Overview of the *IBEX* library

**Jordan Ninin**  
*Lab-STICC / ENSTA-Bretagne*  
*Brest, France*

with  
Gilles CHABERT  
*INRIA / Ecole des Mines de Nantes*  
et al.

Toulouse, March 2016
Observations:

- In Automatic, Robotic, Electronic or Mechanic, engineers know very well their problems.
- They solve iteratively their problem by increasing the difficulty, the number of constraints to converge to the real problem they want to solve.
- But, most of time they change the kind of optimization problem: LP, NLP, MINLP, SDP, DFO,...
Introduction

Observations:

- In Automatic, Robotic, Electronic or Mechanic, engineers know very well their problems.
- They solve iteratively their problem by increasing the difficulty, the number of constraints to converge to the real problem they want to solve.
- But, most of time they change the kind of optimization problem: LP, NLP, MINLP, SDP, DFO, ...

What do they need?

⇒ Involve with the modelization improvement.
⇒ Construct the best optimizer for their own problems.
Introduction

Observations:

- In Automatic, Robotic, Electronic or Mechanic, engineers know very well their problems.
- They solve iteratively their problem by increasing the difficulty, the number of constraints to converge to the real problem they want to solve.
- But, most of time they change the kind of optimization problem: LP, NLP, MINLP, SDP, DFO,...

What do they need?

⇒ Involve with the modelization improvement.
⇒ Construct the best optimizer for their own problems.
## Contents

1. **Concepts**
   - Set-membership approach
   - Contractor
   - Properties
   - Implementation

2. **IBEXopt**
   - Constraint Satisfaction Problem
   - Global Optimisation

3. **Related Project**
   - DynIBEX
   - ViablIBEX
   - $H_{\infty}$ control synthesis
# Contents

1. Concepts
   - Set-membership approach
   - Contractor
   - Properties
   - Implementation

2. IBEXopt
   - Constraint Satisfaction Problem
   - Global Optimisation

3. Related Project
   - DynIBEX
   - ViablIBEX
   - $H_\infty$ control synthesis
Interval Arithmetic

Ideas

- Consider Sets in place of Real Numbers
- Need ordered structure: Lattice

\[ \mathbb{IR} = \{ [x, \bar{x}] : x \in \mathbb{R} \cup \{-\infty\}, \bar{x} \in \mathbb{R} \cup \{+\infty\} \text{ and } x \leq \bar{x} \} \cup \{\emptyset\} \]
Interval Arithmetic

Ideas

- Consider Sets in place of Real Numbers
- Need ordered structure: Lattice

\[ \mathbb{IR} = \{[x, \bar{x}] : x \in \mathbb{R} \cup \{-\infty\}, \bar{x} \in \mathbb{R} \cup \{+\infty\} \text{ and } x \leq \bar{x} \} \cup \{\emptyset\} \]

Why do we use the Interval Arithmetic $\mathbb{IR}^n$?

- Based on about 50 years of experience and studies,
- Best way to manipulate sets and boxes,
- There is no problem to deal with discontinuity or unusual functions:
  - $\forall x \in \mathbb{IR}, f(x) = 1/x$ is well defined: $f([0, 0]) = [-\infty, \infty]$.
  - $(\nabla \text{abs})([0, 0]) = [-1, 1]$,
  - $\chi(x, y, z)$ to implement IF statement
  - $\text{atan2}(y, x)$ give the angle of the vector $(x, y)$.
- Easy to enclose Algorithm as contractor,
Definition: Contractor

Let $X \subseteq \mathbb{R}^n$ be a "feasible" region,

The operator $C_X : \mathbb{I}^n \rightarrow \mathbb{I}^n$ is a contractor for $X$ if:

$$\forall x \in \mathbb{I}^n, \left\{ \begin{array}{l}
C_X(x) \subseteq x, \\
C_X(x) \cap X = x \cap X.
\end{array} \right.$$  (contractance)  (completeness)
Definition: Contractor

Let \( X \subseteq \mathbb{R}^n \) be a ”feasible” region,

The operator \( C_X : \mathbb{R}^n \rightarrow \mathbb{R}^n \) is a contractor for \( X \) if:

\[
\forall x \in \mathbb{R}^n, \left\{ \begin{array}{l}
C_X(x) \subseteq x, \\
C_X(x) \cap X = x \cap X.
\end{array} \right. \quad \text{(contractance)}
\]

\[
\forall x \in \mathbb{R}^n, \left\{ \begin{array}{l}
x \in x \text{ and } f(x) = 0 \implies x \in C(x).
\end{array} \right. \quad \text{(completeness)}
\]

Example:

The operator \( C : \mathbb{R}^n \rightarrow \mathbb{R}^n \) is a contractor for the equation \( f(x) = 0 \), if:

\[
\forall x \in \mathbb{R}^n, \left\{ \begin{array}{l}
C(x) \subseteq x, \\
x \in x \text{ and } f(x) = 0 \implies x \in C(x).
\end{array} \right.
\]
Algebra on Contractors

Let $\mathcal{A}$ a contractor for the equation $f(x) = 0$, and $\mathcal{B}$ a contractor for the equation $g(x) = 0$, then:

**Intersection, Composition**

$\mathcal{A} \cap \mathcal{B}$ and $\mathcal{A} \circ \mathcal{B}$ are two contractors for the region:

$$\{x \in \mathbb{R}^n : f(x) = 0 \ \text{AND} \ \ g(x) = 0\}$$

**Union**

$\mathcal{A} \cup \mathcal{B}$ is a contractor for the region:

$$\{x \in \mathbb{R}^n : f(x) = 0 \ \text{OR} \ \ g(x) = 0\}$$
Let $A$ a contractor for the equation $f(x) = 0$, and $B$ a contractor for the equation $g(x) = 0$, then:

**Intersection, Composition**

$A \cap B$ and $A \circ B$ are two contractors for the region:

$$\{x \in \mathbb{R}^n : f(x) = 0 \text{ AND } g(x) = 0\}$$

**Union**

$A \cup B$ is a contractor for the region:

$$\{x \in \mathbb{R}^n : f(x) = 0 \text{ OR } g(x) = 0\}$$
Implementation

- Arithmetic:
  - **Interval Arithmetic**: 4 versions
    Filib++, Gaol, Profil, 1 homemade.
  - **Affine Arithmetics**: 7 versions
    fast/reliable, dynamic/static, Floating point/Interval

- Contractor:
  - Forward-Backward Contractor,
  - Convex Hull Contractor based on Linear Relaxation,
  - Contractor with quantifier,
  - ”Non-Mathematical” Contractor,
  - ... your own contractor.

- Python3 Interface and userfriendly Installation:

  ```
pip install pyIbex
  ```
Contents

1 Concepts
   Set-membership approach
   Contractor
   Properties
   Implementation

2 IBEXopt
   Constraint Satisfaction Problem
   Global Optimisation

3 Related Project
   DynIBEX
   ViabIBEX
   $H_\infty$ control synthesis
Constraint Satisfaction Problem

Example 1/3

Let us consider the following equations:

\[ S_1 = \{(x, y) \in \mathbb{R}^2 \mid \left(\frac{x}{4}\right)^2 + \left(\frac{y}{3}\right)^2 - 1 \leq 0\} \text{ AND } (y \geq 0) \]  
\[ S_2 = \{(x, y) \in \mathbb{R}^2 \mid (x - \frac{y}{2} \geq -4) \text{ AND } (x + \frac{y}{2} \leq 4) \text{ AND } (y \in [-4, 0])\} \]  
\[ S_3 = \{(x, y) \in \mathbb{R}^2 \mid \left(\frac{x+1.8}{3.5}\right)^2 + \left(\frac{y+0.3}{2}\right)^2 \geq 0.1\} \text{ AND } \left(\frac{x-1.8}{3.5}\right)^2 + \left(\frac{y+0.3}{2}\right)^2 \geq 0.1\} \text{ AND } (x^2 \text{ AND } (y \leq -2.8) \text{ OR } (\cos(10x) \in [0, 2]))} \]  
\[ S_5 = \{(x, y) \in \mathbb{R}^2 \mid ((x+y \geq -4) \text{ AND } (x - y \leq 4)) \text{ OR } \left(y + \frac{\cos 1.1x}{2} \geq -3\right)\} \]  
\[ S_6 = \{(x, y) \in \mathbb{R}^2 \mid ((\frac{5x}{3} + y - 6) (y - \frac{x}{3}) \not\in [0.4999, 0.5001]) \text{ AND } (y \leq 1) \text{ AND } (x \geq 0)\} \]  
\[ S_7 = \{(x, y) \in \mathbb{R}^2 \mid ((\frac{-5x}{3} + y - 6) (y + \frac{x}{3}) \not\in [0.4999, 0.5001]) \text{ AND } (y \leq 1) \text{ AND } (x \leq 0)\} \]  
\[ S_8 = \{(x, y) \in \mathbb{R}^2 \mid y \leq 1\} \]  
\[ S_9 = \{(x, y) \in \mathbb{R}^2 \mid (|x| \not\in [0.1999, 0.2001]) \text{ AND } (y \geq -0.1)\} \]  
\[ S_{10} = \{(x, y) \in \mathbb{R}^2 \mid y \leq 0.1\} \]  

Draw the area defined by the following set \( S \).

\[ S = (S_1 \cup S_2) \cap S_3 \cap S_4 \cap S_5 \cap (S_6 \cup S_7 \cup S_8) \cap (S_9 \cup S_{10}) \]
Example 2/3
Example 3/3

\[
\begin{align*}
\text{sep1} &= \text{SepFwdBwd} \left( f \left( \frac{x}{4} \right)^2 + \frac{y}{3} + 2 - 1 \right), \text{Interval} (-\infty, 0) \right) \text{ & SepFwdBwd} \left( f \left( \frac{x}{4} \right)^2 + \frac{y}{3} + 2 - 1 \right), \text{Interval} (-\infty, 0) \right) \\
\text{sep2} &= \text{SepFwdBwd} \left( f \left( x - \frac{y}{2} \right) \right), \text{Interval} (-4, \infty) \right) \text{ & SepFwdBwd} \left( f \left( x + \frac{y}{2} \right) \right), \text{Interval} (-4, \infty) \right) \\
\text{sep3} &= \text{SepFwdBwd} \left( f \left( \left( \frac{x+1.8}{3.5} \right)^2 + \left( \frac{y+0.3}{2} \right)^2 \right) \right), \text{Interval} (0.1, \infty) \right) \\
\text{sep4} &= \text{SepFwdBwd} \left( f \left( x - y \right) \right), \text{Interval} (-\infty, 1) \right) \text{ & SepFwdBwd} \left( f \left( x + y \right) \right), \text{Interval} (-\infty, 1) \right) \\
\text{sep5} &= \text{SepFwdBwd} \left( f \left( x + y \right) \right), \text{Interval} (-4, 0) \right) \text{ & SepFwdBwd} \left( f \left( x - y \right) \right), \text{Interval} (-4, 0) \right) \\
\text{sep6} &= \text{SepNot} \left( \text{SepFwdBwd} \left( f \left( 5 \cdot \frac{x}{3} + y - 6 \right) \cdot \left( y - \frac{x}{3} \right) \right), \text{Interval} (0.4999, 0.5) \right) \text{ & SepFwdBwd} \left( f \left( x \right) \right), \text{Interval} (0, \infty) \right) \\
\text{sep7} &= \text{SepNot} \left( \text{SepFwdBwd} \left( f \left( 5 \cdot \frac{x}{3} + y - 6 \right) \cdot \left( y + \frac{x}{3} \right) \right), \text{Interval} (0.4999, 0.5) \right) \text{ & SepFwdBwd} \left( f \left( y \right) \right), \text{Interval} (1, \infty) \right) \\
\text{sep8} &= \text{SepFwdBwd} \left( f \left( \left( \frac{x+1.8}{3.5} \right)^2 + \left( \frac{y+0.3}{2} \right)^2 \right) \right), \text{Interval} (0.1, \infty) \right) \\
\text{sep9} &= \text{SepNot} \left( \text{SepFwdBwd} \left( f \left( \left( \frac{x+1.8}{3.5} \right)^2 + \left( \frac{y+0.3}{2} \right)^2 \right) \right), \text{Interval} (0.1, \infty) \right) \text{ & SepFwdBwd} \left( f \left( \left( \frac{x+1.8}{3.5} \right)^2 + \left( \frac{y+0.3}{2} \right)^2 \right) \right), \text{Interval} (0.1, \infty) \right) \\
\text{sep10} &= \text{SepFwdBwd} \left( f \left( \left( \frac{x+1.8}{3.5} \right)^2 + \left( \frac{y+0.3}{2} \right)^2 \right) \right), \text{Interval} (0.1, \infty) \right)
\end{align*}
\]

\[
\text{sep} = (\text{sep1 | sep2}) \text{ & sep3} \text{ & sep4} \text{ & sep5} \text{ & (sep6 | sep7 | sep8)} \text{ & (sep9 | sep10)}
\]

\[
\text{pySIVIA} \left( \text{IntervalVector} \left( [[-10, 10], [-10, 10]] \right), \text{sep}, \text{epsilon} \right)
\]
We consider global optimization of Non Linear Programming problems in a deterministic and reliable way.

**Problem**

\[
\begin{align*}
\min_{x \in X \subset \mathbb{R}^n} & \quad f(x) \\
\text{s.t.} & \quad g_l(x) \leq 0, \quad \forall l \in \{1, \ldots, p\}, \\
& \quad h_k(x) = 0, \quad \forall k \in \{1, \ldots, q\}.
\end{align*}
\]

**Modelization**

- **AMPL**
- formal tool of *IBEX*
Global Optimisation

Branch and Bound Algorithm

Each iteration:

- **Choice** and **Subdivision of the box** \(X\) (into 2 boxes),
  \[ \Rightarrow \mathcal{L} \text{ list of possible solutions} \]

- **Computation of lower bounds**
  \[ \Rightarrow \text{Interval Arithmetic, Affine Arithmetic,...} \]

- **Elimination** of boxes that cannot contain the global optimum
  \[ \Rightarrow \text{Elts which do not satisfy constraints, lower bound } > \tilde{f},.. \]

El: **Store in** \(\mathcal{L}\)

- **STOP**
  \[ \Rightarrow \max_{(Z,f_Z)\in \mathcal{L}} \text{wid}(Z) \leq \epsilon_L \]
  \[ \Rightarrow \tilde{f} - \min_{(Z,f_Z)\in \mathcal{L}} f_Z \leq \epsilon_f \]
Branch and Bound Algorithm

Each iteration:

- **Choice** and **Subdivision of the box** $\mathbf{X}$ (into 2 boxes),
  \[ \implies \mathcal{L} \text{ list of possible solutions} \]

- **Contract each sub-boxes**, :
  \[ \implies \text{CtcAcid} \cap (\text{CtcPolytopeHull} \cap \text{CtcHC4})^\infty. \]

- **Computation of lower bounds**
  \[ \implies \text{Interval Arithmetic, Affine Arithmetic,...} \]

- **Elimination** of boxes that cannot contain the global optimum
  \[ \implies \text{Elts which do not satisfy constraints, lower bound } > \tilde{f}, \ldots \]

  **Else:** **Store in** $\mathcal{L}$

- **STOP**
  \[ \implies \max_{(\mathbb{Z}, f_z) \in \mathcal{L}} \text{wid}(\mathbb{Z}) \leq \epsilon_L \]

  \[ \implies \tilde{f} - \min_{(\mathbb{Z}, f_z) \in \mathcal{L}} f_z \leq \epsilon_f \]
Higher is better

⇒ Still need progress **BUT** we can deal with a more large variaty of problems with cos, sin, atan,...
Contents

1 Concepts
   - Set-membership approach
   - Contractor
   - Properties
   - Implementation

2 IBEXopt
   - Constraint Satisfaction Problem
   - Global Optimisation

3 Related Project
   - DynIBEX
   - ViabIBEX
   \( H_\infty \) control synthesis
Introduction

DynIBEX

DynIBEX: A. Chapoutot and J. Alexandre dit Sandretto

Guaranteed numerical integration method for ODE:

Ordinary Differential Equation with a given initial condition

We consider an initial value problem (IVP):\[
\dot{y}(t) = f(t, y(t)) \quad \text{with} \quad y(0) = y_0
\]

The goal is to compute the sequence of boxes \((t_n, [y_n])_{n \in \mathbb{N}}\) such that \([y_n] \supseteq y(t_n; [y_{n-1}]) = \{y(t_n; y_{n-1}) : \forall y_{n-1} \in [y_{n-1}]\}\).

Differential-algebraic equations in Hessenberg index 1 form with consistent initial conditions

\[
\begin{align*}
\dot{y}(t) &= f(t, y(t), x(t)) & \text{with} & \quad y(0) = y_0 \\
0 &= g(t, y(t), x(t)) & \text{and} & \quad x(0) = x_0
\end{align*}
\]
System $S$ defined by:

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

A state $x$ is viable if at least one evolution of $S$ from $x$ can stay indefinitely in a set of constraints $K$. 
ViabIBEX: D. Monnet, L. Jaulin, J. Ninin

![Image of ViabIBEX software interface]

- **V-viable functions**
- **Show Vector Field**: 0
- **More options**: V-viability, Polygon, Integration
- **Fast computation**: Off
- **Nb control**: 3
- **Ignore viable points**: On

**Evolution Function**

\[ x^2 - 9.81 \sin((1.1 \sin(1.2x1) - 1.2 \sin(1.1x1))/2) - 0.7x2 + u \]

- **Ignore equilibrium point**: (4.29513, 0), in viable set
- **7 viable sets found in 0.000569 seconds**
- **25.1205 % of space viable, 0 % of space not viable**
- **25.1205 % of space characterized**

**Clear log**
ViabIBEX: D. Monnet, L. Jaulin, J. Ninin

![ViabIBEX GUI](image)

Ignore equilibrium point 4/4: (4.29513,0), in viable set
7 viable sets found in 0 seconds
25.1205% of space viable, 0% of space not viable
25.1205% of space characterized
ViabIBEX: D. Monnet, L. Jaulin, J. Ninin
$H_\infty$ control synthesis

**SynthIBEX:** D. Monnet, J. Ninin, C. Clément

\[
\begin{align*}
&H_\infty \text{ control synthesis} \Rightarrow \text{Guarantee the robustness and stability} \\
&\|P\|_\infty = \sup_{\omega}(\sigma_{\text{max}}(P(j\omega)))
\end{align*}
\]
**SynthIBEX:** D. Monnet, J. Ninin, C. Clément

\[ H_\infty \text{ control synthesis} \Rightarrow \text{Guarantee the robustness and stability} \]

\[ ||P||_\infty = \sup_{\omega} (\sigma_{\max}(P(j\omega))) \]

- Classical approach without structural constraint (convex problem) \( \Rightarrow \) LMI system, SDP optimization
- Classical approach with structural constraint (non-convex problem) \( \Rightarrow \) Nonsmooth local optimization [Apkarian and Noll, 2006]
The closed-loop system must be stable.

\[
\begin{align*}
\min_k \sup_{\omega} \max \left( \left\| \frac{W_1(j\omega)}{1 + G(j\omega)K(j\omega)} \right\|_{\infty}, \left\| \frac{W_2(j\omega)K(j\omega)}{1 + G(j\omega)K(j\omega)} \right\|_{\infty} \right)
\end{align*}
\]

Stability:

The system is stable iff its poles are strictly negative.

\[
\Leftrightarrow \frac{1}{1 + G(s)K(s)} \text{ are strictly negative}
\]

Routh-Hurwitz stability criterion
$H_\infty$ control synthesis under structural constraint \iff

Solve a min/max problem with non-convex constraints

\[
\min_{k \in \mathbb{K}} \sup_{\omega} f(k, \omega) \\
s.t. \quad c_i(k) \leq 0, \quad \forall i \in \{1, \ldots, p\},
\]

Branch and Bound algorithm for min/max problem

- Global optimization approach
- Guaranteed enclose of the global minimum
- Certificate of infeasibility
If you want to know more about Contractors

**IAMOOC**
Interval Analysis MOOC
with Luc Jaulin and Jordan Ninin

http://iamooc.ensta-bretagne.fr

**IBEX**
http://www.ibex-lib.org

Fork it on GitHub
http://github.com/ibex-team/ibex-lib

pip install pyIbex