Towards a viability kernel computation in higher dimensions with interval analysis

Réunion MRIS

Olivier Mullier Alexandre Chapoutot Julien Alexandre dit Sandretto

ENSTA Paristech
olivier.mullier@polytechnique.edu

24 novembre 2016

Motivation

Viability kernel computation based on interval methods (D. Monnet et al.)

- Viability theory,
- Interval analysis,
- Validated numerical integration methods.

Drawbacks

Not suitable for state dimension greater than 2

- Time complexity of the method.

Motivation

Viability kernel computation based on interval methods (D. Monnet et al.)

- Viability theory,
- Interval analysis,
- Validated numerical integration methods.

Drawbacks

Not suitable for state dimension greater than 2

- Time complexity of the method.

Goal of this work

- Extend to problems of greater dimension,
- Inherit the last advances in validated integration methods to reduce time complexity.

Context

Let a dynamical system $\mathcal{S}$ be described by

$$(\mathcal{S}) \begin{cases} \dot{y} = f(y(t), u(t)) \\ y(0) \in K \\ u(t) \in \mathcal{U}(t) \end{cases}$$

- $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$,
- K: set of possible values for $y(t)$,
- $\mathcal{U}(t)$: a function space from $\mathbb{R}$ to $\mathbb{R}^m$.

$y(t; y_0, u)$ denotes the solution of $\mathcal{S}$ for $y(0) = y_0$ at time t for a particular u.

Context

Let a dynamical system $\mathcal{S}$ be described by

$$(\mathcal{S}) \begin{cases} \dot{y} = f(y(t), u(t)) \\ y(0) \in K \\ u(t) \in \mathcal{U}(t) \end{cases}$$

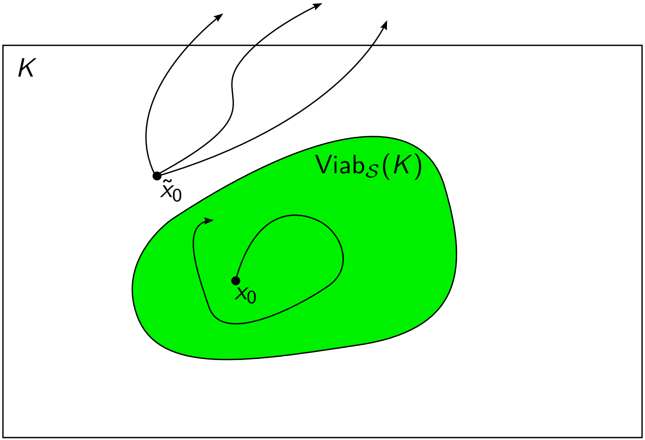
- $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$,
- K: set of possible values for $y(t)$,
- $\mathcal{U}(t)$: a function space from $\mathbb{R}$ to $\mathbb{R}^m$.

$y(t; y_0, u)$ denotes the solution of $\mathcal{S}$ for $y(0) = y_0$ at time t for a particular u.

Goal

Compute the set of initial conditions inside K for which there always exists a control allowing the system $\mathcal{S}$ to remain in K.

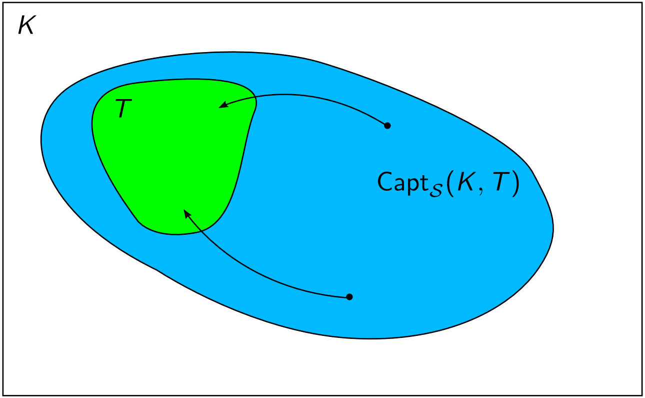
Viability kernel



$$\text{Viab}_\mathcal{S}(K) = \{y_0 \in K \mid (\forall \tilde{t} > 0)(\exists \tilde{u} : t \in [0, \tilde{t}] \to u(t))(y(\tilde{t}, y_0, \tilde{u}) \in K)\}$$

Capture basin

Let $T \subseteq K$ be a target,



$$\text{Capt}_{\mathcal{S}}(K,T) = \left\{ y_0 \in K \;\middle|\; \begin{array}{l} (\exists \tilde{t} \geqslant 0)(\exists \tilde{u} : t \in [0,\tilde{t}] \to u(t))(y(\tilde{t}, y_0, \tilde{u}) \in T) \\ \wedge (\forall t \in [0,\tilde{t}])(y(t, y_0, \tilde{u}) \in K) \end{array} \right\}$$

Computing the viability kernel

We define the capture basin in a time horizon t_{end}

$$\text{Capt}_{\mathcal{S}}^{t_{\text{end}}}(K,T) = \left\{ y_0 \in K \;\middle|\; \begin{array}{r} (\exists \tilde{t} \in \mathbf{[0, t_{end}]})(\exists \tilde{u} : t \in [0,\tilde{t}] \to u(t)) \\ (y(\tilde{t}, y_0, \tilde{u}) \in T) \\ \wedge (\forall t \in [0,\tilde{t}])(y(t, y_0, \tilde{u}) \in K) \end{array} \right\}$$

Property

$$T \subseteq \text{Viab}_{\mathcal{S}}(K) \Rightarrow \text{Capt}_{\mathcal{S}}^{t_{\text{end}}}(K,T) \subseteq \text{Capt}_{\mathcal{S}}(K,T) \subseteq \text{Viab}_{\mathcal{S}}(K)$$

Algorithm (sketch)

1. Compute $V_0^- \subseteq \text{Viab}_{\mathcal{S}}(K)$;
2. Enlarge V_0^- by computing $V_{i+1}^- = \text{Capt}_{\mathcal{S}}^{t_{\text{end}}}(K, V_i^-)$ until $V_{i+1}^- \cap V_i^- = V_i^-$

Outline

Outline

1. Computation of $V_0^- \subseteq \text{Viab}_{\mathcal{S}}(K)$
2. Iteration of $V_{i+1}^- = \text{Capt}_{\mathcal{S}}^{t_{\text{end}}}(K, V_i^-)$
3. Computation of $K \setminus \text{Viab}_{\mathcal{S}}(K)$
4. Examples
5. Conclusion

Stability analysis: domain of attraction

Let the system $\mathcal{S}'$ be defined by $\dot{y} = f(y)$ and $y(t; y_0)$ the solution of the system at time t when $y(0) = y_0$.

Definition (Equilibrium state)

A state $\tilde{y}$ where $f(\tilde{y}) = 0$ is called an *equilibrium state*.

Definition (Domain of attraction)

The set

$$\{y_0 \mid \lim_{t \to \infty} y(t; y_0) = \tilde{y}\}$$

with $\tilde{y}$ an equilibrium state, is a domain of attraction.

Stability analysis: domain of attraction

Let the system $\mathcal{S}'$ be defined by $\dot{y} = f(y)$ and $y(t; y_0)$ the solution of the system at time t when $y(0) = y_0$.

Definition (Equilibrium state)

A state $\tilde{y}$ where $f(\tilde{y}) = 0$ is called an *equilibrium state*.

Definition (Domain of attraction)

The set

$$\{y_0 \mid \lim_{t \to \infty} y(t; y_0) = \tilde{y}\}$$

with $\tilde{y}$ an equilibrium state, is a domain of attraction.

Viability

If there exists $u \in \mathcal{U}$ such that an equilibrium state $\tilde{y} \in K$ exists $(f(\tilde{y}, u) = 0)$, then its corresponding domain of attraction belongs to $\text{Viab}_{\mathcal{S}}(K)$.

Stability analysis: domain of attraction

Definition (Lyapunov function)

Let $\mathcal{M} \subset \mathbb{R}^n$ and $y \in \text{int}(\mathcal{M})$. A differentiable real valued function $L : \mathbb{R}^n \to \mathbb{R}$ is a Lyapunov function for the dynamical system $\dot{y} = f(y)$ if

1. $L(y) = 0 \Leftrightarrow y = \tilde{y}$,
2. $\forall y \in \mathcal{M} \setminus \{\tilde{y}\}$, $L(y) > 0$,
3. $\forall y \in \mathcal{M} \setminus \{\tilde{y}\}$, $\langle DL(y), f(y) \rangle < 0$.

Theorem

Let $L(y)$ be a Lyapunov function for the time continuous system $\dot{y} = f(y)$ with $f(\tilde{y}) = 0$. There exists c a positive real constant such that the domain

$$S = \{y \in \mathbb{R}^n \mid 0 < L(y) < c, \langle DL(y), f(y) \rangle < 0\}$$

belongs to the domain of attraction of $\tilde{y}$

Building a Lyapunov function (quadratic form)

From $\tilde{y}$ an equilibrium state

- Consider the linearized system $\dot{y} = Ay(t)$ with $A = (\frac{\partial f}{\partial y_i})_i(\tilde{y})$ and $u = 0$.
- Find the symmetric positive definite matrix P such that $A^T P + PA = -I$. It is equivalent to the problem of finding $x = \left(x_1, \ldots, x_{\frac{n(n+1)}{2}}\right) \in \mathbb{R}^{\frac{n(n+1)}{2}}$ such that $Mx = b$ with x_i from P (which is symmetric):

$$P = \begin{pmatrix} x_1 & x_2 & \cdots & x_n \\ x_2 & x_{n+1} & \cdots & x_{2n-1} \\ \vdots & & & \vdots \\ x_n & x_{2n-1} & & x_{\frac{n(n+1)}{2}} \end{pmatrix}$$

  and the coefficients of M and b deduced from $A^T P + PA = -I$. The coefficients x_i are then computed using $x = M^{-1} b$ (M must be invertible).
- $L(y) = (y - \tilde{y})^T P (y - \tilde{y})$ is a candidate to be a Lyapunov function.

Building a Lyapunov function (quadratic form)

$$L(y) = (y - \tilde{y})^T P (y - \tilde{y})$$

L is a Lyapunov function if

- P must be a positive definite matrix $\Rightarrow$ Sylvester criteria,
- The criteria (1-3) remains true:

$$\left.\begin{array}{l} L(y) = 0 \Leftrightarrow y = y^* \\ \forall y \in \mathcal{M} \setminus \{y^*\},\ L(y) > 0 \end{array}\right\} \text{true by construction of } L$$

$$\forall y \in \mathcal{M} \setminus \{y^*\},\ \langle DL(y), f(y) \rangle < 0 \ ?$$

Proof

Find c such that for each y inside the ellipsoid $\{y \in \mathbb{R}^n \mid 0 < L(y) < c\}$, $\langle DL(y), f(y) \rangle < 0$ remains true.

Is the ellipsoid inside the attraction domain

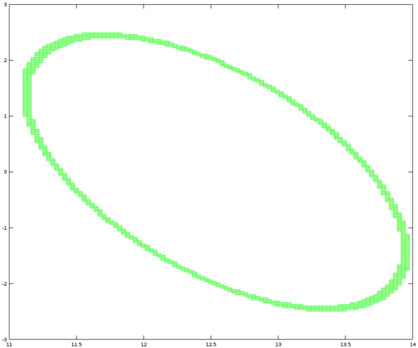


Figure: Frontier of $L(y) < c$

Using interval analysis,

1. We fix $u = 0$,
2. We choose $c > 0$,
3. We pave the frontier of $L(y) < c$ using boxes,
4. We check that for all box $[y]$ in this paving,

$$\langle DL([y]), f([y], 0)\rangle \subset [-\infty, 0]$$

- if this is true, the ellipsoid belongs to the domain of attraction,
- if not, we reduce c and restart from 2.

Is the ellipsoid inside the attraction domain

- If the test fails, we start over using full state feedback.
- For each box $[y]$ in the paving, $\hat{y} \in [y]$ and $\tilde{y}$ the current equilibrium state, we compute

$$u = -K(\hat{y} - \tilde{y})$$

  with K the gain matrix computed using the Faddeev - Leverrier algorithm.
- If $u \in \mathcal{U}$, the test is now

$$\langle DL([y]), f([y], u)\rangle \subset [-\infty, 0]$$

Result

The result is then a paving of the interior of the ellipsoids proven to be included in their domain of attraction.

Outline

Validated numerical integration

Definition (IVP-ode)

An IVP-ODE is defined by

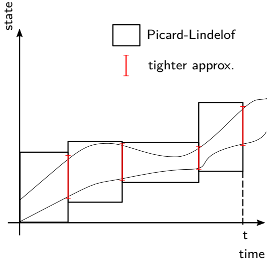
$$\begin{cases} \dot{y} = f(t, y) \\ y(0) \in \mathcal{Y}_0 \subseteq \mathbb{R}^n, \ t \in [0, t_{\text{end}}] \end{cases}.$$

Goal is to compute $y(t; \mathcal{Y}_0) = \{y(t; y_0) \mid y_0 \in \mathcal{Y}_0\}$.

Phase 1 a priori enclosure of

$$\{y(t_k; y_i) \mid t_k \in [t_i, t_{i+1}], y_i \in [y_i]\}$$

Phase 2 tight enclosure of $[y_{i+1}]$ at time t_{i+1}.



DynIBEX

Validated simulation with Runge-Kutta

- Proof of existence and uniqueness of solution for ODEs and DAEs,
- Local truncation error computation for any Runge-Kutta method (implicit or explicit),
- Combined with contractors (HC4).

Verification of temporal constraints

- Stayed in $\mathcal{A}$ until $\tilde{t} < t_{\text{end}}$:

$$\forall t \in [0, \tilde{t}], \ \{y(t; y_0) \mid y_0 \in [y_0]\} \subseteq \text{int}(\mathcal{A})$$

- Included in $\mathcal{A}$ inside $[0, t_{\text{end}}]$:

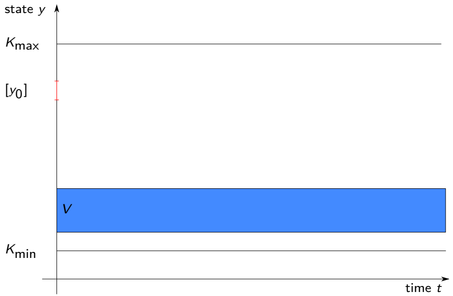
$$\exists t \in [0, t_{\text{end}}], \ \{y(t; y_0) \mid y_0 \in [y_0]\} \subseteq \text{int}(\mathcal{A}).$$

Capture basin $\text{Capt}_{\mathcal{S}}^{t_{\text{end}}}(K, T)$: stayed in K until it is included in T.

Capture basin: Algorithm

input :
V the current inner approximation of the viability kernel;
$[y_0] \in K \setminus V$;
$U_p = (u_1, u_2, \ldots, u_p)$ a sampling of $\mathcal{U}$;

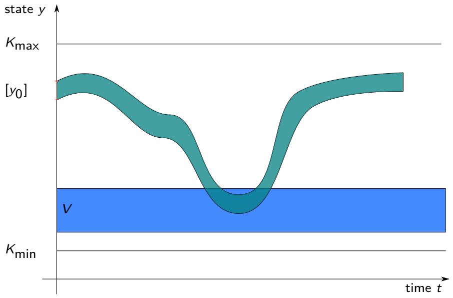
for $u_i \in U_p$ **do**
  compute $[y_{t_{\mathrm{end}}}] \supseteq \{y(t; y_0, u_i) \mid t \in [0, t_{\mathrm{end}}],\ y_0 \in [y_0]\}$;
  if $\exists t_i \in [0, t_{end}]$ *such that* $[y_{t_i}] \subseteq V$ **then**
    if $\forall t \in [0, t_i], [y_t] \subseteq K$ **then**
      $V := V \cup [y_0]$;
    end
  end
end



Capture basin: Algorithm

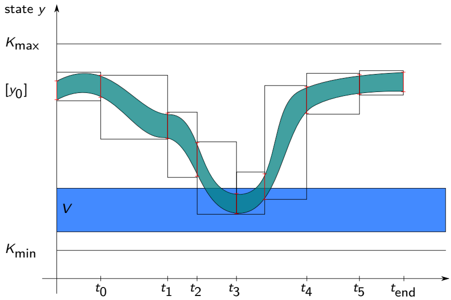
input :
V the current inner approximation of the viability kernel;
$[y_0] \in K \setminus V$;
$U_p = (u_1, u_2, \ldots, u_p)$ a sampling of $\mathcal{U}$;

for $u_i \in U_p$ **do**
  compute $[y_{t_{\text{end}}}] \supseteq \{y(t; y_0, u_i) \mid t \in [0, t_{\text{end}}],\ y_0 \in [y_0]\}$;
  if $\exists t_i \in [0, t_{end}]$ *such that* $[y_{t_i}] \subseteq V$ **then**
    if $\forall t \in [0, t_i], [y_t] \subseteq K$ **then**
      $V := V \cup [y_0]$;
    end
  end
end



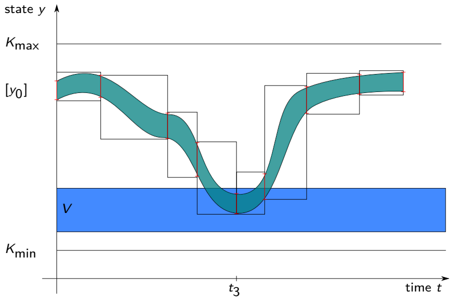
Capture basin: Algorithm

input :
V the current inner approximation of the viability kernel;
$[y_0] \in K \setminus V$;
$U_p = (u_1, u_2, \ldots, u_p)$ a sampling of $\mathcal{U}$;
for $u_i \in U_p$ **do**
  compute $[y_{t_{end}}] \supseteq \{y(t; y_0, u_i) \mid t \in [0, t_{end}],\ y_0 \in [y_0]\}$;
  if $\exists t_i \in [0, t_{end}]$ *such that* $[y_{t_i}] \subseteq V$ **then**
    if $\forall t \in [0, t_i], [y_t] \subseteq K$ **then**
      $V := V \cup [y_0]$;
    end
  end
end



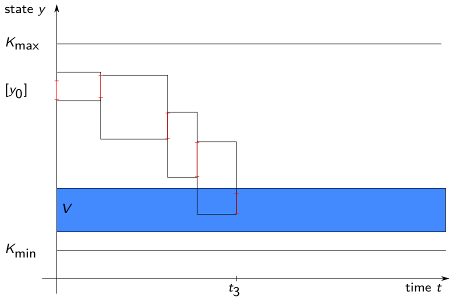
Capture basin: Algorithm

input :
V the current inner approximation of the viability kernel;
$[y_0] \in K \setminus V$;
$U_p = (u_1, u_2, \ldots, u_p)$ a sampling of $\mathcal{U}$;
for $u_i \in U_p$ **do**
  compute $[y_{t_{\mathrm{end}}}] \supseteq \{y(t; y_0, u_i) \mid t \in [0, t_{\mathrm{end}}],\ y_0 \in [y_0]\}$;
  if $\exists t_i \in [0, t_{end}]$ *such that* $[y_{t_i}] \subseteq V$ **then**
    if $\forall t \in [0, t_i], [y_t] \subseteq K$ **then**
      $V := V \cup [y_0]$;
    end
  end
end



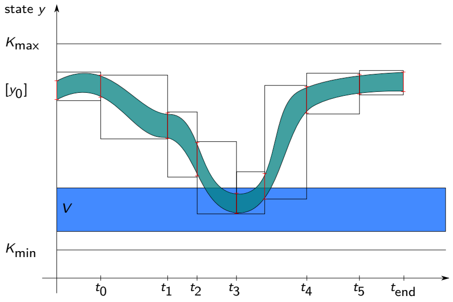
Capture basin: Algorithm

input :
V the current inner approximation of the viability kernel;
$[y_0] \in K \setminus V$;
$U_p = (u_1, u_2, \ldots, u_p)$ a sampling of $\mathcal{U}$;
for $u_i \in U_p$ **do**
  compute $[y_{t_\mathrm{end}}] \supseteq \{y(t; y_0, u_i) \mid t \in [0, t_\mathrm{end}],\ y_0 \in [y_0]\}$;
  if $\exists t_i \in [0, t_{end}]$ *such that* $[y_{t_i}] \subseteq V$ **then**
    if $\forall t \in [0, t_i], [y_t] \subseteq K$ **then**
      $V := V \cup [y_0]$;
    end
  end
end



Capture basin: Algorithm

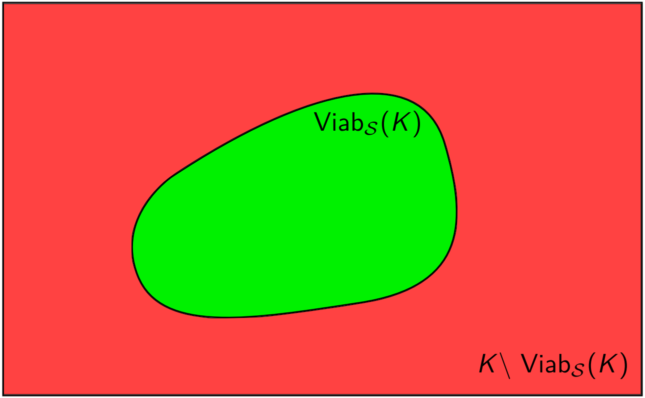
input :
V the current inner approximation of the viability kernel;
$[y_0] \in K \setminus V$;
$U_p = (u_1, u_2, \ldots, u_p)$ a sampling of $\mathcal{U}$;
for $u_i \in U_p$ **do**
compute $[y_{t_{\text{end}}}] \supseteq \{y(t; y_0, u_i) \mid t \in [0, t_{\text{end}}],\ y_0 \in [y_0]\}$;
if $\exists t_i \in [0, t_{end}]$ *such that* $[y_{t_i}] \subseteq V$ **then**
if $\forall t \in [0, t_i]$, $[y_t] \subseteq K$ **then**
$V := V \cup [y_0]$;
end
end
end



This test can be encapsuled in a bisection algorithm to produce an inner approximation of $\text{Viab}_{\mathcal{S}}(K)$.

Outline

$K \setminus \mathrm{Viab}_{\mathcal{S}}(K)$



$$K \setminus \mathrm{Viab}_{\mathcal{S}}(K) = \{y_0 \in K \mid \forall u \in \mathcal{U},\ \exists t \geqslant 0, y(t; y_0, u) \notin K\}$$

Prove that a box $[y_0]$ belongs to $K \setminus \text{Viab}_{\mathcal{S}}(K)$

If for a box $[y_0]$, there exists t such that

$$y(t; [y_0], \mathcal{U}) = \{y(t; y_0, u) \mid y_0 \in [y_0], u \in \mathcal{U}\} \cap K = \emptyset$$

then $[y_0] \subseteq K \setminus \text{Viab}_{\mathcal{S}}(K)$.

Validated numerical integration methods

$y(t; [y_0], \mathcal{U})$ can be over-approximated by a box $[y](t; [y_0], \mathcal{U})$.

$$[y](t; [y_0], \mathcal{U}) \cap K = \emptyset \Rightarrow [y_0] \subset K \setminus \text{Viab}_{\mathcal{S}}(K).$$

Outline

1. Computation of $V_0^- \subseteq \mathrm{Viab}_{\mathcal{S}}(K)$
2. Iteration of $V_{i+1}^- = \mathrm{Capt}_{\mathcal{S}}^{t_{\mathrm{end}}}(K, V_i^-)$
3. Computation of $K \setminus \mathrm{Viab}_{\mathcal{S}}(K)$
4. Examples
5. Conclusion

Car on the hill

Compute the viability kernel for

$$\begin{cases} \dot{y}_1 = y_2(t) \\ \dot{y}_2 = -9.81 \sin\left(\frac{1.1 \sin(1.2 * y_1(t)) - 1.2 \sin(1.1 y_1(t))}{2.0}\right) - 0.7 y_2(t) + u(t) \end{cases}$$

with

$$y(0) \in K = \begin{pmatrix} [-1, 13] \\ [-6, 6] \end{pmatrix}$$

and

$$u(t) \in [-3, 3]$$

Computation of V_0^-

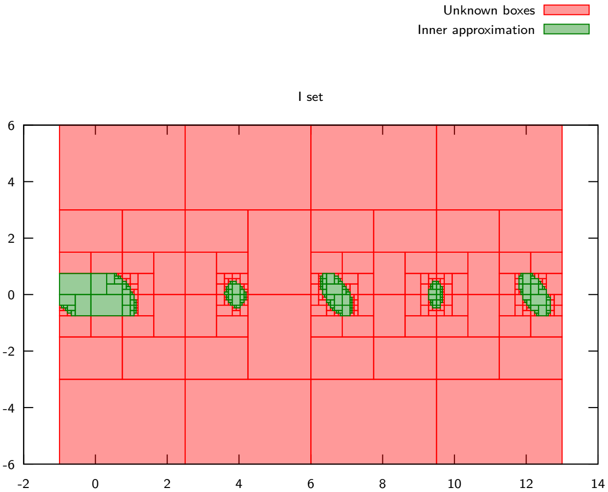


Figure: Ellipsoids

Computation of the recurrence $V_{i+1}^{-} = \text{Capt}_{\mathcal{S}}^{t_{\text{end}}}(K, V_i^{-})$

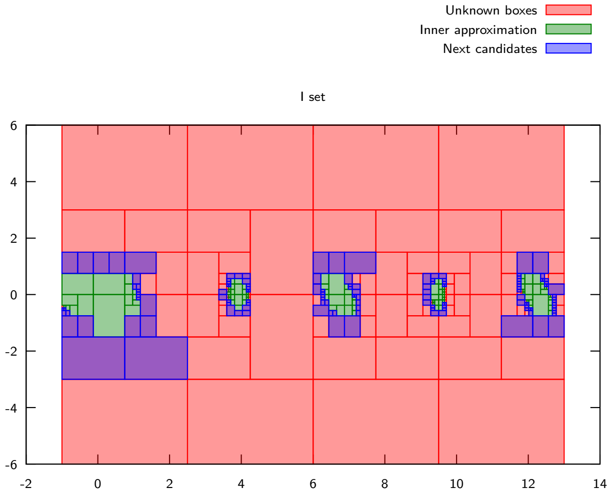


Figure: 1 iteration

Computation of the recurrence $V_{i+1}^{-} = \text{Capt}_{\mathcal{S}}^{t_{\text{end}}}(K, V_i^{-})$

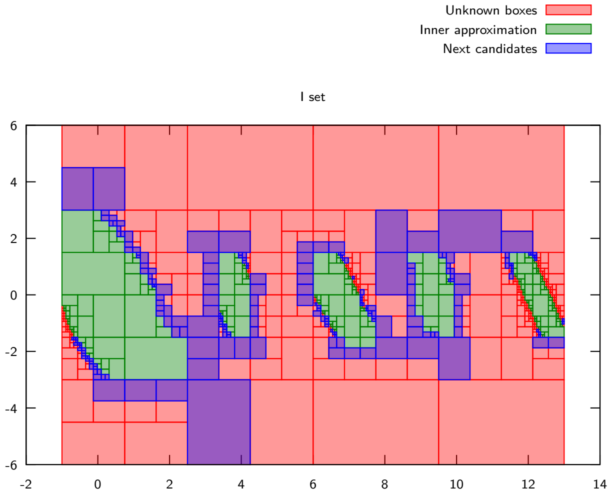


Figure: 10 iterations (2mn50)

Computation of the recurrence $V_{i+1}^{-} = \mathrm{Capt}_{\mathcal{S}}^{t_{\mathrm{end}}}(K, V_i^{-})$

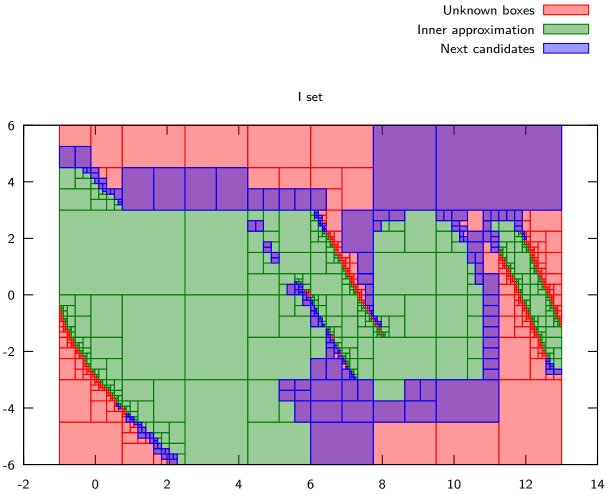


Figure: 20 iterations (6mn30)

Computation of the recurrence $V_{i+1}^{-} = \mathrm{Capt}_{\mathcal{S}}^{t_{\mathrm{end}}}(K, V_i^{-})$

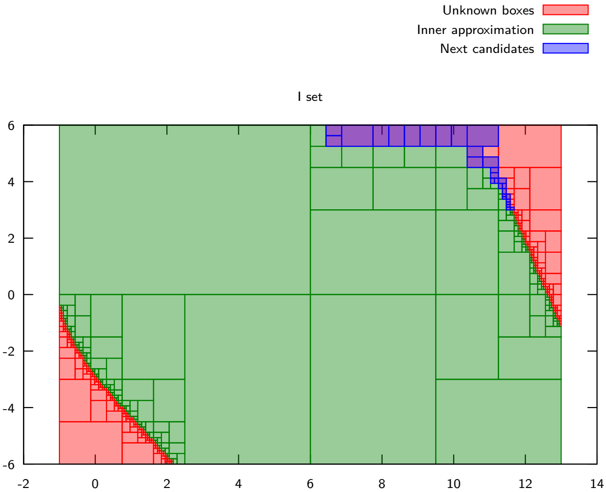


Figure: 30 iterations (7mn45)

Computation of the recurrence $V_{i+1}^{-} = \mathrm{Capt}_{\mathcal{S}}^{t_{\mathrm{end}}}(K, V_i^{-})$

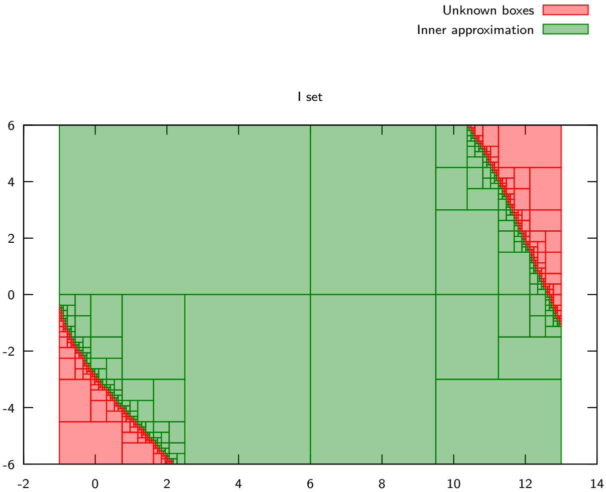


Figure: 40 iterations (8mn)

Computation of the recurrence $V_{i+1}^{-} = \text{Capt}_{\mathcal{S}}^{t_{\text{end}}}(K, V_i^{-})$

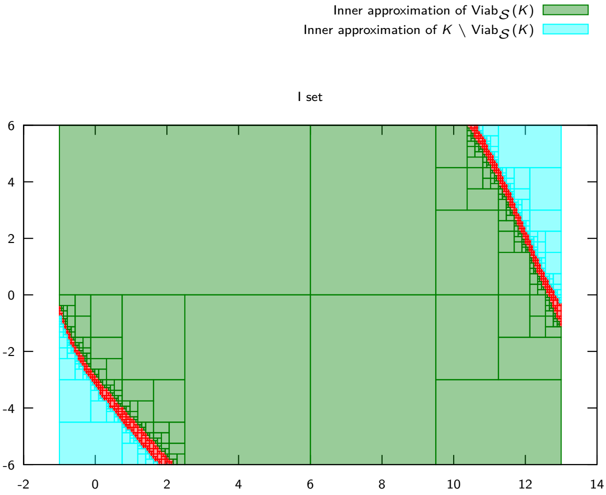


Figure: Result outer and inner (9mn)

Triple integrator

Compute the viability kernel for

$$\begin{cases} \dot{y}_1 = y_2(t) \\ \dot{y}_2 = y_3(t) \\ \dot{y}_3 = u(t) \end{cases}$$

with $y(0) \in K = [-5, 5]^3$ and $u(t) \in [-1, 1]$.

Computation of V_0^-

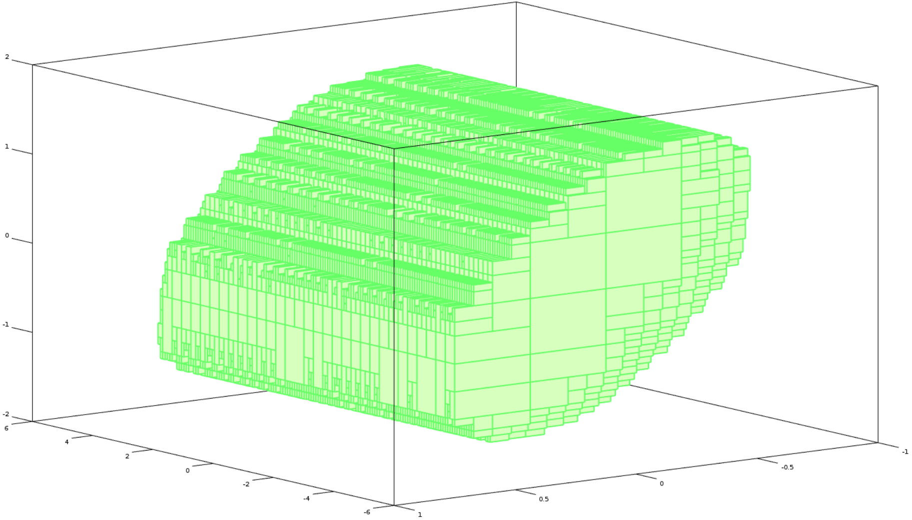


Figure: Ellipsoids

Computation of the recurrence $V_{i+1}^{-} = \mathrm{Capt}_{\mathcal{S}}^{t_{\mathrm{end}}}(K, V_i^{-})$

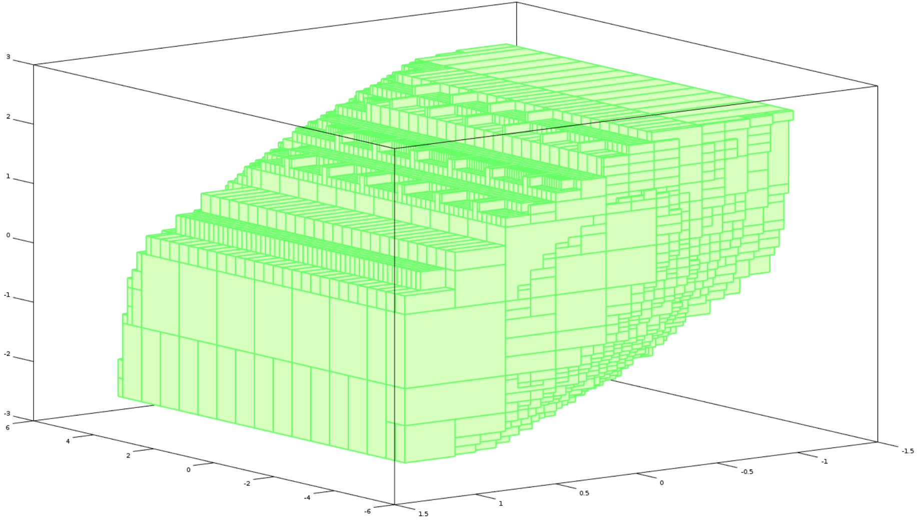


Figure: 10 iterations (5mn)

Computation of the recurrence $V_{i+1}^{-} = \mathrm{Capt}_{\mathcal{S}}^{t_{\text{end}}}(K, V_i^{-})$

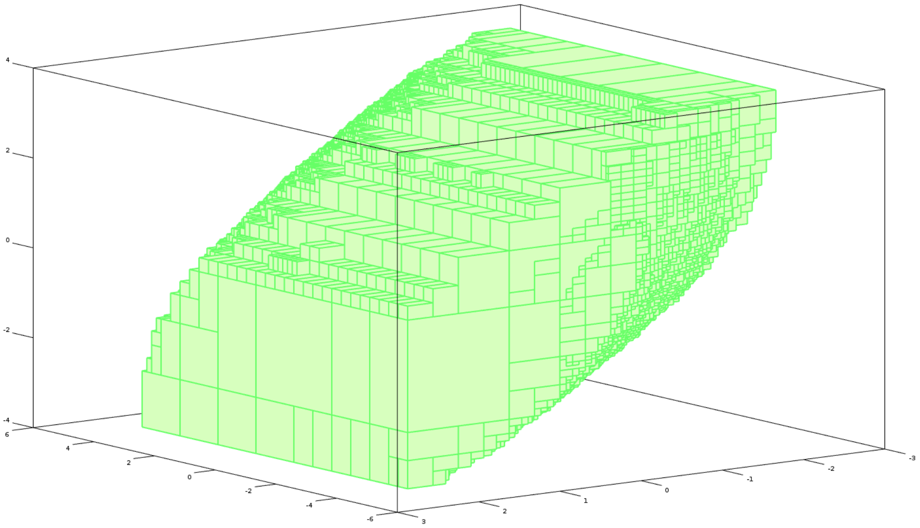


Figure: 20 iterations (11mn40s)

Computation of the recurrence $V_{i+1}^{-} = \mathrm{Capt}_{\mathcal{S}}^{t_{\mathrm{end}}}(K, V_i^{-})$

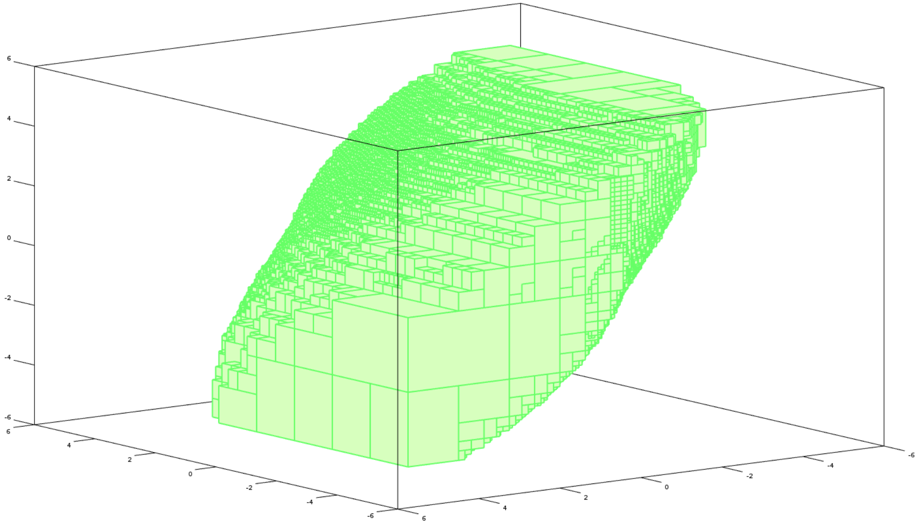


Figure: 30 iterations (25mn30s)

Computation of the recurrence $V_{i+1}^{-} = \mathrm{Capt}_{\mathcal{S}}^{t_{\mathrm{end}}}(K, V_i^{-})$

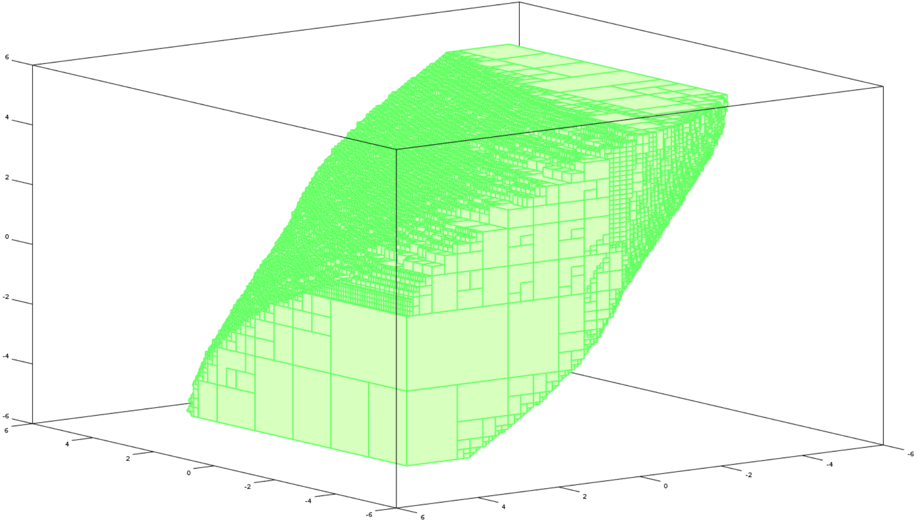


Figure: 40 iterations (46mn)

Computation of the recurrence $V_{i+1}^{-} = \mathrm{Capt}_{\mathcal{S}}^{t_{\mathrm{end}}}(K, V_i^{-})$

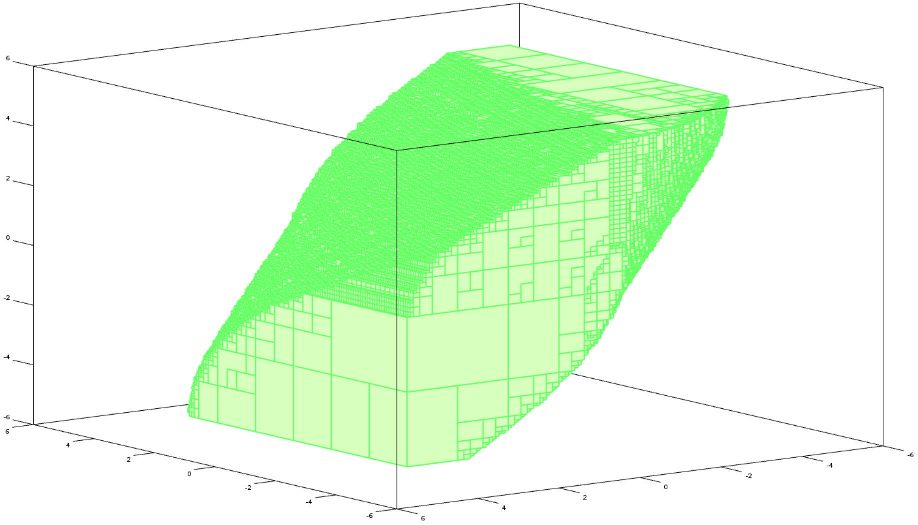


Figure: 50 iterations (58mn)

Computation of the recurrence $V_{i+1}^{-} = \text{Capt}_{\mathcal{S}}^{t_{\text{end}}}(K, V_i^{-})$

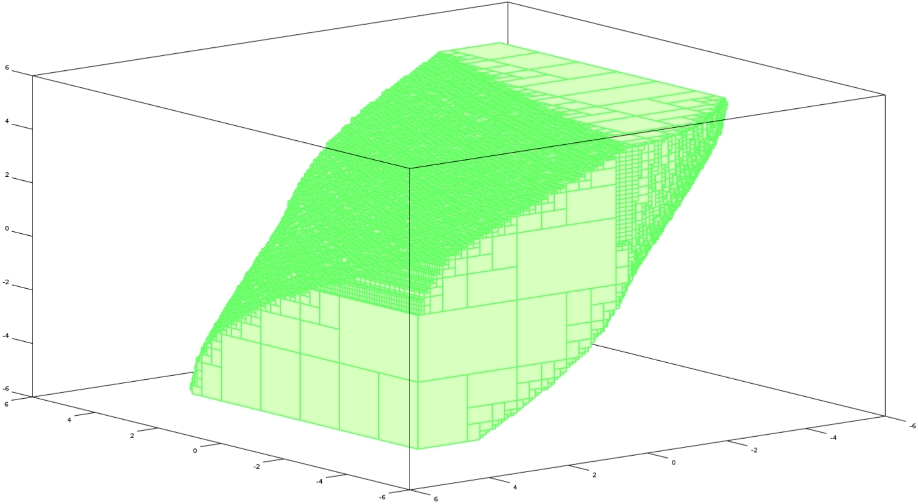


Figure: 62 iterations: end (61mn)

Outline

Conclusion

We implemented an algorithm to produce an inner approximation of the viability kernel of a dynamical system.

- It is an improvement of the previously mentioned algorithm to handle state dimension greater than 2,
- Benefit from last advances on validated numerical integration.

Drawbacks

- bisection algorithm (exponential complexity);
- strong parametrization in the method:
  - ellipsoid proof:
    - precision of the ellipsoid,
    - first parameter c to consider;
  - inner approximation V_0^-: precision (size of the boxes);
  - Numerical integration:
    - numerical integration precision,
    - final time t_{end} for $\text{Capt}_{\mathcal{S}}^{t_{\text{end}}}(K, V_i^-)$,
    - final time t_{end} for $K \setminus \text{Viab}_{\mathcal{S}}(K)$;
  - precision for the bisection.

Perspectives

Example shows that time complexity is still a drawback for the method. It still remains improvements to consider to ameliorate this:

- Find a better first inner approximation V_0^-,
- "Intelligent" enumeration of the control that prove that a box is in the viability kernel.