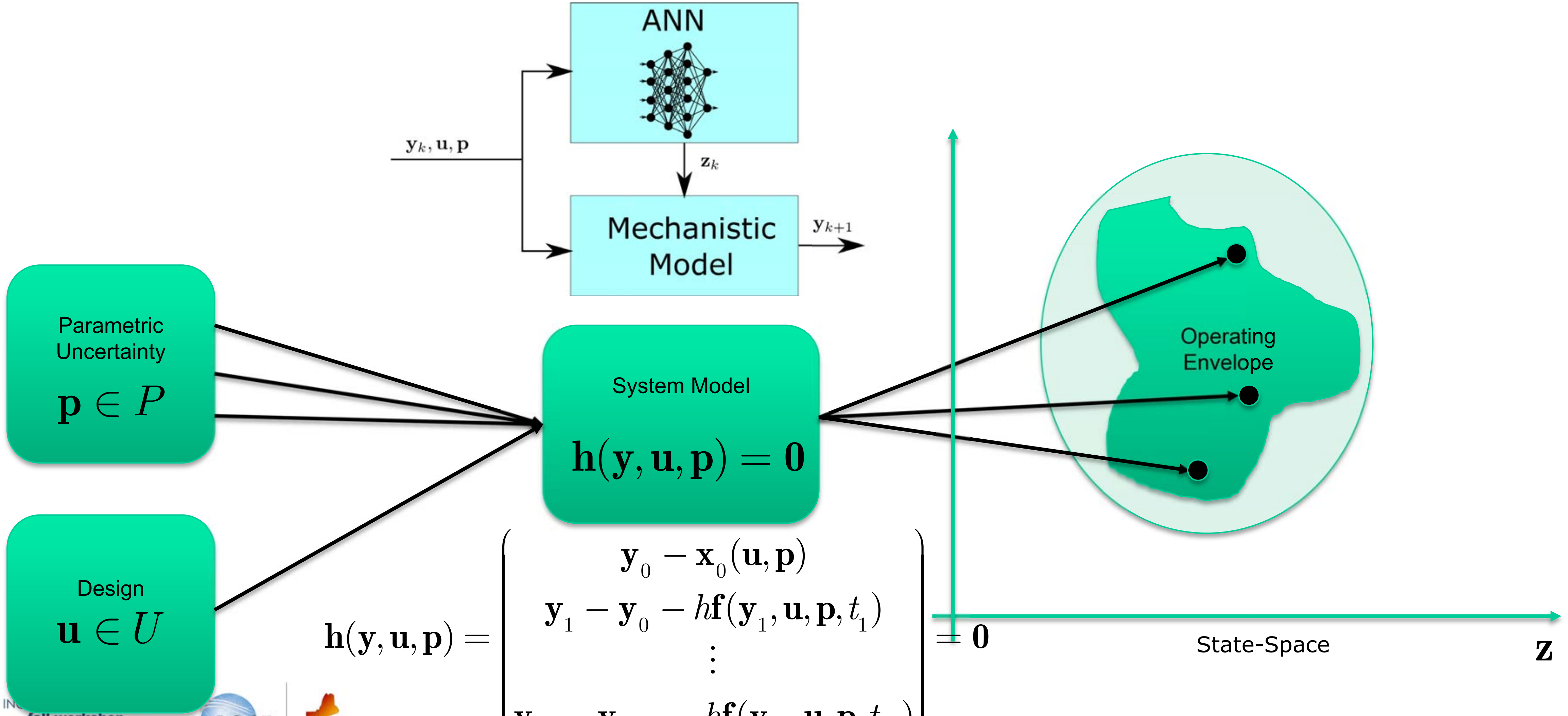
- Developed a compartment model with *unknown* parameters for mass transfer between compartments
- Applied deterministic global optimization for training to obtain guaranteed best-possible fit

Physics-Informed Data-Driven Models

Percent error relative to physics-informed data-driven model

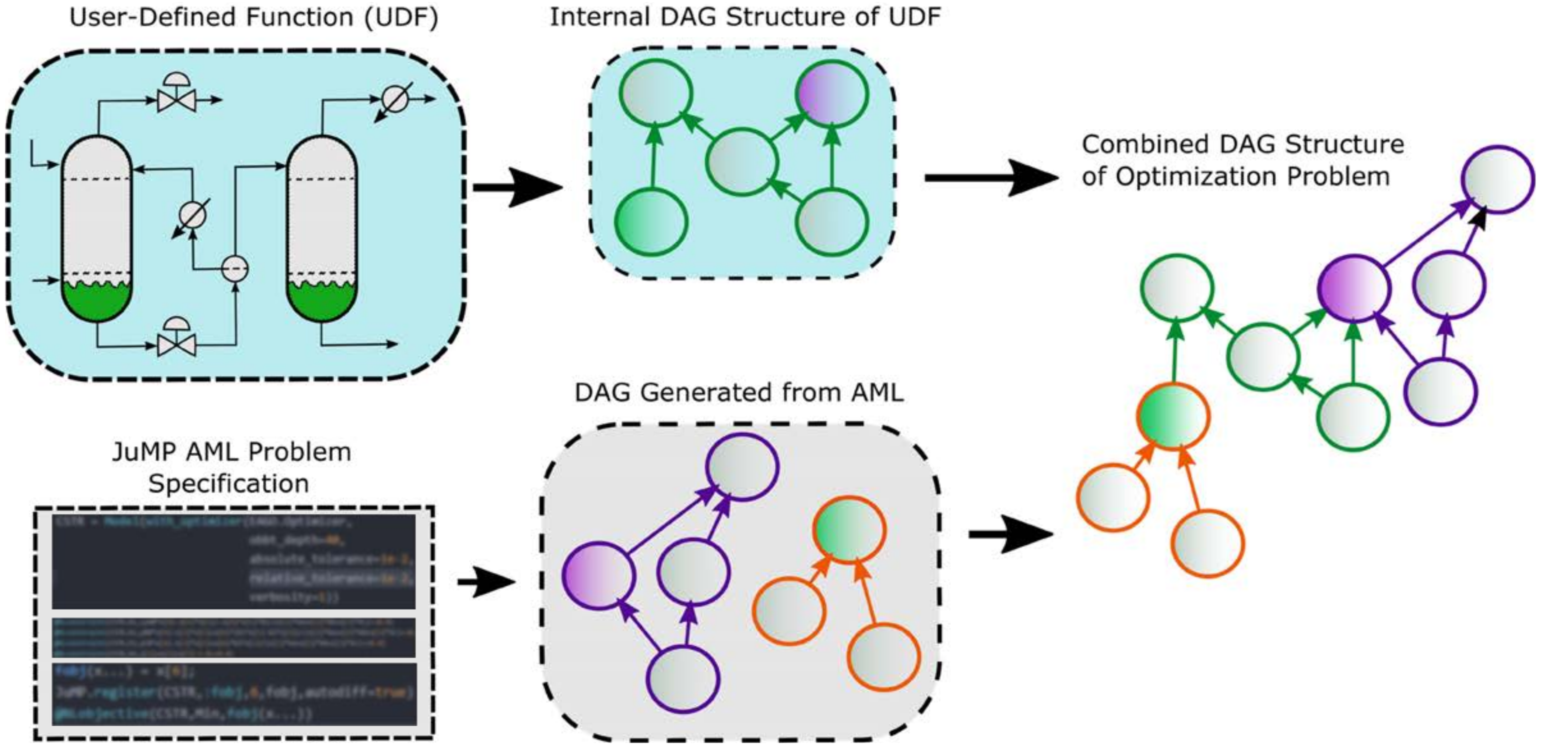
	CFD	Pure DD (ML) Model
Experiment 1	108%	329%
Experiment 2	1287%	588%
Experiment 3	511%	319%

Hybrid Mechanistic ML Models

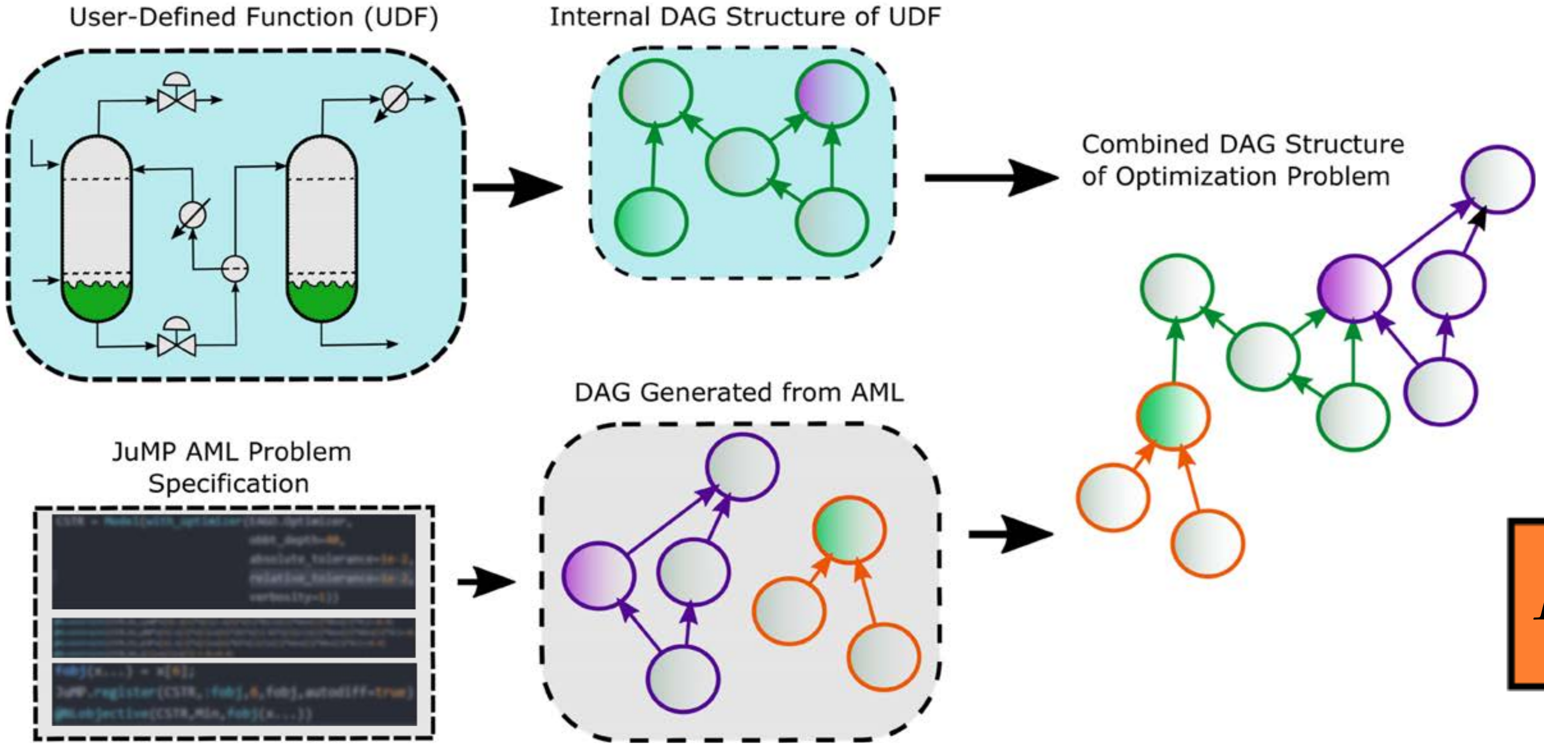


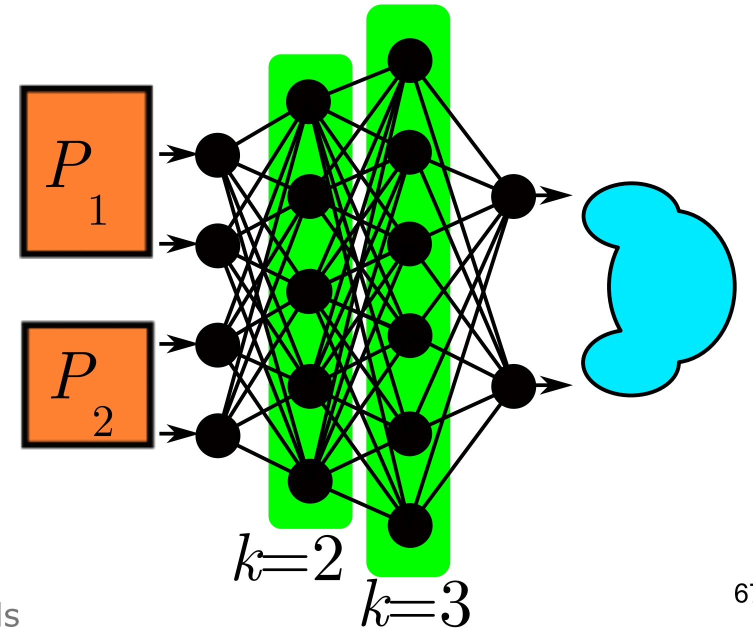
$$\mathbf{h}(\mathbf{y}, \mathbf{u}, \mathbf{p}) = \begin{pmatrix} \mathbf{y}_0 - \mathbf{x}_0(\mathbf{u}, \mathbf{p}) \\ \mathbf{y}_1 - \mathbf{y}_0 - h\mathbf{f}(\mathbf{y}_1, \mathbf{u}, \mathbf{p}, t_1) \\ \vdots \\ \mathbf{y}_K - \mathbf{y}_{K-1} - h\mathbf{f}(\mathbf{y}_K, \mathbf{u}, \mathbf{p}, t_K) \end{pmatrix} = \mathbf{0}$$

Robust Simulation of Hybrid Models



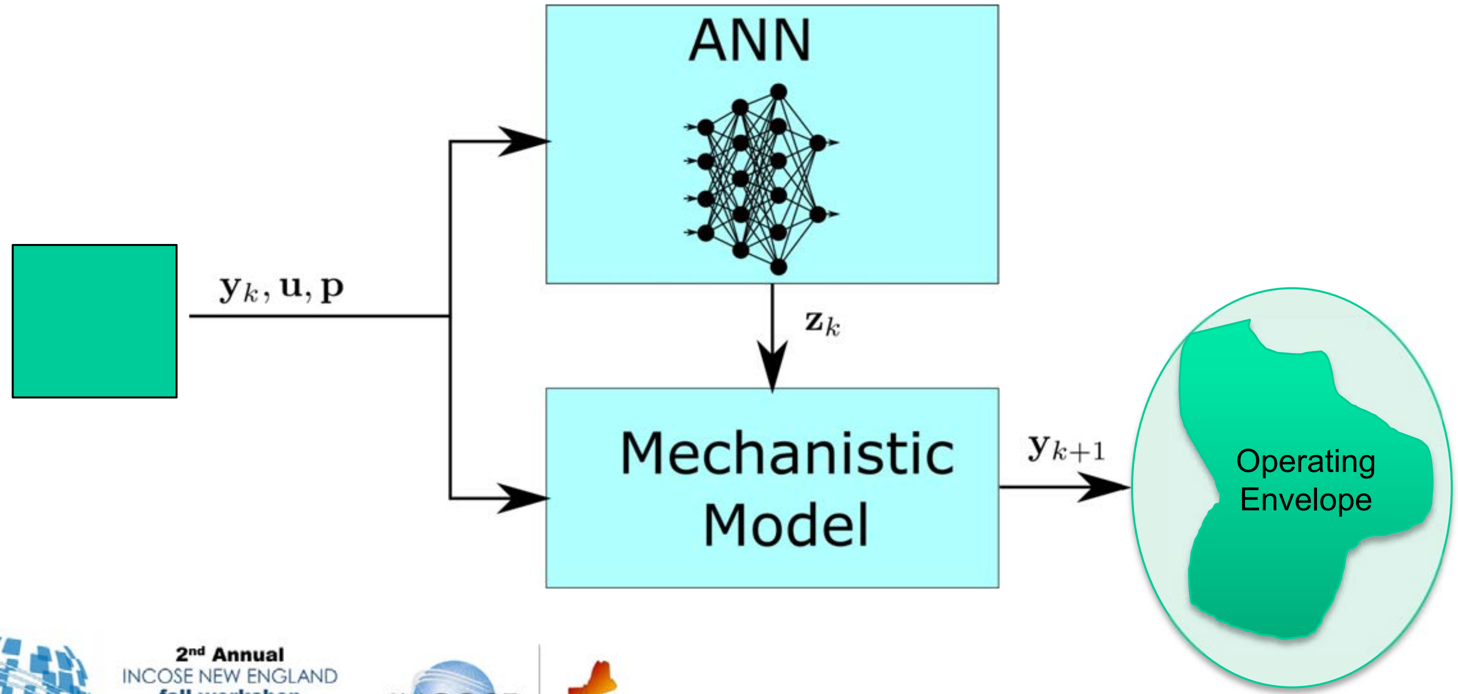
Robust Simulation of Hybrid Models

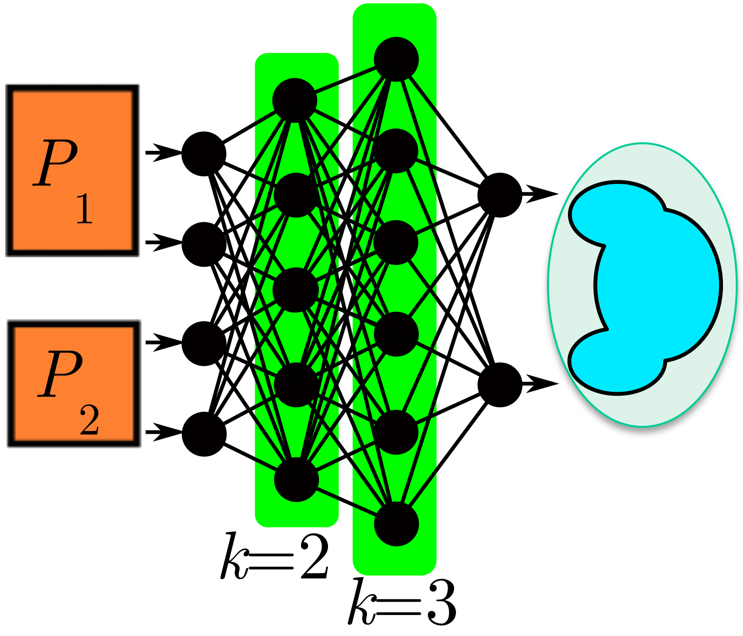




Robust Simulation of Hybrid Models

- Use our set-valued bounding theory to rigorously bound the states



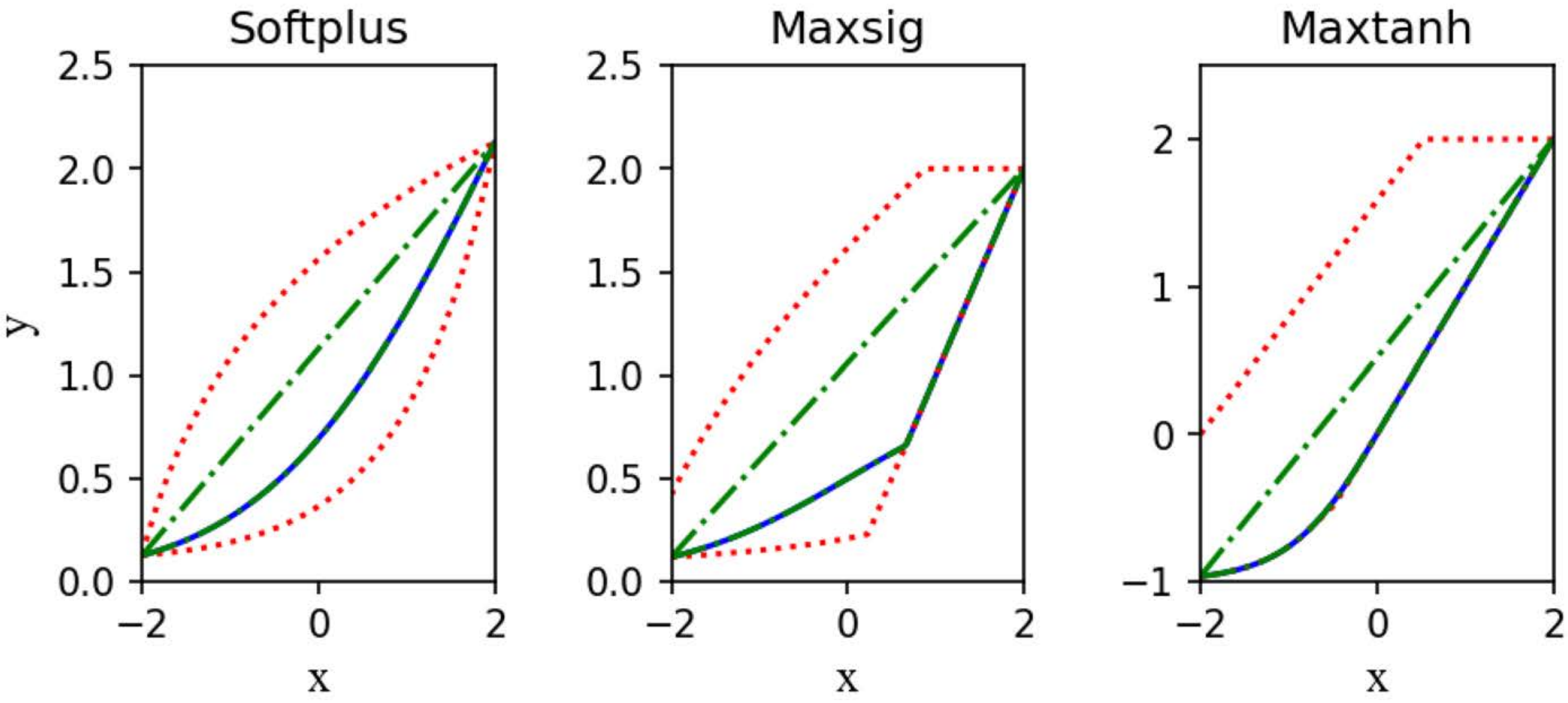


Relaxations of Activation Functions

$\log(1 + \exp(x))$

$\max\left(x, \frac{1}{1 + \exp(-x)}\right)$

$\max(x, \tanh(x))$

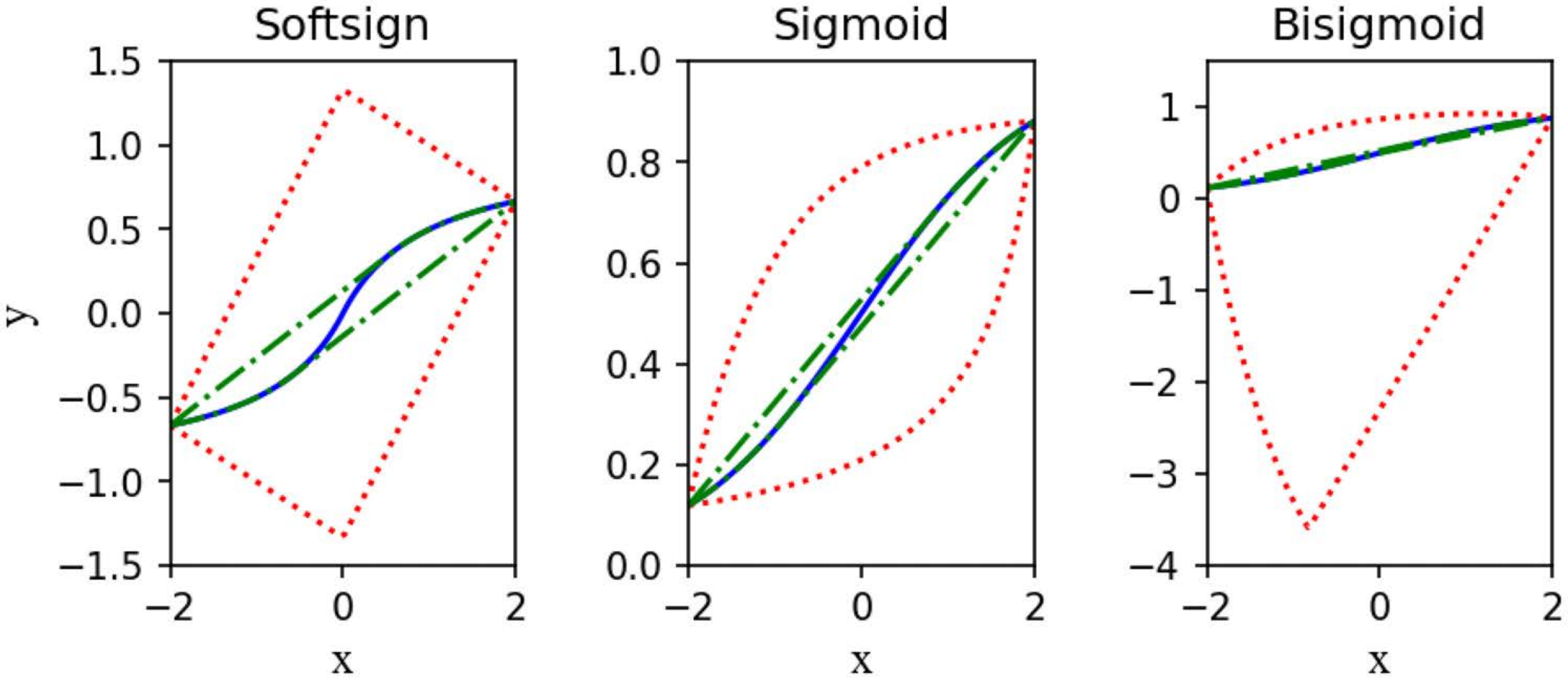


Relaxations of Activation Functions

$$\frac{x}{1+|x|}$$

$$\frac{1}{1+\exp(-x)}$$

$$1-\frac{2}{1+\exp(-x)}$$



Conclusion

- Developed rigorous bounding theory for steady-state and dynamical systems for mechanistic models
  - Formal uncertainty quantification
  - Extremely powerful open-source deterministic global optimizer for advanced user-defined models
- Want to exploit hybrid modeling approaches to overcome challenges with pure mechanistic and pure data-driven approaches

Conclusion

- Applied global optimization for training a physics-informed data-driven model to demonstrate the tradeoff
- Preliminary work on bounding a library of common basis functions for NN
  - Enable rigorously bounding hybrid models
  - Formal uncertainty quantification of hybrid models

THANK YOU

Matthew Wilhelm
PhD Candidate

Chenyu Wang
PhD Candidate

Any Questions?

Process Systems and Operations Research Laboratory

UCONN
SCHOOL OF ENGINEERING

This material is based upon work supported by the National Science Foundation under Grant No.: 1706343, 1932723. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.