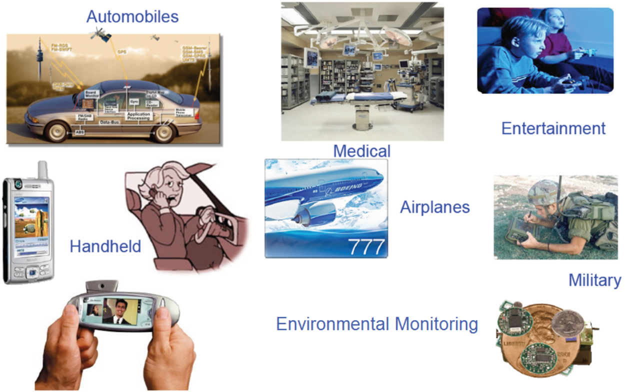
Synthesizing Invariant Barrier Certificates
via Difference-of-Convex Programming

Qiuye Wang, **Mingshuai Chen**, Bai Xue, Naijun Zhan, Joost-Pieter Katoen

CAV · July 2021

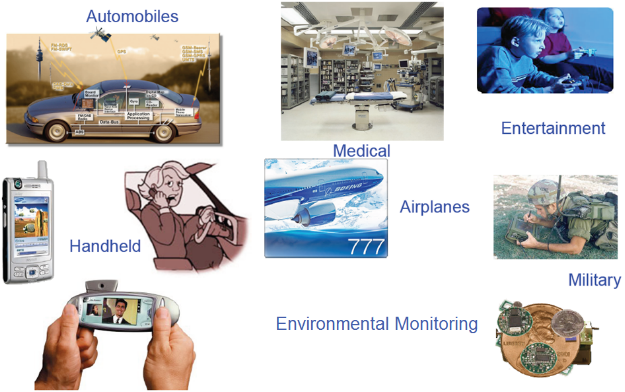
Cyber-Physical Systems (CPS)

An open, interconnected form of embedded systems that integrates capabilities of computing, communication and control, among which many are safety-critical.



Cyber-Physical Systems (CPS)

An open, interconnected form of embedded systems that integrates capabilities of computing, communication and control, among which many are safety-critical.



"How can we provide people with CPS they can bet their lives on?"

[Jeannette Wing]

Hybrid Systems

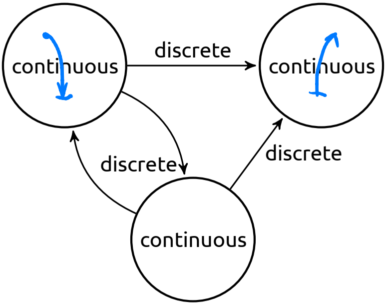


Figure – Macro : switching modes

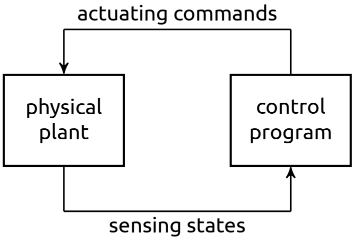


Figure – Micro : closed-loop feedback

Hybrid Systems

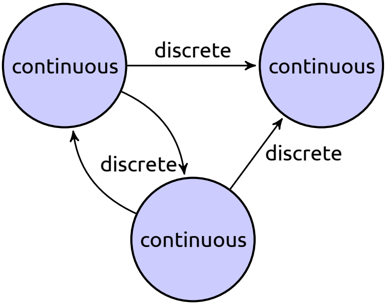


Figure – Macro : switching modes

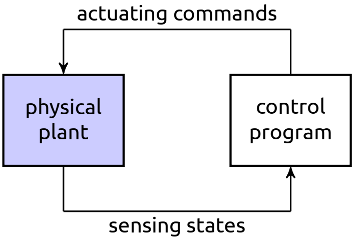


Figure – Micro : closed-loop feedback

Differential Dynamical Systems

Ordinary Differential Equations (ODEs) of the autonomous type :

$$\dot{\mathbf{x}} = \boldsymbol{f}(\mathbf{x})$$

Differential Dynamical Systems

Ordinary Differential Equations (ODEs) of the autonomous type :

$$\dot{\mathbf{x}} \;=\; \boldsymbol{f}(\mathbf{x})$$

⇒ Unique *solution* (*trajectory*) : $\zeta_{\mathbf{x}_0} : [0, T) \rightarrow \mathbb{R}^n$.

Differential Dynamical Systems

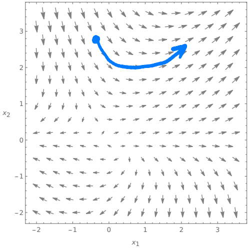
Ordinary Differential Equations (ODEs) of the autonomous type :

$$\dot{\mathbf{x}} = \boldsymbol{f}(\mathbf{x})$$

⇒ Unique *solution* (*trajectory*) : $\zeta_{\mathbf{x}_0} : [0, T) \to \mathbb{R}^n$.

Example (running [Djaballah *et al.*, 2017])

$$\dot{\mathbf{x}} = \begin{pmatrix} \dot{x}_1 \\ \dot{x}_2 \end{pmatrix} = \begin{pmatrix} x_1 + x_2 \\ x_1 x_2 - 0.5 x_2^2 + 0.1 \end{pmatrix}.$$



Differential Dynamical Systems

Ordinary Differential Equations (ODEs) of the autonomous type :

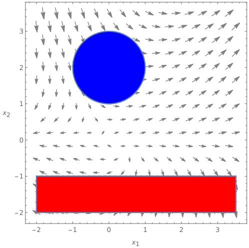
$$\dot{\mathrm{x}} = \boldsymbol{f}(\mathrm{x})$$

⇒ Unique *solution* (*trajectory*) : $\zeta_{\mathrm{x}_0} : [0, T) \to \mathbb{R}^n$.

Example (running [Djaballah *et al.*, 2017])

$$\dot{\mathrm{x}} = \begin{pmatrix} \dot{x}_1 \\ \dot{x}_2 \end{pmatrix} = \begin{pmatrix} x_1 + x_2 \\ x_1 x_2 - 0.5 x_2^2 + 0.1 \end{pmatrix}.$$

$\mathcal{X}_0 = \{\mathrm{x} \mid \mathcal{I}(\mathrm{x}) \le 0\}$ with $\mathcal{I}(\mathrm{x}) = x_1^2 + (x_2 - 2)^2 - 1$;
$\mathcal{X}_u = \{\mathrm{x} \mid \mathcal{U}(\mathrm{x}) \le 0\}$ with $\mathcal{U}(\mathrm{x}) = x_2 + 1$;
$\mathcal{X} = \mathbb{R}^n$.



Differential Dynamical Systems

Ordinary Differential Equations (ODEs) of the autonomous type :

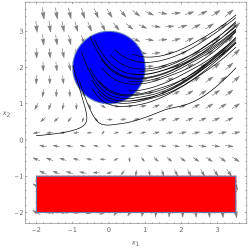
$$\dot{\mathrm{x}} \;=\; \boldsymbol{f}(\mathrm{x})$$

⇒ Unique *solution* (*trajectory*) : $\zeta_{\mathrm{x}_0} : [0, T) \to \mathbb{R}^n$.

Example (running [Djaballah *et al.*, 2017])

$$\dot{\mathrm{x}} = \begin{pmatrix} \dot{x}_1 \\ \dot{x}_2 \end{pmatrix} = \begin{pmatrix} x_1 + x_2 \\ x_1 x_2 - 0.5 x_2^2 + 0.1 \end{pmatrix}.$$

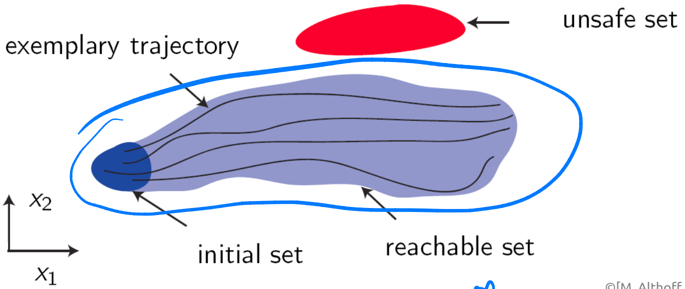
$\mathcal{X}_0 = \{\mathrm{x} \mid \mathcal{I}(\mathrm{x}) \leq 0\}$ with $\mathcal{I}(\mathrm{x}) = x_1^2 + (x_2 - 2)^2 - 1$;
$\mathcal{X}_u = \{\mathrm{x} \mid \mathcal{U}(\mathrm{x}) \leq 0\}$ with $\mathcal{U}(\mathrm{x}) = x_2 + 1$;
$\mathcal{X} = \mathbb{R}^n$.



Safety Verification of ODEs

Given $\mathcal{X} \subseteq \mathbb{R}^n$, $\mathcal{X}_0 \subseteq \mathcal{X}$, $\mathcal{X}_u \subseteq \mathcal{X}$, whether

$$\mathcal{R}_{\mathcal{X}_0} \cap \mathcal{X}_u = \emptyset \quad ?$$



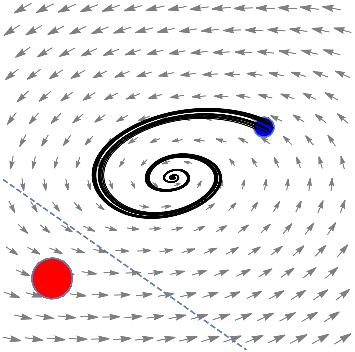
©[M. Althoff, 2010]

∞

⇒ System is safe, if no trajectory enters $\mathcal{X}_u$ over $[0, T)$, and unsafe otherwise.

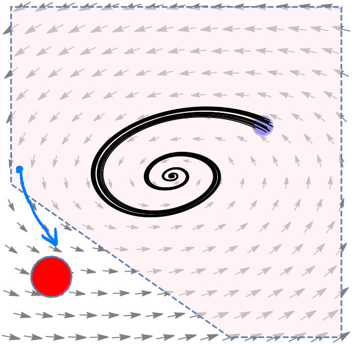
Barrier Certificates (BCs)

$$\forall \mathbf{x}_0 \in \mathcal{X}_0.\, \forall t \in [0, T)\colon\, B(\zeta_{\mathbf{x}_0}(t)) \leq 0,$$
$$\forall \mathbf{x} \in \mathcal{X}_u\colon\, B(\mathbf{x}) > 0.$$



Barrier Certificates (BCs)

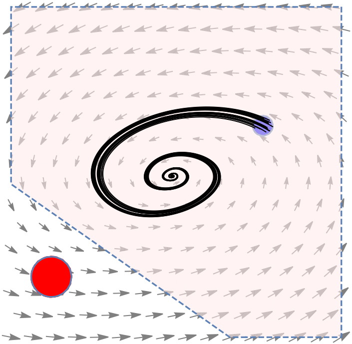
$$\forall \mathbf{x}_0 \in \mathcal{X}_0.\, \forall t \in [0, T) \colon\ B(\zeta_{\mathbf{x}_0}(t)) \leq 0,$$
$$\forall \mathbf{x} \in \mathcal{X}_u \colon\ B(\mathbf{x}) > 0.$$

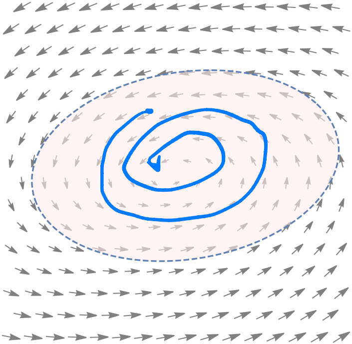


Barrier Certificates (BCs) vs. Inductive Invariants

$$\forall \mathbf{x}_0 \in \mathcal{X}_0.\, \forall t \in [0, T)\colon\ B(\zeta_{\mathbf{x}_0}(t)) \leq 0,$$
$$\forall \mathbf{x} \in \mathcal{X}_u\colon\ B(\mathbf{x}) > 0.$$

$$\forall \mathbf{x}_0 \in \Psi.\, \forall t \in [0, T)\colon\ \zeta_{\mathbf{x}_0}(t) \in \Psi,$$

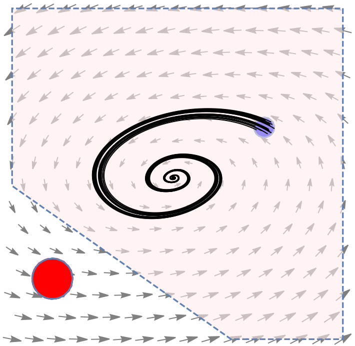


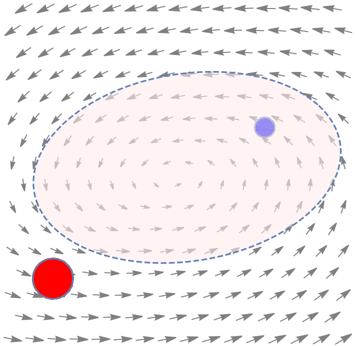


Barrier Certificates (BCs) vs. Inductive Invariants

$$\forall \mathbf{x}_0 \in \mathcal{X}_0. \forall t \in [0, T): B(\zeta_{\mathbf{x}_0}(t)) \leq 0,$$
$$\forall \mathbf{x} \in \mathcal{X}_u: B(\mathbf{x}) > 0.$$

$$\forall \mathbf{x}_0 \in \Psi. \forall t \in [0, T): \zeta_{\mathbf{x}_0}(t) \in \Psi,$$
$$\mathcal{X}_0 \subseteq \Psi \text{ and } \mathcal{X}_u \cap \Psi = \emptyset.$$

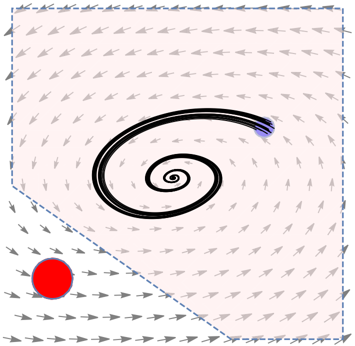


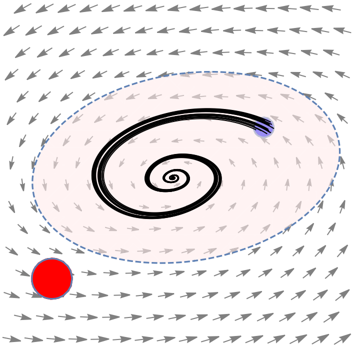


Barrier Certificates (BCs) vs. Inductive Invariants

$$\forall \mathbf{x}_0 \in \mathcal{X}_0.\, \forall t \in [0, T) \colon B(\zeta_{\mathbf{x}_0}(t)) \leq 0,$$
$$\forall \mathbf{x} \in \mathcal{X}_u \colon B(\mathbf{x}) > 0.$$

$$\forall \mathbf{x}_0 \in \Psi.\, \forall t \in [0, T) \colon \zeta_{\mathbf{x}_0}(t) \in \Psi,$$
$$\mathcal{X}_0 \subseteq \Psi \text{ and } \mathcal{X}_u \cap \Psi = \emptyset.$$





BC Conditions vs. Inductive Invariance

semantic cond. [Sogokon *et al.*, 2018] $\Longleftarrow$ inductive invariance [Platzer & Clarke, 2008] [Liu *et al.*, 2011]

BC Conditions vs. Inductive Invariance

semantic cond. [Sogokon *et al.*, 2018] $\Longleftarrow$ inductive invariance [Platzer & Clarke, 2008] [Liu *et al.*, 2011]

$\Uparrow$

original cond. [Prajna & Jadbabaie, 2004]
exponential-type cond. [Kong *et al.*, 2013]
Darboux-type cond. [Zeng *et al.*, 2016]
general cond. [Dai *et al.*, 2017]
vector cond. [Sogokon *et al.*, 2018]

⋮

non-convex cond.'s [Yang *et al.*, 2015] [Zhang *et al.*, 2018], [Chen *et al.*, 2020]

BC Conditions vs. Inductive Invariance

semantic cond. [Sogokon *et al.*, 2018]

inductive invariance [Platzer & Clarke, 2008] [Liu *et al.*, 2011]

original cond. [Prajna & Jadbabaie, 2004]
exponential-type cond. [Kong *et al.*, 2013]
Darboux-type cond. [Zeng *et al.*, 2016]
general cond. [Dai *et al.*, 2017]
vector cond. [Sogokon *et al.*, 2018]

⋮

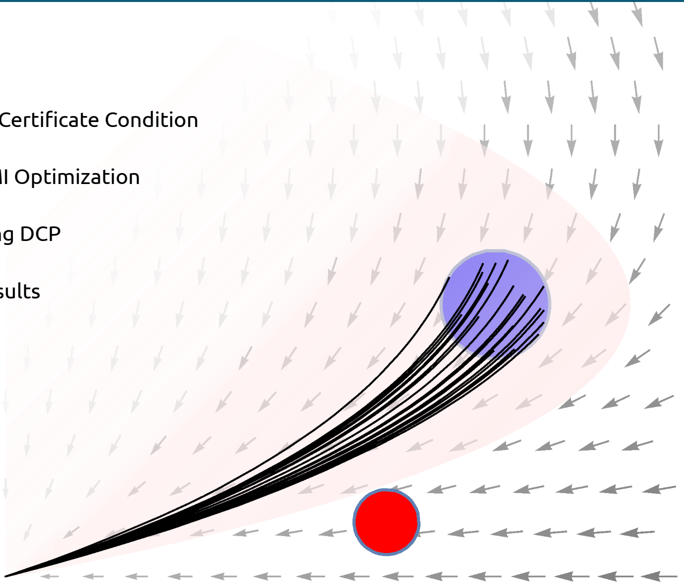
non-convex cond.'s [Yang *et al.*, 2015] [Zhang *et al.*, 2018], [Chen *et al.*, 2020]

invariant BC cond.

Q. Wang, M. Chen, B. Xue, N. Zhan, J.-P. Katoen : *Synthesizing Invariant Barrier Certificates via DCP*. CAV'21.

Outline

1. Invariant Barrier-Certificate Condition
2. Encoding as a BMI Optimization
3. Solving BMIs using DCP
4. Experimental Results

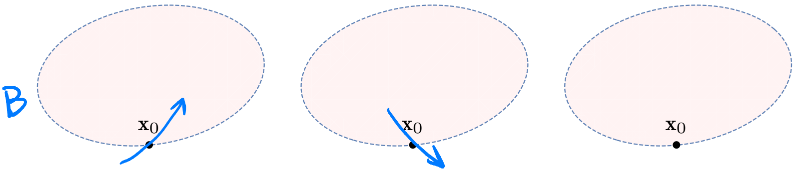


Lie Derivatives

$$\mathcal{L}_{\boldsymbol{f}}^{k} B(\mathbf{x}) \mathrel{\widehat{=}} \begin{cases} B(\mathbf{x}), & k = 0, \\ \left\langle \frac{\partial}{\partial \mathbf{x}} \mathcal{L}_{\boldsymbol{f}}^{k-1} B(\mathbf{x}), \boldsymbol{f}(\mathbf{x}) \right\rangle, & k > 0. \end{cases} \qquad = \frac{\partial B}{\partial t}$$

Lie Derivatives

$$\mathcal{L}_{\boldsymbol{f}}^{k} B(\mathbf{x}) \;\widehat{=}\; \begin{cases} B(\mathbf{x}), & k = 0, \\ \left\langle \frac{\partial}{\partial \mathbf{x}} \mathcal{L}_{\boldsymbol{f}}^{k-1} B(\mathbf{x}), \boldsymbol{f}(\mathbf{x}) \right\rangle, & k > 0. \end{cases}$$



$\mathcal{L}_{\boldsymbol{f}}^{1} B(\mathbf{x}_0) < 0$

$\mathcal{L}_{\boldsymbol{f}}^{1} B(\mathbf{x}_0) > 0$

$\mathcal{L}_{\boldsymbol{f}}^{1} B(\mathbf{x}_0) = 0$

$\mathcal{L}_{\boldsymbol{f}}^{2} B(\mathbf{x}_0) = 0$

$\vdots$

$\mathcal{L}_{\boldsymbol{f}}^{N_{B,\boldsymbol{f}}} B(\mathbf{x}_0) \leq 0$

Invariant Barrier Certificates

1. $\forall \mathbf{x} \in \mathcal{X}_0 :\ B(\mathbf{x}) \leq 0,$ (initial)
2. $\forall \mathbf{x} \in \mathcal{X}_u :\ B(\mathbf{x}) > 0,$ (separation)
3. $\forall \mathbf{x} \in \mathbb{R}^n :\ \bigwedge_{i=1}^{N_{B,\boldsymbol{f}}} \left(\left(\bigwedge_{j=0}^{i-1} \mathcal{L}_{\boldsymbol{f}}^{j} B(\mathbf{x}) = 0 \right) \implies \mathcal{L}_{\boldsymbol{f}}^{i} B(\mathbf{x}) \leq 0 \right).$ (consecution)

Invariant Barrier Certificates

1. $\forall \mathbf{x} \in \mathcal{X}_0 \colon\ B(\mathbf{x}) \le 0$, (initial)
2. $\forall \mathbf{x} \in \mathcal{X}_u \colon\ B(\mathbf{x}) > 0$, (separation)
3. $\forall \mathbf{x} \in \mathbb{R}^n \colon\ \bigwedge_{i=1}^{N_{B,\boldsymbol{f}}} \left(\left(\bigwedge_{j=0}^{i-1} \mathcal{L}_{\boldsymbol{f}}^{j} B(\mathbf{x}) = 0 \right) \implies \mathcal{L}_{\boldsymbol{f}}^{i} B(\mathbf{x}) \le 0 \right)$. (consecution)

$$\Updownarrow$$

$\Psi = \{\mathbf{x} \mid B(\mathbf{x}) \le 0\}$ is an invariant (separating $\mathcal{X}_0$ and $\mathcal{X}_u$).

Sufficient Condition for Invariant BCs

For $B(\mathbf{x}) \in \mathbb{R}[\mathbf{x}]$, $\epsilon \in \mathbb{R}^+$, $v_{i,j} \in \mathbb{R}[\mathbf{x}]$ and sum-of-squares (SOS) polynomials $\sigma(\mathbf{x}), \sigma'(\mathbf{x})$, the following polynomials are SOS :

1. $-B(\mathbf{x}) + \sigma(\mathbf{x}) \cdot \mathcal{I}(\mathbf{x})$, (initial)
2. $B(\mathbf{x}) + \sigma'(\mathbf{x}) \cdot \mathcal{U}(\mathbf{x}) - \epsilon$, (separation)
3. for all $1 \leq i \leq N_{B,\boldsymbol{f}}$, $-\mathcal{L}_{\boldsymbol{f}}^{i} B(\mathbf{x}) + \sum_{j=0}^{i-1} v_{i,j}(\mathbf{x}) \cdot \mathcal{L}_{\boldsymbol{f}}^{j} B(\mathbf{x})$. (consecution)

Sufficient Condition for Invariant BCs

For $B(\mathbf{x}) \in \mathbb{R}[\mathbf{x}]$, $\epsilon \in \mathbb{R}^+$, $v_{i,j} \in \mathbb{R}[\mathbf{x}]$ and sum-of-squares (SOS) polynomials $\sigma(\mathbf{x}), \sigma'(\mathbf{x})$, the following polynomials are SOS :

1. $-B(\mathbf{x}) + \sigma(\mathbf{x}) \cdot \mathcal{I}(\mathbf{x})$, (initial)
2. $B(\mathbf{x}) + \sigma'(\mathbf{x}) \cdot \mathcal{U}(\mathbf{x}) - \epsilon$, (separation)
3. for all $1 \leq i \leq N_{B,\boldsymbol{f}}$, $-\mathcal{L}^i_{\boldsymbol{f}} B(\mathbf{x}) + \sum_{j=0}^{i-1} \underbrace{v_{i,j}(\mathbf{x})}_{\text{unknown}} \cdot \mathcal{L}^j_{\boldsymbol{f}} \underbrace{B(\mathbf{x})}_{\text{unknown}}$ (consecution)

bilinearity arises!

Sufficient Condition for Invariant BCs

For $B(\mathbf{x}) \in \mathbb{R}[\mathbf{x}]$, $\epsilon \in \mathbb{R}^+$, $v_{i,j} \in \mathbb{R}[\mathbf{x}]$ and sum-of-squares (SOS) polynomials $\sigma(\mathbf{x}), \sigma'(\mathbf{x})$, the following polynomials are SOS :

1. $-B(\mathbf{x}) + \sigma(\mathbf{x}) \cdot \mathcal{I}(\mathbf{x})$, (initial)
2. $B(\mathbf{x}) + \sigma'(\mathbf{x}) \cdot \mathcal{U}(\mathbf{x}) - \epsilon$, (separation)
3. for all $1 \leq i \leq N_{B,\boldsymbol{f}}$, $-\mathcal{L}_{\boldsymbol{f}}^{i} B(\mathbf{x}) + \sum_{j=0}^{i-1} \underbrace{v_{i,j}(\mathbf{x})}_{\text{unknown}} \cdot \underbrace{\mathcal{L}_{\boldsymbol{f}}^{j} B(\mathbf{x})}_{\text{unknown}}$ (consecution)

bilinearity arises!

Example (running)

Assume templates $B(\mathrm{a}, \mathbf{x}) = a x_2$ and $v(\mathrm{s}, \mathbf{x}) = s_0 + s_1 x_1 + s_2 x_2$, we have $N_{B,\boldsymbol{f}} = 1$, and

$$-\underbrace{a\left(x_1 x_2 - 0.5 x_2^2 + 0.1\right)}_{\mathcal{L}_{\boldsymbol{f}}^{1} B} + \underbrace{\left(s_0 + s_1 x_1 + s_2 x_2\right)}_{v} \cdot \underbrace{a x_2}_{\mathcal{L}_{\boldsymbol{f}}^{0} B} \in \Sigma^{\leq 2}[\mathbf{x}] \qquad \text{(consecution)}$$

Sufficient Condition for Invariant BCs

For $B(\mathbf{x}) \in \mathbb{R}[\mathbf{x}]$, $\epsilon \in \mathbb{R}^+$, $v_{i,j} \in \mathbb{R}[\mathbf{x}]$ and sum-of-squares (SOS) polynomials $\sigma(\mathbf{x}), \sigma'(\mathbf{x})$, the following polynomials are SOS :

1. $-B(\mathbf{x}) + \sigma(\mathbf{x}) \cdot \mathcal{I}(\mathbf{x})$, (initial)
2. $B(\mathbf{x}) + \sigma'(\mathbf{x}) \cdot \mathcal{U}(\mathbf{x}) - \epsilon$, (separation)
3. for all $1 \leq i \leq N_{B,\boldsymbol{f}}$, $-\mathcal{L}^i_{\boldsymbol{f}} B(\mathbf{x}) + \sum_{j=0}^{i-1} \underbrace{v_{i,j}(\mathbf{x})}_{\text{unknown}} \cdot \underbrace{\mathcal{L}^j_{\boldsymbol{f}} B(\mathbf{x})}_{\text{unknown}}$ (consecution)

bilinearity arises!

Example (running)

Assume templates $B(\mathrm{a}, \mathbf{x}) = a x_2$ and $v(\mathrm{s}, \mathbf{x}) = s_0 + s_1 x_1 + s_2 x_2$, we have $N_{B,\boldsymbol{f}} = 1$, and

$$-\underbrace{a\left(x_1 x_2 - 0.5 x_2^2 + 0.1\right)}_{\mathcal{L}^1_{\boldsymbol{f}} B} + \underbrace{\left(s_0 + s_1 x_1 + s_2 x_2\right)}_{v} \cdot \underbrace{a x_2}_{\mathcal{L}^0_{\boldsymbol{f}} B} \in \Sigma^{\leq 2}[\mathbf{x}] \qquad \text{(consecution)}$$

⇒ Mitigate bilinearity by difference-of-convex programming (DCP).

Bilinear Matrix Inequality (BMI) Feasibility

$$h(\mathbf{a}, \mathbf{s}, \mathbf{x}) \in \Sigma^{\leq 2d}[\mathbf{x}]$$

Bilinear Matrix Inequality (BMI) Feasibility

$(1, x_1x_2, x_1, x_2, \cdots x_n^d)$

$$
\begin{aligned}
h(\mathbf{a},\mathbf{s},\mathbf{x}) &\in \Sigma^{\leq 2d}[\mathbf{x}] \\
&\Updownarrow \; h(\mathbf{a},\mathbf{s},\mathbf{x}) = \mathbf{b}^{\mathsf{T}} Q(\mathbf{a},\mathbf{s})\mathbf{b} \\
Q(\mathbf{a},\mathbf{s}) &\succeq 0
\end{aligned}
$$

Bilinear Matrix Inequality (BMI) Feasibility

$$
\begin{aligned}
h(\mathbf{a}, \mathbf{s}, \mathbf{x}) \;&\in\; \Sigma^{\leq 2d}[\mathbf{x}] \\
&\Updownarrow \; h(\mathbf{a}, \mathbf{s}, \mathbf{x}) = \mathbf{b}^{\mathsf{T}} Q(\mathbf{a}, \mathbf{s}) \mathbf{b} \\
Q(\mathbf{a}, \mathbf{s}) \;&\succeq\; 0 \\
&\Updownarrow \; \mathcal{F}(\mathbf{a}, \mathbf{s}) = -Q(\mathbf{a}, \mathbf{s}) \\
\mathcal{F}(\mathbf{a}, \mathbf{s}) \;&\preceq\; 0
\end{aligned}
$$

Bilinear Matrix Inequality (BMI) Feasibility

$$
\begin{aligned}
h(\mathbf{a},\mathbf{s},\mathbf{x}) &\in \Sigma^{\leq 2d}[\mathbf{x}] \\
&\Updownarrow\; h(\mathbf{a},\mathbf{s},\mathbf{x}) = \mathbf{b}^{\mathsf{T}} Q(\mathbf{a},\mathbf{s})\mathbf{b} \\
Q(\mathbf{a},\mathbf{s}) &\succeq 0 \\
&\Updownarrow\; \mathcal{F}(\mathbf{a},\mathbf{s}) = -Q(\mathbf{a},\mathbf{s}) \\
\mathcal{F}(\mathbf{a},\mathbf{s}) &\preceq 0
\end{aligned}
$$

Example (running)

$$-a\left(x_1 x_2 - 0.5x_2^2 + 0.1\right) + (s_0 + s_1 x_1 + s_2 x_2)\cdot a x_2 \in \Sigma^{\leq 2}[\mathbf{x}] \qquad \text{(consecution)}$$

Bilinear Matrix Inequality (BMI) Feasibility

$$
\begin{aligned}
h(\mathbf{a},\mathbf{s},\mathbf{x}) &\in \Sigma^{\leq 2d}[\mathbf{x}] \\
&\Big\Updownarrow\; h(\mathbf{a},\mathbf{s},\mathbf{x}) = \mathbf{b}^{\mathsf{T}} Q(\mathbf{a},\mathbf{s})\mathbf{b} \\
Q(\mathbf{a},\mathbf{s}) &\succeq 0 \\
&\Big\Updownarrow\; \mathcal{F}(\mathbf{a},\mathbf{s}) = -Q(\mathbf{a},\mathbf{s}) \\
\mathcal{F}(\mathbf{a},\mathbf{s}) &\preceq 0
\end{aligned}
$$

Example (running)

$$
-a\left(x_1x_2 - 0.5x_2^2 + 0.1\right) + (s_0 + s_1x_1 + s_2x_2)\cdot ax_2 \in \Sigma^{\leq 2}[\mathbf{x}] \qquad \text{(consecution)}
$$

$$
\Big\Updownarrow
$$

$$
\mathcal{F}(\mathbf{a},\mathbf{s}) = -\begin{pmatrix} -0.1\,a & 0 & 0.5\,a\,s_0 \\ 0 & 0 & 0.5\,(a\,s_1 - a) \\ 0.5\,a\,s_0 & 0.5\,(a\,s_1 - a) & a\,s_2 + 0.5\,a \end{pmatrix} \preceq 0
$$

BMIs

BMI Optimization

(SOS Cons)

$$\mathcal{F}_\iota(\mathbf{a},\mathbf{s}) \preceq 0, \quad \iota = 1, 2, \ldots, l. \tag{1}$$

BMI Optimization

$$\mathcal{F}_\iota(\mathbf{a},\mathbf{s}) \preceq 0, \quad \iota = 1,2,\ldots,l. \tag{1}$$

$$\Updownarrow$$

$$\begin{aligned} \underset{\lambda,\mathbf{a},\mathbf{s}}{\text{maximize}} \quad & \lambda \\ \text{subject to} \quad & \mathcal{B}_\iota(\lambda,\mathbf{a},\mathbf{s}) \mathrel{\widehat{=}} \mathcal{F}_\iota(\mathbf{a},\mathbf{s}) + \lambda \boldsymbol{I} \preceq 0, \quad \iota = 1,2,\ldots,l. \end{aligned} \tag{2}$$

BMI Optimization

$$\mathcal{F}_\iota(\mathbf{a},\mathbf{s}) \preceq 0, \quad \iota = 1,2,\ldots,l. \tag{1}$$

$\Updownarrow$ (1) is feasible iff $\lambda^* \geq 0$ in (2)

$$\begin{aligned} \underset{\lambda,\mathbf{a},\mathbf{s}}{\text{maximize}} \quad & \lambda \\ \text{subject to} \quad & \mathcal{B}_\iota(\lambda,\mathbf{a},\mathbf{s}) \mathrel{\widehat{=}} \mathcal{F}_\iota(\mathbf{a},\mathbf{s}) + \lambda I \preceq 0, \quad \iota = 1,2,\ldots,l. \end{aligned} \tag{2}$$

General BMI Optimization

$$\begin{aligned}
\underset{\mathbf{z}=(\mathbf{x},\mathbf{y})}{\text{maximize}} \quad & g(\mathbf{z}) \\
\text{subject to} \quad & \mathcal{B}(\mathbf{x},\mathbf{y}) \mathrel{\widehat{=}} \sum_{i=1}^{m}\sum_{j=1}^{n} x_i y_j F_{i,j} + \sum_{i=1}^{m} x_i H_i + \sum_{j=1}^{n} y_j G_j + F \preceq 0
\end{aligned}$$

General BMI Optimization

$$\begin{array}{ll} \underset{\mathbf{z}=(\mathbf{x},\mathbf{y})}{\text{maximize}} & g(\mathbf{z}) \\ \text{subject to} & \mathcal{B}(\mathbf{x},\mathbf{y}) \mathrel{\widehat{=}} \sum_{i=1}^{m}\sum_{j=1}^{n} x_i y_j F_{i,j} + \sum_{i=1}^{m} x_i H_i + \sum_{j=1}^{n} y_j G_j + F \preceq 0 \end{array}$$

$$\mathcal{B}(\mathbf{x},\mathbf{y}) = \begin{pmatrix} \mathbf{x}\otimes I \\ \mathbf{y}\otimes I \end{pmatrix}^{\mathsf{T}} \begin{pmatrix} 0 & \Gamma \\ \Gamma^{\mathsf{T}} & 0 \end{pmatrix} \begin{pmatrix} \mathbf{x}\otimes I \\ \mathbf{y}\otimes I \end{pmatrix} + \begin{pmatrix} \Omega_1 & \Omega_2 \end{pmatrix} \begin{pmatrix} \mathbf{x}\otimes I \\ \mathbf{y}\otimes I \end{pmatrix} + F$$

General BMI Optimization

$$\begin{array}{ll} \text{maximize} & g(\mathbf{z}) \\ \mathbf{z} = (\mathbf{x}, \mathbf{y}) & \\ \text{subject to} & \mathcal{B}(\mathbf{x}, \mathbf{y}) \mathrel{\widehat{=}} \sum_{i=1}^{m} \sum_{j=1}^{n} x_i y_j F_{i,j} + \sum_{i=1}^{m} x_i H_i + \sum_{j=1}^{n} y_j G_j + F \preceq 0 \end{array}$$

$$\underset{\substack{\shortparallel \\ B^+ - B^-}}{\mathcal{B}(\mathbf{x}, \mathbf{y})} = \begin{pmatrix} \mathbf{x} \otimes I \\ \mathbf{y} \otimes I \end{pmatrix}^{\mathsf{T}} \underbrace{\begin{pmatrix} 0 & \Gamma \\ \Gamma^{\mathsf{T}} & 0 \end{pmatrix}}_{M \,=\, M_1 - M_2} \begin{pmatrix} \mathbf{x} \otimes I \\ \mathbf{y} \otimes I \end{pmatrix} + \begin{pmatrix} \Omega_1 & \Omega_2 \end{pmatrix} \begin{pmatrix} \mathbf{x} \otimes I \\ \mathbf{y} \otimes I \end{pmatrix} + F$$

$\mathcal{B}(\mathbf{x}, \mathbf{y})$ is convex $\iff$ $M \succeq 0$.

Difference-of-Convex (DC) Decomposition

$$M = V^{\mathsf{T}} D V \qquad \text{(eigendecomposition)}$$

Difference-of-Convex (DC) Decomposition

$$\begin{aligned} M &= V^{\mathsf{T}} D V && \text{(eigendecomposition)} \\ &= \underbrace{V^{\mathsf{T}} D^{+} V}_{M_1 \succeq 0} - \underbrace{V^{\mathsf{T}} D^{-} V}_{M_2 \succeq 0} \end{aligned}$$

D^{+} $-D$

Difference-of-Convex (DC) Decomposition

$$\begin{aligned} M &= V^{\mathsf{T}} D V && \text{(eigendecomposition)} \\ &= \underbrace{V^{\mathsf{T}} D^{+} V}_{M_1 \succeq 0} - \underbrace{V^{\mathsf{T}} D^{-} V}_{M_2 \succeq 0} \end{aligned}$$

$$\mathcal{B}(\mathbf{x}, \mathbf{y}) = \underbrace{\mathcal{B}^{+}(\mathbf{x}, \mathbf{y})}_{\text{convex}} - \underbrace{\mathcal{B}^{-}(\mathbf{x}, \mathbf{y})}_{\text{convex}} \qquad \text{(DC decomposition)}$$

Difference-of-Convex (DC) Decomposition

$$\begin{aligned} M &= V^\mathsf{T} D V && \text{(eigendecomposition)} \\ &= \underbrace{V^\mathsf{T} D^+ V}_{M_1 \succeq 0} - \underbrace{V^\mathsf{T} D^- V}_{M_2 \succeq 0} \end{aligned}$$

$$\mathcal{B}(\mathbf{x}, \mathbf{y}) = \underbrace{\mathcal{B}^+(\mathbf{x}, \mathbf{y})}_{\text{convex}} - \underbrace{\mathcal{B}^-(\mathbf{x}, \mathbf{y})}_{\text{convex}} \qquad \text{(DC decomposition)}$$

Example (running)

The decomposition of $\mathcal{B}(\lambda, \mathbf{a}, \mathbf{s})$ for consecution, for instance, gives

$$\mathcal{B}^+(\lambda, \mathbf{a}, \mathbf{s}) =$$

$$\frac{1}{8}\begin{pmatrix} 8\lambda+0.08a+a^2+0.408s_0^2 & 0.408s_0 s_1 & -2as_0+0.816s_0 s_2 \\ 0.408s_0 s_1 & 8\lambda+a^2+0.408s_1^2 & 4a-2as_1+0.816s_1 s_2 \\ -2as_0+0.816s_0 s_2 & 4a-2as_1+0.816s_1 s_2 & 8\lambda-4a+2.449a^2-4as_2+s_0^2+s_1^2+1.632s_2^2 \end{pmatrix}$$

$$\mathcal{B}^-(\lambda, \mathbf{a}, \mathbf{s}) =$$

$$\frac{1}{8}\begin{pmatrix} a^2+0.408s_0^2 & 0.408s_0 s_1 & 2as_0+0.816s_0 s_2 \\ 0.408s_0 s_1 & a^2+0.408s_1^2 & 2as_1+0.816s_1 s_2 \\ 2as_0+0.816s_0 s_2 & 2as_1+0.816s_1 s_2 & 2.449a^2+4as_2+s_0^2+s_1^2+1.632s_2^2 \end{pmatrix}.$$

Reducing to a Series of Convex Programs

Linearize the "concave part" $-\mathcal{B}^{-}(\mathbf{x}, \mathbf{y})$ around a feasible solution $\mathbf{z}^k$:

$$\underbrace{\mathcal{B}^{+}(\mathbf{z}) - \mathcal{B}^{-}\left(\mathbf{z}^k\right) - \mathcal{D}\mathcal{B}^{-}\left(\mathbf{z}^k\right)\left(\mathbf{z} - \mathbf{z}^k\right)}_{\text{convex}} \preceq 0 \qquad \text{(QMIs)}$$

Reducing to a Series of Convex Programs

Linearize the "concave part" $-\mathcal{B}^{-}(\mathbf{x}, \mathbf{y})$ around a feasible solution $\mathbf{z}^k$:

$$\underbrace{\mathcal{B}^{+}(\mathbf{z}) - \mathcal{B}^{-}\left(\mathbf{z}^k\right) - \mathcal{D}\mathcal{B}^{-}\left(\mathbf{z}^k\right)\left(\mathbf{z} - \mathbf{z}^k\right)}_{\text{convex}} \preceq 0 \qquad \text{(QMIs)}$$

$$\Updownarrow \text{ Schur complement}$$

$z^{k+1} = z^{*,k}$

$$\underbrace{\begin{pmatrix} -I & N(\mathbf{z} \otimes I) \\ (\mathbf{z} \otimes I)^{\mathsf{T}} N^{\mathsf{T}} & -\mathcal{B}^{-}\left(\mathbf{z}^k\right) - \mathcal{D}\mathcal{B}^{-}\left(\mathbf{z}^k\right)\left(\mathbf{z} - \mathbf{z}^k\right) + \Omega(\mathbf{z} \otimes I) + F \end{pmatrix}}_{\text{convex}} \preceq 0 \qquad \text{(LMIs)}$$

Reducing to a Series of Convex Programs

Linearize the "concave part" $-\mathcal{B}^-(\mathbf{x}, \mathbf{y})$ around a feasible solution $\mathbf{z}^k$:

$$\underbrace{\mathcal{B}^+(\mathbf{z}) - \mathcal{B}^-\left(\mathbf{z}^k\right) - \mathcal{D}\mathcal{B}^-\left(\mathbf{z}^k\right)\left(\mathbf{z} - \mathbf{z}^k\right)}_{\text{convex}} \preceq 0 \qquad \text{(QMIs)}$$

$$\Big\Updownarrow \text{ Schur complement}$$

$$\underbrace{\begin{pmatrix} -I & N(\mathbf{z} \otimes I) \\ (\mathbf{z} \otimes I)^\mathsf{T} N^\mathsf{T} & -\mathcal{B}^-\left(\mathbf{z}^k\right) - \mathcal{D}\mathcal{B}^-\left(\mathbf{z}^k\right)\left(\mathbf{z} - \mathbf{z}^k\right) + \Omega(\mathbf{z} \otimes I) + F \end{pmatrix}}_{\text{convex}} \preceq 0 \qquad \text{(LMIs)}$$

z^0

⇒ DCP : an iterative procedure ($\mathbf{z}^{k+1} = \mathbf{z}^{*,k}$) that solves a series of convex programs.

Finding the Initial Solution

$$\begin{array}{ll} \underset{\lambda,\mathbf{a}}{\text{maximize}} & \lambda \\ \text{subject to} & \mathcal{F}_\iota(\mathbf{a},\mathbf{s})\big|_{\mathbf{s}=(c_\iota,0,\ldots,0)} + \lambda I \preceq 0, \quad \iota = 1,2,\ldots,l. \end{array} \Bigg\} \text{LMI}$$

Here, $c_\iota \in \mathbb{R}_0^+$ encodes a non-negative constant multiplier polynomial :

$$\begin{cases} c_\iota = 0: & \text{original BC cond. [Prajna \& Jadbabaie, 2004]} \\ c_\iota > 0: & \text{exponential-type BC cond. [Kong } et\ al.\text{, 2013]} \end{cases}$$

Finding the Initial Solution

$$\begin{aligned} \underset{\lambda,\, \mathbf{a}}{\text{maximize}} \quad & \lambda \\ \text{subject to} \quad & \mathcal{F}_\iota(\mathbf{a}, \mathbf{s})\big|_{\mathbf{s}=(c_\iota, 0, \ldots, 0)} + \lambda I \preceq 0, \quad \iota = 1, 2, \ldots, l. \end{aligned}$$

Here, $c_\iota \in \mathbb{R}_0^+$ encodes a non-negative constant multiplier polynomial :

$$\begin{cases} c_\iota = 0 : & \text{original BC cond. [Prajna \& Jadbabaie, 2004]} \\ c_\iota > 0 : & \text{exponential-type BC cond. [Kong } et\ al.\text{, 2013]} \end{cases}$$

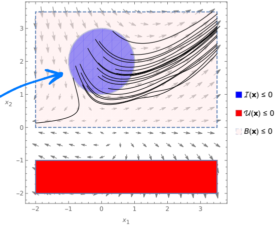
⇛ This LMI optimization always admits a strictly feasible solution $(\lambda, \mathbf{a})$ which induces also a strictly feasible solution $(\lambda, \mathbf{a}, (c_\iota, 0, \ldots, 0))$ to the original BMI optimization.

Properties of DCP

Example (running)

Our iterative procedure starts with a strictly feasible initial solution $\mathbf{z}^0$ and terminates with $\boxed{\lambda^2 \geq 0}$ (subject to numerical round-off) and $a^2 = -0.00363421$, yielding the barrier certificate

$$B\left(\mathbf{a}^2, \mathbf{x}\right) \;=\; -0.00363421 x_2 \leq 0.$$

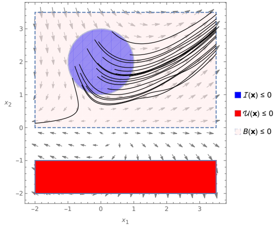


Properties of DCP

Example (running)

Our iterative procedure starts with a strictly feasible initial solution $\mathbf{z}^0$ and terminates with $\lambda^2 \geq 0$ (subject to numerical round-off) and $a^2 = -0.00363421$, yielding the barrier certificate

$$B\left(\mathbf{a}^2, \mathbf{x}\right) = -0.00363421 x_2 \leq 0.$$



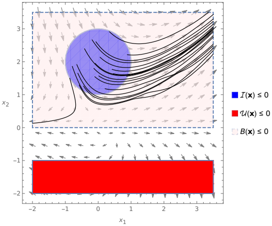
⇒ Soundness : every $\mathbf{z}^i$ is a feasible solution to the original BMI optimization;

Properties of DCP

Example (running)

Our iterative procedure starts with a strictly feasible initial solution $\mathbf{z}^0$ and terminates with $\lambda^2 \geq 0$ (subject to numerical round-off) and $a^2 = -0.00363421$, yielding the barrier certificate

$$B\left(\mathbf{a}^2, \mathbf{x}\right) \ = \ -0.00363421 x_2 \leq 0.$$



⇒ Soundness : every $\mathbf{z}^i$ is a feasible solution to the original BMI optimization;

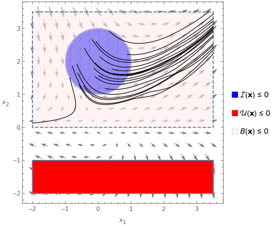
⇒ Convergence : $\{\mathbf{z}^i\}_{i\in\mathbb{N}}$ converges to a KKT point (local optimum);

Properties of DCP

Example (running)

Our iterative procedure starts with a strictly feasible initial solution $\mathbf{z}^0$ and terminates with $\lambda^2 \geq 0$ (subject to numerical round-off) and $a^2 = -0.00363421$, yielding the barrier certificate

$$B\left(\mathbf{a}^2, \mathbf{x}\right) \;=\; -0.00363421 x_2 \leq 0.$$

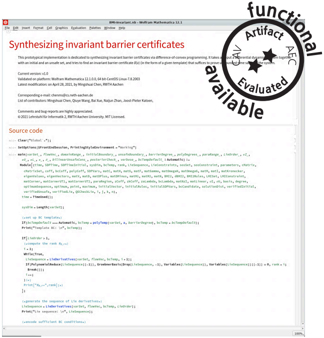


- ⇒ Soundness : every $\mathbf{z}^i$ is a feasible solution to the original BMI optimization;
- ⇒ Convergence : $\{\mathbf{z}^i\}_{i\in\mathbb{N}}$ converges to a KKT point (local optimum);
- ⇒ Weak completeness (via branch-and-bound) : an invariant BC is guaranteed to be found (under mild assumptions) whenever there exists an inductive invariant (in the form of the given template).

Prototypical Implementation

BMI-DC : open-source in Wolfram Mathematica :

- CSDP : COIN semidefinite programming library;
- Reduce & Z3 : posterior check of candidate BCs.



In comparison with off-the-shelf solvers in Matlab :

- PENLAB : solving BMIs (with no guarantee on convergence) [Fiala *et al.*, 2013];
- SOSTOOLS : solving LMIs as per Prajna and Jadbabaie's original BC condition [Papachristodoulou *et al.*, 2013].

Empirical Results

Table – Empirical results on benchmark examples (time in seconds)

Example name	n_{sys}	d_{flow}	d_{BC}	BMI-DC			PENLAB		SOSTOOLS	
				#iter.	time	verified	time	verified	time	verified
overview	2	2	1	2	**0.03**	✓	0.31	✓	0.07	✓
contrived	2	1	2	0	**0.01**	✓	0.48	✓	0.75	✓
lie-der	2	2	1	0	**0.01**	✓	0.22	✓	0.04	✓
lorenz	3	2	2	8	**2.37**	✓	75.11	✗	1.47	✗
lti-stable	2	1	2	0	**0.01**	✓	0.23	✓	0.14	✓
lotka-volterra	3	2	1	3	**0.07**	✓	0.36	✓	0.21	✓
clock	2	3	1	0	**0.01**	✓	0.88	✗	0.18	✗
lyapunov	3	3	2	4	1.25	✓	56.98	✗	**0.35**	✓
arch1	2	5	2	0	**0.01**	✓	33.76	✗	0.31	✓
arch2	2	2	2	5	**0.37**	✓	0.38	✗	0.17	✗
arch3	2	3	2	1	**0.07**	✓	0.54	✓	0.18	✓
arch4	2	2	1	2	0.09	✓	0.49	✗	**0.06**	✓
barr-cert1	2	3	2	12	**0.85**	✓	2.53	✗	0.09	✗
barr-cert2	2	2	2	6	**1.57**	✓	1.16	✗	0.15	✓
barr-cert3	2	2	1	0	**0.01**	✓	0.20	✓	0.11	✗
barr-cert4	2	3	2	13	**0.96**	✓	0.89	✗	0.23	✗
fitzhugh-nagumo	2	3	2	2	**0.16**	✓	1.24	✓	0.25	✗
stabilization	3	2	2	9	2.88	✓	55.22	✓	**0.11**	✓
lie-high-order	2	1	2	32	**4.12**	✓	1.56	✗	0.25	✗
raychaudhuri	4	2	2	34	**9.51**	✓	33.64	✗	0.14	✗
focus	2	1	4	100	54.89	✗	0.95	✗	0.48	✗
sys-bio1	7	2	2	2	73.22	?	101.9 5	?	1.35	?
sys-bio2	9	2	1	1	1.03	?	15.54	?	0.16	?
quadcopter	12	1	1	0	0.03	?	65.42	?	0.36	?

Empirical Results

Table – Empirical results on benchmark examples (time in seconds)

Example name	n_{sys}	d_{flow}	d_{BC}	BMI-DC			PENLAB		SOSTOOLS	
				#iter.	time	verified	time	verified	time	verified
overview	2	2	1	2	**0.03**	✓	0.31	✓	0.07	✓
contrived	2	1	2	0	**0.01**	✓	0.48	✓	0.75	✓
lie-der	2	2	1	0	**0.01**	✓	0.22	✓	0.04	✓
lorenz	3	2	2	8	**2.37**	✓	75.11	✗	1.47	✗
lti-stable	2	1	2	0	**0.01**	✓	0.23	✓	0.14	✓
lotka-volterra	3	2	1	3	**0.07**	✓	0.36	✓	0.21	✓
clock	2	3	1	0	**0.01**	✓	0.88	✗	0.18	✗
lyapunov	3	3	2	4	1.25	✓	56.98	✗	**0.35**	✓
arch1	2	5	2	0	**0.01**	✓	33.76	✗	0.31	✓
arch2	2	2	2	5	**0.37**	✓	0.38	✗	0.17	✗
arch3	2	3	2	1	**0.07**	✓	0.54	✓	0.18	✓
arch4	2	2	1	2	0.09	✓	0.49	✗	**0.06**	✓
barr-cert1	2	3	2	12	**0.85**	✓	2.53	✗	0.09	✗
barr-cert2	2	2	2	6	**1.57**	✓	1.16	✗	0.15	✓
barr-cert3	2	2	1	0	**0.01**	✓	0.20	✓	0.11	✗
barr-cert4	2	3	2	13	**0.96**	✓	0.89	✗	0.23	✗
fitzhugh-nagumo	2	3	2	2	**0.16**	✓	1.24	✓	0.25	✗
stabilization	3	2	2	9	2.88	✓	55.22	✓	**0.11**	✓
lie-high-order	2	1	2	32	**4.12**	✓	1.56	✗	0.25	✗
raychaudhuri	4	2	2	34	**9.51**	✓	33.64	✗	0.14	✗
focus	2	1	4	100	54.89	✗	0.95	✗	0.48	✗
sys-bio1	7	2	2	2	73.22	?	101.9 5	?	1.35	?
sys-bio2	9	2	1	1	1.03	?	15.54	?	0.16	?
quadcopter	12	1	1	0	0.03	?	65.42	?	0.36	?

Empirical Results

Table – Empirical results on benchmark examples (time in seconds)

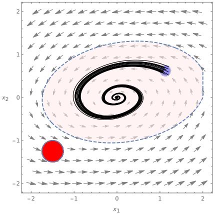
Example name	n_{sys}	d_{flow}	d_{BC}	BMI-DC			PENLAB		SOSTOOLS	
				#iter.	time	verified	time	verified	time	verified
overview	2	2	1	2	**0.03**	✓	0.31	✓	0.07	✓
contrived	2	1	2	0	**0.01**	✓	0.48	✓	0.75	✓
lie-der	2	2	1	0	**0.01**	✓	0.22	✓	0.04	✓
lorenz	3	2	2	8	**2.37**	✓	75.11	✗	1.47	✗
lti-stable	2	1	2	0	**0.01**	✓	0.23	✓	0.14	✓
lotka-volterra	3	2	1	3	**0.07**	✓	0.36	✓	0.21	✓
clock	2	3	1	0	**0.01**	✓	0.88	✗	0.18	✗
lyapunov	3	3	2	4	1.25	✓	56.98	✗	**0.35**	✓
arch1	2	5	2	0	**0.01**	✓	33.76	✗	0.31	✓
arch2	2	2	2	5	**0.37**	✓	0.38	✗	0.17	✗
arch3	2	3	2	1	**0.07**	✓	0.54	✓	0.18	✓
arch4	2	2	1	2	0.09	✓	0.49	✗	**0.06**	✓
barr-cert1	2	3	2	12	**0.85**	✓	2.53	✗	0.09	✗
barr-cert2	2	2	2	6	**1.57**	✓	1.16	✗	0.15	✓
barr-cert3	2	2	1	0	**0.01**	✓	0.20	✓	0.11	✗
barr-cert4	2	3	2	13	**0.96**	✓	0.89	✗	0.23	✗
fitzhugh-nagumo	2	3	2	2	**0.16**	✓	1.24	✓	0.25	✗
stabilization	3	2	2	9	2.88	✓	55.22	✓	**0.11**	✓
lie-high-order	2	1	2	32	**4.12**	✓	1.56	✗	0.25	✗
raychaudhuri	4	2	2	34	**9.51**	✓	33.64	✗	0.14	✗
focus	2	1	4	100	54.89	✗	0.95	✗	0.48	✗
sys-bio1	7	2	2	2	73.22	?	101.9 5	?	1.35	?
sys-bio2	9	2	1	1	1.03	?	15.54	?	0.16	?
quadcopter	12	1	1	0	0.03	?	65.42	?	0.36	?

Empirical Results

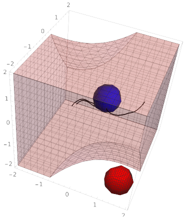
Table – Empirical results on benchmark examples (time in seconds)

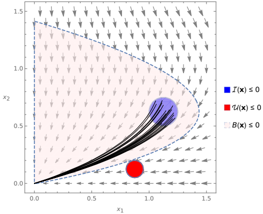
Example name	n_{sys}	d_{flow}	d_{BC}	BMI-DC			PENLAB		SOSTOOLS	
				#iter.	time	verified	time	verified	time	verified
overview	2	2	1	2	**0.03**	✓	0.31	✓	0.07	✓
contrived	2	1	2	0	**0.01**	✓	0.48	✓	0.75	✓
lie-der	2	2	1	0	**0.01**	✓	0.22	✓	0.04	✓
lorenz	3	2	2	8	**2.37**	✓	75.11	✗	1.47	✗
lti-stable	2	1	2	0	**0.01**	✓	0.23	✓	0.14	✓
lotka-volterra	3	2	1	3	**0.07**	✓	0.36	✓	0.21	✓
clock	2	3	1	0	**0.01**	✓	0.88	✗	0.18	✗
lyapunov	3	3	2	4	1.25	✓	56.98	✗	**0.35**	✓
arch1	2	5	2	0	**0.01**	✓	33.76	✗	0.31	✓
arch2	2	2	2	5	**0.37**	✓	0.38	✗	0.17	✗
arch3	2	3	2	1	**0.07**	✓	0.54	✓	0.18	✓
arch4	2	2	1	2	0.09	✓	0.49	✗	**0.06**	✓
barr-cert1	2	3	2	12	**0.85**	✓	2.53	✗	0.09	✗
barr-cert2	2	2	2	6	**1.57**	✓	1.16	✗	0.15	✓
barr-cert3	2	2	1	0	**0.01**	✓	0.20	✓	0.11	✗
barr-cert4	2	3	2	13	**0.96**	✓	0.89	✗	0.23	✗
fitzhugh-nagumo	2	3	2	2	**0.16**	✓	1.24	✓	0.25	✗
stabilization	3	2	2	9	2.88	✓	55.22	✓	**0.11**	✓
lie-high-order	2	1	2	32	**4.12**	✓	1.56	✗	0.25	✗
raychaudhuri	4	2	2	34	**9.51**	✓	33.64	✗	0.14	✗
focus	2	1	4	100	54.89	✗	0.95	✗	0.48	✗
sys-bio1	7	2	2	2	73.22	?	101.95	?	1.35	?
sys-bio2	9	2	1	1	1.03	?	15.54	?	0.16	?
quadcopter	12	1	1	0	0.03	?	65.42	?	0.36	?

Phase Portraits



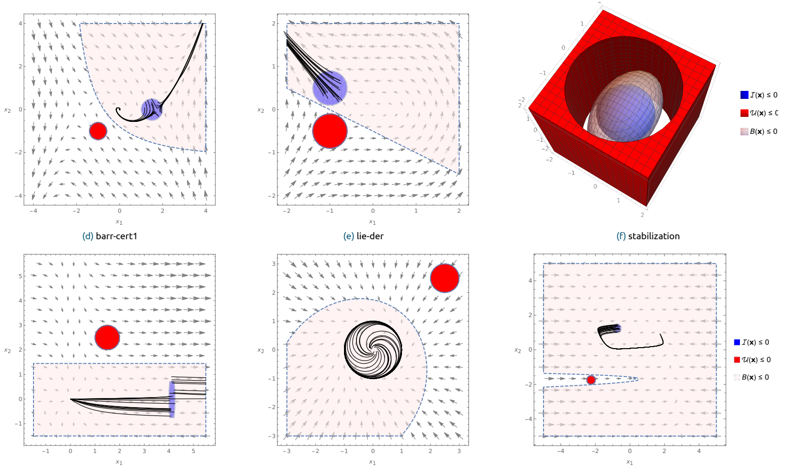
(a) lti-stable


(b) lyapunov



(c) barr-cert2

Phase Portraits



(d) barr-cert1 (e) lie-der (f) stabilization

(g) clock (h) arch3 (i) fitzhugh-nagumo

Summary

"Find a BC condition that is as weak as possible while admitting efficient synthesis."

⇒ Q. Wang, M. Chen, B. Xue, N. Zhan, J.-P. Katoen :
Synthesizing Invariant Barrier Certificates via Difference-of-Convex Programming.

functional
Artifact
CAV AEC
Evaluated
available

Summary

"Find a BC condition that is as weak as possible while admitting efficient synthesis."

- Invariant BC condition : the weakest possible condition to attain inductive invariance ;

⇒ Q. Wang, M. Chen, B. Xue, N. Zhan, J.-P. Katoen : *Synthesizing Invariant Barrier Certificates via Difference-of-Convex Programming.*

Summary

"Find a BC condition that is as weak as possible while admitting efficient synthesis."

- Invariant BC condition : the weakest possible condition to attain inductive invariance ;
- Discharging the invariant BC condition can be encoded as solving a BMI optimization ;

⇛ Q. Wang, M. Chen, B. Xue, N. Zhan, J.-P. Katoen :
Synthesizing Invariant Barrier Certificates via Difference-of-Convex Programming.

functional Artifact CAV AEC Evaluated available

Summary

"Find a BC condition that is as weak as possible while admitting efficient synthesis."

- Invariant BC condition : the weakest possible condition to attain inductive invariance ;
- Discharging the invariant BC condition can be encoded as solving a BMI optimization ;
- DCP : approaches a local optimum via solving a series of convex problems ;

⇒ Q. Wang, M. Chen, B. Xue, N. Zhan, J.-P. Katoen :
Synthesizing Invariant Barrier Certificates via Difference-of-Convex Programming.

Summary

"Find a BC condition that is as weak as possible while admitting efficient synthesis."

- Invariant BC condition : the weakest possible condition to attain inductive invariance;
- Discharging the invariant BC condition can be encoded as solving a BMI optimization;
- DCP : approaches a local optimum via solving a series of convex problems;
- branch-and-bound : searches for the global optimum in a divide-and-conquer fashion, yielding a weak completeness result.

⇒ Q. Wang, M. Chen, B. Xue, N. Zhan, J.-P. Katoen :
Synthesizing Invariant Barrier Certificates via Difference-of-Convex Programming.