Rapid and Accurate Reachability Analysis for Nonlinear Systems by Exploiting Model Redundancy

Joseph K. Scott
Chemical and Biomolecular Engineering
Clemson University
Scott Research Lab

Joseph K. Scott
Assistant Professor, ChBE
Clemson University

PhD, ChE, MIT 2012
MS, ChE, MIT 2018
BS, ChE, Wayne State, 2006

AFOSR YIP (2016)
David W. Smith, Jr. Award,
AIChE CAST (2016)
Automatica Paper Prize, IFAC (2016)

7 PhDs, 9 UGs

Funding:
AFOSR
Rapid Transforming Process Industries
NIH
SRNL
U.S. Department of Energy
Sandia National Laboratories
NSF
South Carolina Department of Commerce
Research Overview

Advanced Process Control
- Optimization theory and algorithms
- Global AC optimal power flow

Modeling/Applications
- Decomposition and high-performance computing
- Stochastic optimization
- Global/mixed-integer dynamic optimization

Uncertainty propagation
- Safety/performance verification

Fault detection and diagnosis
- Fault detection and diagnosis

Stochastic MPC
- Adsorption and membrane processes

Renewable energy systems
- Lignin recovery and valorization

Downstream biologics
- Perfusion cell culture
- Capture
- Polish
- Virus filtration
- UF/DF
Reachability Analysis

\[ \dot{x}(t, p) = f(t, p, x(t, p)), \quad x(t_0, p) = x_0(p) \]

\( p \) lies in an \( n_p \)-dimensional interval \( P \)

Model parameters (physical/empirical constants)

Reachable Set

\[ x(\hat{t}, P) \]
Reachability Analysis

\[
\dot{x}(t, p) = f(t, p, x(t, p)), \quad x(t_0, p) = x_0(p)
\]

\(p\) lies in an \(n_p\)-dimensional interval \(P\)

Model parameters (physical/empirical constants)

![Diagram showing state bounds and reachable set](image)
Reachability Analysis

\[ \dot{x}(t, p) = f(t, p, x(t, p)), \quad x(t_0, p) = x_0(p) \]

\( p \) lies in an \( n_p \)-dimensional interval \( P \)

**Objective:** Compute guaranteed enclosure of \( x(t,P) \) fast enough for use in online control algorithms and with as little overestimation as possible.
Motivation

Uncertainty Quantification

Robot Motion Planning

Safety Verification

Process Fault Detection

Autonomous Vehicles

Global Optimization

Objectives

Existing Methods and Challenges

Sampling methods
- Do not give rigorous bounds
- High accuracy is very expensive

Conservative Linearization\(^1\)
- Accurate but slow
- Problematic for strong nonlinearity, large uncertainties

Taylor Model Methods\(^2,3\)
- Potentially very accurate
- Scales exponentially in states and parameters

Differential Inequalities\(^4\) (DI)
- Fast, but often very weak
- Recent advances have drastically improved accuracy at low cost in some cases

\(\text{Fundamental tradeoff between speed and accuracy}\)
Often unworkable for problems with >5 states and parameters

---
1. Althoff and Krogh, IEEE TAC, 2014
4. Scott and Barton, Automatica, 2013
Existing Methods and Challenges

Sampling methods
• Do not give rigorous bounds
• High accuracy is very expensive

Conservative Linearization\(^1\)
• Accurate but slow
• Problematic for strong nonlinearity, large uncertainties

Taylor Model Methods\(^2,3\)
• Potentially very accurate
• Scales exponentially in states and parameters

**Differential Inequalities\(^4\)** (DI)
• Fast, but often very weak
• Recent advances have drastically improved accuracy at low cost in some cases

**Objective:**
Simple, effective means to dramatically reduce conservatism of DI without compromising efficiency

---

1. Althoff and Krogh, IEEE TAC, 2014
4. Scott and Barton, Automatica, 2013
Differential Inequalities (DI)

\[ \dot{x}_i(t, p) = f_i(t, p, x(t, p)) \]

\[ \dot{x}_i^L(t) < \min_{p \in P, z \in [x^L(t), x^U(t)], z_i = x_i^L(t)} f_i(t, p, z) \]

\[ \dot{x}_i^U(t) > \max_{p \in P, z \in [x^L(t), x^U(t)], z_i = x_i^U(t)} f_i(t, p, z) \]

\[ x_i(t_0, p) = x_{i,0}(p) \]

\[ x_i^L(t_0) < \min_{p \in P} x_{i,0}(p) \]

\[ x_i^U(t_0) > \max_{p \in P} x_{i,0}(p) \]
Differential Inequalities (DI)

Right-hand Sides:
Interval extensions of $f$

$$\dot{x}_i(t, p) = f_i(t, p, x(t, p)), \quad x_i(t_0, p) = x_{0,i}(p)$$

$$\dot{x}_i^L(t) = f_i^L(t, P, X(t)), \quad x_i^L(t_0) = x_{0,i}^L(P)$$

$$\dot{x}_i^U(t) = f_i^U(t, P, X(t)), \quad x_i^U(t_0) = x_{0,i}^U(P)$$

Initial conditions:
Interval extensions of $x_0$

Compute bounds as the solutions of auxiliary ODEs

\[ \text{Diagram:} \quad \begin{align*} x_2^U \\ x_2^L \end{align*} \quad \text{and} \quad \begin{align*} x_1^L & \quad \text{at} \quad x(t, p) \quad \text{and} \quad X(t) \end{align*} \]
Differential Inequalities (DI)

Right-hand Sides: Interval extensions of $f$

$$
\begin{align*}
\dot{x}_i(t, p) &= f_i(t, p, x(t, p)), \\
\dot{x}_i^L(t) &= f_i^L(t, P, \beta^L_i(X(t))), \\
\dot{x}_i^U(t) &= f_i^U(t, P, \beta^U_i(X(t))),
\end{align*}
$$

Initial conditions: Interval extensions of $x_0$

$$
\begin{align*}
x_i(t_0, p) &= x_{0,i}(p), \\
x_i^L(t_0) &= x_{0,i}^L(P), \\
x_i^U(t_0) &= x_{0,i}^U(P),
\end{align*}
$$

Compute bounds as the solutions of auxiliary ODEs

\[ x(t, p) \]

\[ \beta^U_2(X(t)) \]
**Differential Inequalities (DI)**

### Key Ideas
- Compute bounds as the solutions of an auxiliary system of ODEs
- Exploit factorable representation of governing equations
- Solve numerically with any state-of-the-art simulator
- **Very efficient but potentially very weak bounds**

#### Right-hand Sides:
Interval extensions of \( f \)

\[
\dot{x}_i(t, p) = f_i(t, p, x(t, p)), \quad x_i(t_0, p) = x_{0,i}(p)
\]

\[
\dot{x}_i^L(t) = f_i^L(t, P, \beta_i^L(X(t))), \quad x_i^L(t_0) = x_{0,i}^L(P)
\]

\[
\dot{x}_i^U(t) = f_i^U(t, P, \beta_i^U(X(t))), \quad x_i^U(t_0) = x_{0,i}^U(P)
\]

#### Initial conditions:
Interval extensions of \( x_0 \)

Compute bounds as the solutions of auxiliary ODEs
An Illustrative Example

\[ A + B \xrightarrow{k_f, k_r} C \]

**Bounds on Concentration of C**

- \( \dot{x}_A = -k_f x_A x_B + k_r x_C \)
- \( \dot{x}_B = -k_f x_A x_B + k_r x_C \)
- \( \dot{x}_C = k_f x_A x_B - k_r x_C \)

Initial conditions:
- \( x_{A,0} = 0.5 \)
- \( x_{B,0} = 1 \)
- \( x_{C,0} = 0 \)

Parameter bounds:
- \( p = (k_f, k_r) \)
- \( P = [100, 500] \times [0.001, 0.01] \)
An Illustrative Example

Implicit relations in model not preserved by bounding procedure
Affine Reaction Invariants

Left null vectors of the stoichiometry matrix
Automatically computed by matrix factorization

Can use these redundant equations to refine bounds in each time step of the bounding procedure

Scott and Barton, Computers and Chemical Engineering, 34, 2010
Scott and Barton, Automatica, 49, 2013
An Illustrative Example

Implicit relations in model not preserved by bounding procedure
Affine Reaction Invariants

Left null vectors of the stoichiometry matrix
Automatically computed by matrix factorization

Can use these redundant equations to refine bounds in each time step of the bounding procedure

Scott and Barton, Computers and Chemical Engineering, 34, 2010
Scott and Barton, Automatica, 49, 2013
\[
\dot{x}(t, p) = f(t, p, x(t, p))
\]

\[(p, x(t, p)) \in G\]
Differential Inequalities with Constraints

\[ \dot{x}(t, p) = f(t, p, x(t, p)) \]

\[ (p, x(t, p)) \in G \]

\[ \dot{x}_i^L(t) = f_i^L(t, \mathcal{I}_G\{P, \beta_i^L(X(t))\}) \]

\[ \dot{x}_i^U(t) = f_i^U(t, \mathcal{I}_G\{P, \beta_i^U(X(t))\}) \]

**Interval Refinement Operator**

1. \[ \mathcal{I}_G\{P, Z\} \supset (P \times Z) \cap G \]
2. \[ d_H(\mathcal{I}_G\{P, Z_1\}, \mathcal{I}_G\{P, Z_2\}) \leq Ld_H(Z_1, Z_2) \]
Differential Inequalities with Constraints

\[ \dot{x}(t, p) = f(t, p, x(t, p)) \]
\[ (p, x(t, p)) \in G \]

\[ \dot{x}_i^L(t) = f_i^L(t, \mathcal{I}_G \{ P, \beta_i^L(X(t)) \}) \]
\[ \dot{x}_i^U(t) = f_i^U(t, \mathcal{I}_G \{ P, \beta_i^U(X(t)) \}) \]

Interval Refinement Operator

1. \( \mathcal{I}_G \{ P, Z \} \supset (P \times Z) \cap G \)
2. \( d_H(\mathcal{I}_G \{ P, Z_1 \}, \mathcal{I}_G \{ P, Z_2 \}) \leq Ld_H(Z_1, Z_2) \)
Example: Nonlinear Enzyme Kinetics

\[ \begin{align*}
\dot{x}_A &= -k_1 x_A x_F + k_2 x_{F:A} + k_6 x_{R:A'} \\
\dot{x}_{F:A} &= k_1 x_A x_F - k_2 x_{F:A} - k_3 x_{F:A} \\
\dot{x}_R &= -k_4 x_{A'} x_R + k_5 x_{R:A'} + k_6 x_{R:A'} \\
\dot{x}_{F} &= -k_1 x_A x_F + k_2 x_{F:A} + k_3 x_{F:A} \\
\dot{x}_{A'} &= k_3 x_{F:A} - k_4 x_{A'} x_R + k_5 x_{R:A'} \\
\dot{x}_{R:A'} &= k_4 x_{A'} x_R - k_5 x_{R:A'} - k_6 x_{R:A'}
\end{align*} \]

Three affine reaction invariants (metabolic pools)

Scott and Barton, Automatica, 49, 2013
Example: Nonlinear Enzyme Kinetics

- ~250x faster than state-of-the-art zonotope method

Scott and Barton, Automatica, 49, 2013
Example: Lotka – Volterra Predator Prey Model

\[ \dot{x}_1 = p_1 x_1 (1 - x_2), \quad x_{0,1} = 1.2 \]
\[ \dot{x}_2 = p_2 x_2 (x_2 - 1), \quad x_{0,2} = 1.1 \]

\[ p_1 \in [2.999, 3.000] \]
\[ p_2 \in [0.999, 1.000] \]

Redundant constraint
\[
\begin{align*}
p_2 \left[ \ln \left( \frac{x_1}{x_{0,1}} - (x_1 - x_{0,1}) \right) \right] + p_1 \left[ \ln \left( \frac{x_2}{x_{0,2}} - (x_2 - x_{0,2}) \right) \right] &= 0
\end{align*}
\]

- ~ 7x of single simulations
- ~ 368x faster than Taylor Model (TM) with interval remainder*

Example: Lotka – Volterra Predator-Prey Model

\[
\begin{align*}
\dot{x}_1 &= p_1 x_1 (1 - x_2), \quad x_{0,1} = 1.2 \\
\dot{x}_2 &= p_2 x_2 (x_2 - 1), \quad x_{0,2} = 1.1
\end{align*}
\]

\[p_1 \in [2.999, 3.000] \quad \quad \quad \quad \quad \quad \quad \quad \quad p_2 \in [0.999, 1.000]\]

Redundant constraint

\[p_2 \left[ \ln \left( \frac{x_1}{x_{0,1}} - (x_1 - x_{0,1}) \right) \right] + p_1 \left[ \ln \left( \frac{x_2}{x_{0,2}} - (x_2 - x_{0,2}) \right) \right] = 0\]

DI + constraints is typically \textit{much} more accurate than standard DI

\textit{How can this be extended to cases with no (known) constraints?}

*Lin and Stadtherr, Applied Numerical Mathematics, 57, 2007*
DI with Manufactured Constraints

\[ \dot{x}_A = -k_f x_A x_B + k_r x_C + \tau^{-1}(x_{A,in} - x_A) \]

\[ \dot{x}_B = -k_f x_A x_B + k_r x_C + \tau^{-1}(x_{B,in} - x_B) \]

\[ \dot{x}_C = k_f x_A x_B - k_r x_C - \tau^{-1}x_C \]
DI with Manufactured Constraints

\[ \dot{x}_A = -k_f x_A x_B + k_r x_C + \tau^{-1}(x_{A,\text{in}} - x_A) \]
\[ \dot{x}_B = -k_f x_A x_B + k_r x_C + \tau^{-1}(x_{B,\text{in}} - x_B) \]
\[ \dot{x}_C = k_f x_A x_B - k_r x_C - \tau^{-1}x_C \]

Possible to *manufacture* them by introducing redundant states and ODEs

\[ x_D \equiv x_A + x_B + 2x_C \]
\[ x_E \equiv x_A - x_B \]

\[ \dot{x}_D = \tau^{-1}(x_{A,\text{in}} + x_{B,\text{in}} - x_D) \]
\[ \dot{x}_E = \tau^{-1}(x_{A,\text{in}} - x_{B,\text{in}} - x_E) \]  

Indep. of uncertainty!
DI with Manufactured Constraints

\[ \dot{x}_A = -k_f x_A x_B + k_r x_C + \tau^{-1}(x_{A,in} - x_A) \]
\[ \dot{x}_B = -k_f x_A x_B + k_r x_C + \tau^{-1}(x_{B,in} - x_B) \]
\[ \dot{x}_C = k_f x_A x_B - k_r x_C - \tau^{-1}x_C \]
\[ \dot{x}_D = \tau^{-1}(x_{A,in} + x_{B,in} - x_D) \]
\[ \dot{x}_E = \tau^{-1}(x_{A,in} - x_{B,in} - x_E) \]

\[ x_D = x_A + x_B + 2x_C \]
\[ x_E = x_A - x_B \]

Apply fast DI bounding to this ‘lifted’ system
Refine using ‘manufactured’ invariants
DI with Manufactured Constraints

\[ \dot{x}_A = -k_f x_A x_B + k_r x_C + \tau^{-1}(x_{A,in} - x_A) \]
\[ \dot{x}_B = -k_f x_A x_B + k_r x_C + \tau^{-1}(x_{B,in} - x_B) \]
\[ \dot{x}_C = k_f x_A x_B - k_r x_C - \tau^{-1}x_C \]
\[ \dot{x}_D = \tau^{-1}(x_{A,in} + x_{B,in} - x_D) \]
\[ \dot{x}_E = \tau^{-1}(x_{A,in} - x_{B,in} - x_E) \]

Indep. of uncertainty!

Solution invariants

Apply fast DI bounding to this ‘lifted’ system
Refine using ‘manufactured’ invariants

\[ x_D = x_A + x_B + 2x_C \]
\[ x_E = x_A - x_B \]
General Method

\[ \dot{x}(t, p) = f(t, p, x(t, p)) \]
General Method

\[ \dot{x}(t, p) = f(t, p, x(t, p)) \]

Define redundant states

\[ y(t) \equiv g(x(t)) \]
General Method

\[
\dot{x}(t, p) = f(t, p, x(t, p))
\]

Define redundant states

\[
y(t) \equiv g(x(t))
\]

Differentiate and form ‘lifted’ system

\[
\frac{d}{dt} \begin{bmatrix} x(t) \\ y(t) \end{bmatrix} = \begin{bmatrix} f(t, p, x(t, p)) \\ \frac{\partial g}{\partial x}(x(t)) f(t, p, x(t, p)) \end{bmatrix}
\]
General Method

\[ \dot{x}(t, p) = f(t, p, x(t, p)) \]

Define redundant states

\[ y(t) \equiv g(x(t)) \]

Differentiate and form 'lifted' system

\[
\frac{d}{dt}\begin{bmatrix}
  x(t) \\
  y(t)
\end{bmatrix} = \begin{bmatrix}
  f(t, p, x(t, p)) \\
  \frac{\partial g}{\partial x}(x(t)) f(t, p, x(t, p))
\end{bmatrix}
\]
General Method

\[ \dot{x}(t, p) = f(t, p, x(t, p)) \]

Define redundant states

\[ y(t) \equiv g(x(t)) \]

Differentiate and form 'lifted' system

\[
\begin{bmatrix}
\frac{d}{dt} x(t) \\
\frac{d}{dt} y(t)
\end{bmatrix} =
\begin{bmatrix}
f(t, p, x(t, p)) \\
\frac{\partial g}{\partial x}(x(t)) f(t, p, x(t, p))
\end{bmatrix}
\]

\[ y(t) - g(x(t)) = 0 \]

Lifted system satisfies solution invariants by design


**General Method**

- Define redundant states: \( y(t) \equiv g(x(t)) \)
- Differentiate and form 'lifted' system:
  \[
  \begin{bmatrix}
  \dot{x}(t, p) \\
  y(t)
  \end{bmatrix} =
  \begin{bmatrix}
  f(t, p, x(t, p)) \\
  \frac{\partial g}{\partial x}(x(t)) f(t, p, x(t, p))
  \end{bmatrix}
  \]
- Lifted system satisfies solution invariants by design: \( y(t) - g(x(t)) = 0 \)

\[
\begin{align*}
\dot{x}(t, p) &= f(t, p, x(t, p)) \\
\frac{d}{dt} \begin{bmatrix} x(t) \\ y(t) \end{bmatrix} &= \begin{bmatrix} f(t, p, x(t, p)) \\ \frac{\partial g}{\partial x}(x(t)) f(t, p, x(t, p)) \end{bmatrix} \\
y(t) - g(x(t)) &= 0
\end{align*}
\]
General Method

Define redundant states

\[ \dot{x}(t,p) = f(t,p,x(t,p)) \]

\[ y(t) \equiv g(x(t)) \]

Differentiate and form "lifted" system

\[
\begin{bmatrix}
\frac{d}{dt} x(t) \\
\frac{d}{dt} y(t)
\end{bmatrix} =
\begin{bmatrix}
f(t,p,x(t,p)) \\
\frac{\partial g}{\partial x}(x(t)) f(t,p,x(t,p))
\end{bmatrix}
\]

\[ y(t) - g(x(t)) = 0 \]

Lifted system satisfies solution invariants by design
Example: Multistage Liquid-liquid Extraction

\[ V_L \dot{x}_n = L(x_{n-1} - x_n) - Q_n \]
\[ V_G \dot{y}_n = G(y_{n+1} - y_n) + Q_n \]
\[ Q_n = K_L a (x_n - x^*_n) V \]
\[ x^*_n = p_1 y_n^4 + p_2 y_n^3 + p_3 y_n^2 \]
\[ + p_4 y_n + p_5 \]
\[ n = 1, \ldots, 5 \]
Example: Multistage Liquid-liquid Extraction

\[ V_L \dot{x}_n = L(x_{n-1} - x_n) - Q_n \]
\[ V_G \dot{y}_n = G(y_{n+1} - y_n) + Q_n \]
\[ Q_n = K_L a(x_n - x^*_n)V \]
\[ x^*_n = p_1 y_n^4 + p_2 y_n^3 + p_3 y_n^2 + p_4 y_n + p_5 \]
\[ n = 1, \ldots, 5 \]

\[ N_n = V_L x_n + V_G y_n \]

\[ \dot{N}_n = L(x_{n-1} - x_n) - Q_n + G(y_{n+1} - y_n) + Q_n \]
Example: Multistage Liquid-liquid Extraction

\[ V_L \dot{x}_n = L(x_{n-1} - x_n) - Q_n \]
\[ V_G \dot{y}_n = G(y_{n+1} - y_n) + Q_n \]
\[ Q_n = K_L a(x_n - x_n^*) V \]
\[ x_n^* = p_1 y_n^4 + p_2 y_n^3 + p_3 y_n^2 + p_4 y_n + p_5 \]
\[ n = 1, \ldots, 5 \]

\[ N_n = V_L x_n + V_G y_n \]

\[ \dot{N}_n = L(x_{n-1} - x_n) - Q_n + G(y_{n+1} - y_n) + Q_n \]
Why Does This Work?

<table>
<thead>
<tr>
<th>Equivalent</th>
<th></th>
<th>Equivalent</th>
</tr>
</thead>
<tbody>
<tr>
<td>in real arithmetic</td>
<td>$f(x) = x - x^2$</td>
<td>$F(X) = X - X^2 = [0,1] - [0,1] = [-1,1]$</td>
</tr>
<tr>
<td>in interval arithmetic</td>
<td>$g(x) = x(1 - x)$</td>
<td>$G(X) = X(1 - X) = <a href="%5B0,1%5D">0,1</a> = [0,1]$</td>
</tr>
</tbody>
</table>
Why Does This Work?

Equivalent in real arithmetic $\iff$ Equivalent in interval arithmetic

\[
\begin{align*}
  f(x) &= x - x^2 \\
  F(X) &= X - X^2 = [0, 1] - [0, 1] = [-1, 1]
\end{align*}
\]

\[
\begin{align*}
  g(x) &= x(1 - x) \\
  G(X) &= X(1 - X) = [0, 1]([0, 1]) = [0, 1]
\end{align*}
\]

This is the “dependency problem”

Multiple redundant expressions $\Rightarrow$ Sharper enclosures
Why Does This Work?

Equivalent
in real arithmetic

⇒

Equivalent
in interval arithmetic

Extension to dynamic systems

Augmented dynamics:

\[
\begin{bmatrix}
\dot{x}(t) \\
\dot{y}(t)
\end{bmatrix} =
\begin{bmatrix}
f(t,p,x(t,p)) \\
\frac{\partial g}{\partial x}(x(t))f(t,p,x(t,p))
\end{bmatrix}
\]

Manufactured Invariant:

\[
0 = y(t) - g(x(t))
\]

Method provides multiple redundant ‘expressions’ for \(x(t)\)
What are Good Invariants to Add?

\[
\begin{bmatrix}
\dot{x}(t) \\
\dot{y}(t)
\end{bmatrix} =
\begin{bmatrix}
f(t, p, x(t, p)) \\
\frac{\partial g}{\partial x}(x(t))f(t, p, x(t, p))
\end{bmatrix}
\]

Make this ‘simple’

\[\rightarrow\text{Easy to bound (Minimize dependency problem)}\]

Linear Case:

\[y(t) \equiv \mu^T x(t)\]

\[
\begin{bmatrix}
\dot{x}(t) \\
\dot{y}(t)
\end{bmatrix} =
\begin{bmatrix}
f(t, p, x(t, p)) \\
\mu^T f(t, p, x(t, p))
\end{bmatrix}
\]

Make this ‘simple’

\[\rightarrow\text{Term cancellations (zero is best)}\]
\[\rightarrow\text{Algebraic simplifications}\]
Example: Anaerobic Digester with pH Self-Regulation and Liquid-Gas Transfer

\begin{align*}
\dot{X}_1 &= (\mu_1(S_1) - \alpha D)X_1 \\
\dot{X}_2 &= (\mu_2(S_2) - \alpha D)X_2 \\
\dot{Z} &= D(Z^{in} - Z) \\
\dot{S}_1 &= D(S_1^{in} - S_1) - k_1\mu_1(S_1)X_1 \\
\dot{S}_2 &= D(S_2^{in} - S_2) + k_2\mu_1(S_1)X_1 - k_3\mu_2(S_2)X_2 \\
\dot{C} &= D(C^{in} - C) - q_{CO_2} + k_4\mu_1(S_1)X_1 + k_5\mu_2(S_2)X_2 \\
q_{CO_2} &= k_La(C + S_2 - Z - K_HP_{CO_2}) \\
P_{CO_2} &= \left(\phi_{CO_2} - \sqrt{\phi_{CO_2}^2 - 4K_HP_t(C + S_2 - Z)}\right)(2K_H)^{-1} \\
\phi_{CO_2} &= C + S_2 - Z + K_HP_t + \frac{k_6}{k_La}\mu_2(S_2)X_2 \\
\mu_1(S_1) &= \bar{\mu}_1 \frac{S_1}{S_1 + K_{S_1}} \\
\mu_2(S_2) &= \bar{\mu}_2 \frac{S_2}{S_2 + K_{S_2} + S_2^2 / K_{I_2}}
\end{align*}

- Complex dynamics due to pH self-regulating and liquid gas transfer
- Fast dynamics on a time-scale of hours
- Slow dynamics on a time-scale of days

Uncertainty

\begin{align*}
X_{1,0} &\in [0.49, 0.51] & k_1 &\in [42.14, 42.98] \\
X_{2,0} &\in [0.98, 1.02] & k_2 &\in [116.5, 118.24] \\
C_0 &\in [39.2, 40.8]
\end{align*}
Example: Anaerobic Digester with pH Self-Regulation and Liquid-Gas Transfer

\[
\begin{align*}
\dot{X}_1 &= (\mu_1(S_1) - \alpha D)X_1 \\
\dot{X}_2 &= (\mu_2(S_2) - \alpha D)X_2 \\
\dot{Z} &= D(Z^{\text{in}} - Z) \\
\dot{S}_1 &= D(S_1^{\text{in}} - S_1) - k_1 \mu_1(S_1)X_1 \\
\dot{S}_2 &= D(S_2^{\text{in}} - S_2) + k_2 \mu_1(S_1)X_1 - k_3 \mu_2(S_2)X_2 \\
\dot{C} &= D(C^{\text{in}} - C) - q_{\text{CO}_2} + k_4 \mu_1(S_1)X_1 + k_5 \mu_2(S_2)X_2
\end{align*}
\]

Redundant states and ODEs:

\[
\begin{align*}
N_1 &\equiv k_1 X_1 + S_1 \\
N_2 &\equiv -k_2 X_1 + k_3 X_2 + S_2 \\
\dot{N}_1 &= k_1 (\mu(S_1) - \alpha D)X_1 + D(S_1^{\text{in}} - S_1) - k_1 \mu_1(S_1)X_1 \\
\dot{N}_2 &= D(S_2^{\text{in}} + S_1(\alpha - 1) - \alpha N_1) \\
\dot{N}_2 &= D(S_2^{\text{in}} + S_2(\alpha - 1) - \alpha N_2)
\end{align*}
\]

Redundant constraints

\[
\begin{align*}
0 &= -N_1 + k_1 X_1 + S_1 \\
0 &= -N_2 - k_2 X_1 + k_3 X_2 + S_2
\end{align*}
\]

- ~24x the cost of a single simulation
- ~846x faster than 4th order TM w/ ellipsoidal remainder bounds

Shen and Scott, ACC, 2018
Example: Lotka – Volterra Predator-Prey Model

\[
\begin{align*}
\dot{x}_1 &= p_1 x_1 (1 - x_2), \quad x_{0,1} = 1.2 \\
\dot{x}_2 &= p_2 x_2 (x_2 - 1), \quad x_{0,2} = 1.1 \\
p_1 &\in [2.999, 3.000] \\
p_2 &\in [0.999, 1.000]
\end{align*}
\]

Manufactured Constraints (in addition to existing nonlinear invariant)

\[
\begin{align*}
y_1 &= \frac{x_1}{p_1} - \frac{x_2}{p_2}, \\
y_2 &= x_1 x_2, \\
y_3 &= \frac{x_1}{x_2}, \\
y_i &= (\sin \frac{i\pi}{16}) x_1 + (\cos \frac{i\pi}{16}) x_2, \quad i = 1 : 7
\end{align*}
\]
Example: Trajectory Tracking

Simplified Dubins Car

\[
\begin{align*}
\dot{x} &= v \cos(\theta) \\
\dot{y} &= v \sin(\theta) \\
\dot{\theta} &= \omega
\end{align*}
\]

\(v \in [5, 6] \text{ (m/s)}\)
Example: Trajectory Tracking

Simplified Dubins Car

$$\begin{align*}
\dot{x} &= v \cos(\theta) \\
\dot{y} &= v \sin(\theta) \\
\dot{\theta} &= \omega
\end{align*}$$

$$v \in [5, 6] \text{ (m/s)}$$

Change of Coordinates (Error Dynamics)

$$\begin{align*}
\dot{s} &= \frac{v \cos(\theta_e)}{1 - c(s)} \\
\dot{e} &= v \sin(\theta_e) \\
\dot{\theta}_e &= \omega - \frac{c(s)v \cos(\theta_e)}{1 - c(s)e}
\end{align*}$$

Reference trajectory
Example: Trajectory Tracking

Simplified Dubins Car

\[
\begin{align*}
\dot{x} &= v \cos(\theta) \\
\dot{y} &= v \sin(\theta) \\
\dot{\theta} &= \omega
\end{align*}
\]

\[v \in [5, 6] \text{ (m/s)}\]

Change of Coordinates (Error Dynamics)

\[
\begin{align*}
\dot{s} &= \frac{v \cos(\theta_e)}{1 - c(s)} \\
\dot{e} &= v \sin(\theta_e) \\
\dot{\theta}_e &= \omega - \frac{c(s)v \cos(\theta_e)}{1 - c(s)e}
\end{align*}
\]

Controller

\[
\omega = \frac{vc(s)\cos(\theta_e)}{1 - c(s)e} - g_1(e, \theta_e, t)\theta_e - g_2v \frac{\sin(\theta_e)}{\theta_e} e
\]

Reference trajectory
Example: Trajectory Tracking

Simplified Dubins Car

\[
\begin{align*}
\dot{x} &= v \cos(\theta) \\
\dot{y} &= v \sin(\theta) \\
\dot{\theta} &= \omega
\end{align*}
\]

\[v \in [5, 6] \text{ (m/s)}\]

Change of Coordinates (Error Dynamics)

\[
\begin{align*}
\dot{s} &= \frac{v \cos(\theta_e)}{1 - c(s)} \\
\dot{e} &= v \sin(\theta_e) \\
\dot{\theta}_e &= \omega - \frac{c(s)v \cos(\theta_e)}{1 - c(s)e}
\end{align*}
\]

Controller

\[
\omega = \frac{vc(s) \cos(\theta_e)}{1 - c(s)e} - g_1(e, \theta_e, t)\theta_e - g_2 v \frac{\sin(\theta_e)}{\theta_e} e
\]

A couple of key lessons learned …

- Bound closed-loop systems in the error coordinates (note the term cancellation)
Example: Trajectory Tracking

Simplified Dubins Car

\[ \dot{x} = v \cos(\theta) \quad \dot{y} = v \sin(\theta) \quad \dot{\theta} = \omega \quad v \in [5, 6] \text{ (m/s)} \]

Closed-Loop Error Dynamics

\[ \dot{s} = \frac{v \cos(\theta_e)}{1 - c(s)} \quad \dot{e} = v \sin(\theta_e) \quad \dot{\theta}_e = -g_1(e, \theta_e, t)\theta_e - g_2v \frac{\sin(\theta_e)}{\theta_e}e \]

A couple of key lessons learned …

- Bound closed-loop systems in the error coordinates (note the term cancellation)
Example: Trajectory Tracking

Simplified Dubins Car
\[ \dot{x} = v \cos(\theta) \quad \dot{y} = v \sin(\theta) \quad \dot{\theta} = \omega \quad v \in [5, 6] \ (\text{m/s}) \]

Closed-Loop Error Dynamics
\[ \dot{s} = \frac{v \cos(\theta_e)}{1 - c(s)} \quad \dot{e} = v \sin(\theta_e) \quad \dot{\theta}_e = -g_1(e, \theta_e, t)\theta_e - g_2 v \frac{\sin(\theta_e)}{\theta_e} e \]

A couple of key lessons learned …
- Bound closed-loop systems in the error coordinates (note the term cancellation)
- A great choice of redundant state: …
Example: Trajectory Tracking

**Simplified Dubins Car**

\[
\begin{align*}
\dot{x} &= v \cos(\theta) & \dot{y} &= v \sin(\theta) & \dot{\theta} &= \omega
\end{align*}
\]

\[v \in [5, 6] \text{ (m/s)}\]

**Closed-Loop Error Dynamics**

\[
\begin{align*}
\dot{s} &= \frac{v \cos(\theta_e)}{1 - c(s)} & \dot{e} &= v \sin(\theta_e) & \dot{\theta}_e &= -g_1(e, \theta_e, t)\theta_e - g_2v \frac{\sin(\theta_e)}{\theta_e} e
\end{align*}
\]

A couple of key lessons learned ...

- Bound closed-loop systems in the error coordinates (note the term cancellation)
- A great choice of redundant state: a Lyapunov function
Example: Trajectory Tracking

Simplified Dubins Car

\[
\begin{align*}
\dot{x} &= v \cos(\theta) \\
\dot{y} &= v \sin(\theta) \\
\dot{\theta} &= \omega
\end{align*}
\]

\( v \in [5, 6] \) (m/s)

Closed-Loop Error Dynamics

\[
\begin{align*}
\dot{s} &= \frac{v \cos(\theta_e)}{1 - c(s)} \\
\dot{e} &= v \sin(\theta_e) \\
\dot{\theta}_e &= -g_1(e, \theta_e, t) \theta_e - g_2v \frac{\sin(\theta_e)}{\theta_e} e
\end{align*}
\]

Lyapunov Function

\[
V \equiv \frac{1}{2} \left( e^2 + \frac{1}{g_2} \theta_e^2 \right) \implies \dot{V} = \frac{1}{2} \left( 2e \dot{e} + \frac{2}{g_2} \theta_e \dot{\theta}_e \right)
\]

\[
= ev \sin(\theta_e) - \frac{1}{g_2} \theta_e \left( g_1(e, \theta_e, t) \theta_e + g_2v \frac{\sin(\theta_e)}{\theta_e} e \right)
\]

\[
= -\frac{g_1(e, \theta_e, t)}{g_2} \theta_e^2 \leq 0
\]
**Example: Trajectory Tracking**

### Simplified Dubins Car
\[
\begin{align*}
\dot{x} &= v \cos(\theta) \\
\dot{y} &= v \sin(\theta) \\
\dot{\theta} &= \omega
\end{align*}
\]
\[v \in [5, 6] \text{ (m/s)}\]

### Closed-Loop Error Dynamics
\[
\begin{align*}
\dot{s} &= \frac{v \cos(\theta_e)}{1 - c(s)} \\
\dot{e} &= v \sin(\theta_e) \\
\dot{\theta}_e &= -g_1(e, \theta_e, t)\theta_e - g_2v \frac{\sin(\theta_e)}{\theta_e}e
\end{align*}
\]

### Lyapunov Function
\[
V \equiv \frac{1}{2} \left( e^2 + \frac{1}{g_2} \theta_e^2 \right) \quad \Rightarrow \quad \dot{V} = \frac{1}{2} \left( 2e\dot{e} + \frac{2}{g_2} \theta_e \dot{\theta}_e \right)
\]
\[
= ev \sin(\theta_e) - \frac{1}{g_2} \theta_e \left( g_1(e, \theta_e, t)\theta_e + g_2v \frac{\sin(\theta_e)}{\theta_e}e \right)
\]
\[
= -\frac{g_1(e, \theta_e, t)}{g_2} \theta_e^2 \leq 0
\]

Simple ODE for bounding
Example: Trajectory Tracking

Computational time for the methods

- Red: SDI
- Green: DI with constraints using the $\kappa$-operator
- Blue: DI with constraints using the inverse function

Graphs showing the performance of different methods over time.
Example: Trajectory Tracking
Automatic Invariant Construction

\[ \dot{x}(t, p) = f(t, p, x(t, p)), \quad x(t_0, p) = x_0(p) \]

Need general methods for atomically constructing effective invariants
Eliminate the need for deep user insights in every problem

Developed a new approach based on parametric sensitivities
Key ingredients are the Mean Value Theorem and the forward sensitivity equations
Automatic Invariant Construction

\[ \dot{x}(t, p) = f(t, p, x(t, p)), \quad x(t_0, p) = x_0(p) \]

Need general methods for atomically constructing effective invariants
Eliminate the need for deep user insights in every problem

Developed a new approach based on parametric sensitivities
Key ingredients are the Mean Value Theorem and the forward sensitivity equations

**Mean Value Differential Inequalities (MVDI)**

- Define forward sensitivities as redundant states: \( s_i(t, p) \equiv \frac{\partial x_i}{\partial p} (t, p) \)
- Augment original ODEs with forward sensitivity system:

\[
\frac{d}{dt} \begin{bmatrix} x(t, p) \\ S(t, p) \end{bmatrix} = \begin{bmatrix} f(t, p, x(t, p)) \\ \frac{\partial f}{\partial x} (t, p, x(t, p))S(t, p) + \frac{\partial f}{\partial p} (t, p, x(t, p)) \end{bmatrix}
\]
Automatic Invariant Construction

\[
\frac{d}{dt} \begin{bmatrix} x(t, p) \\ S(t, p) \end{bmatrix} = \begin{bmatrix} f(t, p, x(t, p)) \\ \frac{\partial f}{\partial x}(t, p, x(t, p))S(t, p) + \frac{\partial f}{\partial p}(t, p, x(t, p)) \end{bmatrix}
\]

- States are related by the Mean Value Theorem
  \[x_i(t, p) = x_i(t, \hat{p}) + s_i(t, \xi)(p - \hat{p}), \quad \xi \in P\]

- This is not a true invariant, but it admits a rigorous bound refinement
  \[s_i(t, \xi) \in S_i(t) \quad \Rightarrow \quad x_i(t, p) \in x_i(t, \hat{p}) + S_i(t)(p - \hat{p})\]

- Apply constrained DI to augmented system with

\[G \equiv \{(t, p, x, S) : x = \hat{x}(t) + \tilde{S}(p - \hat{p}), \quad \tilde{S} \in S(t)\}\]

\(1\) Dependence of G on current bounds completely violates existing DI theory. Had to fix this.
Mean Value Differential Inequalities

Use the mean-value enclosure to refine \((X,P)\) continuously as bounds are integrated forward in time.

\[
\begin{align*}
(\hat{X}(t), \hat{P}) & \leftarrow \text{Refine}(X(t), S(t), P) \\
(\hat{X}_i^L(t), \hat{P}_i^L) & \leftarrow \text{Refine}(\beta_i^L(X(t)), S(t), P) \\
(\hat{X}_i^U(t), \hat{P}_i^U) & \leftarrow \text{Refine}(\beta_i^U(X(t)), S(t), P)
\end{align*}
\]

\[
\begin{align*}
\dot{x}_i^L(t) &= f_i^L(\hat{P}_i^L, \hat{X}_i^L(t)) \\
\dot{x}_i^U(t) &= f_i^U(\hat{P}_i^U, \hat{X}_i^U(t)) \\
\dot{s}_{ij}^L(t) &= \left[ \frac{\partial f_i}{\partial x}(\hat{P}, \hat{X}(t)) S_j(t) + \frac{\partial f_i}{\partial p_j}(\hat{P}, \hat{X}(t)) \right]^L \\
\dot{s}_{ij}^U(t) &= \left[ \frac{\partial f_i}{\partial x}(\hat{P}, \hat{X}(t)) S_j(t) + \frac{\partial f_i}{\partial p_j}(\hat{P}, \hat{X}(t)) \right]^U
\end{align*}
\]

\[
x = \hat{x}(t) + \tilde{S}(p - \hat{p})
\]

\[
x \in X_i(t) \\
\tilde{S} \in S(t) \\
p \in P
\]

Sharp bounds for many difficult problems

Requires a nonlinear refinement operator
Example: Anaerobic Digester with pH Self-Regulation and Liquid-Gas Transfer

\[
\begin{align*}
\dot{X}_1 &= (\mu_1(S_1) - \alpha D)X_1 \\
\dot{X}_2 &= (\mu_2(S_2) - \alpha D)X_2 \\
\dot{Z} &= D(Z^{in} - Z) \\
\dot{S}_1 &= D(S_1^{in} - S_1) - k_1\mu_1(S_1)X_1 \\
\dot{S}_2 &= D(S_2^{in} - S_2) + k_2\mu_1(S_1)X_1 - k_3\mu_2(S_2)X_2 \\
\dot{C} &= D(C^{in} - C) - q_{CO_2} + k_4\mu_1(S_1)X_1 + k_5\mu_2(S_2)X_2 \\
q_{CO_2} &= k_L a(C + S_2 - Z - K_H P_{CO_2}) \\
P_{CO_2} &= \left( \phi_{CO_2} - \sqrt{\phi_{CO_2}^2 - 4K_H P_t(C + S_2 - Z)} \right)(2K_H)^{-1} \\
\phi_{CO_2} &= C + S_2 - Z + K_H P_t + \frac{k_6}{k_L a} \mu_2(S_2)X_2 \\
\mu_1(S_1) &= \bar{\mu}_1 \frac{S_1}{S_1 + K_{S_1}} \\
\mu_2(S_2) &= \bar{\mu}_2 \frac{S_2}{S_2 + K_{S_2} + S_2^2 / K_{I_2}} 
\end{align*}
\]


- Complex dynamics due to pH self-regulating and liquid gas transfer
- Fast dynamics on a time-scale of hours
- Slow dynamics on a time-scale of days

Uncertainty

\[
\begin{align*}
X_{1,0} &\in [0.49, 0.51] & k_1 &\in [42.14, 42.98] \\
X_{2,0} &\in [0.98, 1.02] & k_2 &\in [116.5, 118.24] \\
C_0 &\in [39.2, 40.8] 
\end{align*}
\]
Example: Anaerobic Digester with pH Self-Regulation and Liquid-Gas Transfer

![Graph showing S2 Concentration (mmol-L⁻¹) vs Time (days)]

Shen and Scott, CDC, 2018
Example: Aircraft Trajectory Tracking

**Fixed Wing Aircraft Model**

\[
\begin{align*}
\dot{x} &= v_{xy} \cos(\psi) & \dot{v}_{xy} &= a_{xy} \\
\dot{y} &= v_{xy} \sin(\psi) & \dot{v}_z &= a_z \\
\dot{z} &= v_z \\
\dot{\psi} &= \frac{g}{v_{xy}} \tan(\theta)
\end{align*}
\]

**Controller**

\[
\begin{align*}
a_{xy} &= k_5 \epsilon_x + k_6 (v_{xy,d} - v_{xy}) \\
a_z &= k_7 \epsilon_z + k_8 (v_{z,d} - v_z) \\
\omega &= k_4 (k_1 \epsilon_y + k_2 (\psi_d - \psi) + k_3 (\dot{\psi}_d - \dot{\psi}) - \theta) \\
\epsilon_x &= \cos(\psi_d) (x_d - x) + \sin(\psi_d) (y_d - y) \\
\epsilon_y &= -\sin(\psi_d) (x_d - x) + \cos(\psi_d) (y_d - y) \\
\epsilon_z &= z_d - z.
\end{align*}
\]

**Uncertainty:**

\[x_0, y_0, z_0 \in [-3, 3] \text{ (m)}\]
Example: Aircraft Trajectory Tracking

Sharp bounds on 10 seconds of flight time in hundredths of a second in CPU time

$\begin{align*}
  & t_f = 10 \text{ s} \\
  & t_0 = 0 \text{ s}
\end{align*}$

Trajectory 1.4E-4 CPUs
SDI 5.8E-3 CPUs
MVDI 1.6E-2 CPUs
Example: Aircraft Trajectory Tracking

Sharp bounds on 10 seconds of flight time in hundredths of a second in CPU time

<table>
<thead>
<tr>
<th>Trajectory</th>
<th>1.4E-4 CPUs</th>
</tr>
</thead>
<tbody>
<tr>
<td>SDI</td>
<td>5.8E-3 CPUs</td>
</tr>
<tr>
<td>MVDI</td>
<td>1.6E-2 CPUs</td>
</tr>
</tbody>
</table>
Conclusions and Future Work

Conclusions
- Developed new methods for computing rigorous, accurate bounds on the solutions of nonlinear dynamic models under uncertainty
- Advances based on two key ideas
  - Manufactured/redundant constraints
  - Mean-value differential inequalities (automatic constraint generation)
- Much more accurate than existing methods of similar computational complexity
- In many cases, high-accuracy bounds are achieved fast enough for online control applications

Future Work
- Adding many constraints eventually compromises efficiency
- Need better theory or heuristics for determining effective constraints
- Need additional strategies methods for automatically generating constraints
  *(symbolic methods?)*

This research was supported by the US Air Force Office of Scientific Research under award number FA9550-16-1-0158