

Programmation par contraintes sur intervalles

*Application en optimisation globale et à des problèmes
d'identification*

Gilles Trombettoni

Journée du LIRMM
Montarnaud, le 19 juin 2014



Plan

Introduction

Exemples d'applications

Intervalles, boîtes, contracteurs, identification à intervalles

Optimisation globale

Introduction

A quoi servent les méthodes à intervalles ?

⇒ Outil informatico-mathématique pour travailler sur des
fonctions mathématiques sur les nombres réels
(équations/inégalités)

Motivation initiale (dont je ne parlerai plus) :
calcul sur les nombres flottants **rigoureux**
(erreurs d'arrondis pris en compte). Exemple :

Surface d'un disque =

$$\pi \cdot r^2 = [3.14, 3.15] * [9.9, 10.1]^2 = [307.75, 321.34]$$

Exemple de système de contraintes (AOL-legentil.bch)

Trouver toutes les solutions de :

Variables

```
x in [1e-10, pi/2-1e-10];  
y in [0, pi/2-1e-10];  
z in [-1e8, 1e8];
```

Constraints

```
10/3*cos(x)/sin(x)^2+4*(1+tan(x)^2)/cos(y)  
+z*(-50/3*sin(y)*cos(x)/(sin(x)^2*(3.5-5*sin(y)))  
-10/3*cos(x)/sin(x)^2-4*(1+tan(x)^2)/cos(y))=0;  
  
4*tan(x)*sin(y)/cos(y)^2+z*(50/3*cos(y)/(sin(x)  
*(3.5-5*sin(y)))+250/3*sin(y)*cos(y)/(sin(x)  
*(3.5-5*sin(y))^2)-4*tan(x)*sin(y)/cos(y)^2)=0;  
  
50/3*sin(y)/(sin(x)*(3.5-5*sin(y)))+20+10/3/sin(x)  
-4*tan(x)/cos(y)=0;
```

Exemple d'optimisation sous contraintes (ex6_1_1)

Trouver des valeurs de variables qui minimisent une fonction tout en respectant des contraintes :

Variables

x_2, x_3, x_4, x_5 in $[1e-7, 0.5]$; x_6 in $[0, 0.901]$;
 x_7 in $[0, 0.274]$; x_8 in $[0, 0.69]$; x_9 in $[0, 0.998]$;

Minimize

$$\begin{aligned} & x_2 * (\log(x_2) - \log(x_2 + x_4)) + x_4 * (\log(x_4) - \log(x_2 + x_4)) \\ & + x_3 * (\log(x_3) - \log(x_3 + x_5)) + x_5 * (\log(x_5) - \log(x_3 + x_5)) \\ & + 0.92 * x_2 * x_8 + 0.746 * x_4 * x_6 + 0.92 * x_3 * x_9 + 0.746 * x_5 * x_7; \end{aligned}$$

Subject to $x_6 * (x_2 + 0.159 * x_4) - x_2 = 0$;
 $x_7 * (x_3 + 0.159 * x_5) - x_3 = 0$;
 $x_8 * (0.308 * x_2 + x_4) - x_4 = 0$;
 $x_9 * (0.308 * x_3 + x_5) - x_5 = 0$;
 $x_2 + x_3 = 0.5$; $x_4 + x_5 = 0.5$;

Opérateurs continus et différentiables par morceaux :

$+, -, *, /, power, sin, exp, abs, \dots$

A retenir de l'exposé

- ▶ Les intervalles peuvent résoudre des systèmes de **contraintes numériques difficiles** (de taille petite mais croissante).
- ▶ Idée des principes algorithmiques derrière ces méthodes.
- ▶ Ces méthodes travaillent avec des **ensembles continus de points** (intervalles), et non pas des points comme en analyse numérique.

Communautés scientifiques

Méthodes à intervalles étudiées par plusieurs communautés

Epoque	Communauté	Section du CNU
Ordinateurs	Opérations sur les nombres flottants	
≥ 1965	Analyse (numérique) par intervalles	25 (maths)
Années 1990	Programmation par contraintes	27 (info)
Années 1990/2000	Programmation mathématique	26 (maths applis)
Années 2000	Sciences de l'ingénieur	61 (robotique), ...

Techniques implémentées dans la bibliothèque en C++ Ibex
(*Interval-Based EXplorer*)



Responsable : Gilles Chabert (EMN/LINA, Nantes)

Principaux contributeurs :

Ignacio Araya	U. Catolica, Valparaiso
Bertrand Neveu	LIGM, Paris
Jordan Ninin	ENSTA, Brest
Gilles Trombettoni	LIRMM, Montpellier

Plan

Introduction

Exemples d'applications

Intervalles, boîtes, contracteurs, identification à intervalles

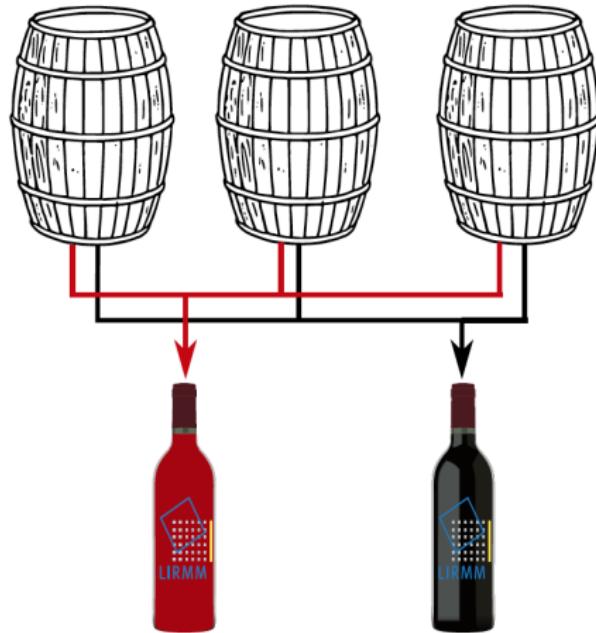
Optimisation globale

Domaines d'application

- ▶ Chimie (conception de colonnes de distillation)
- ▶ Géolocalisation par GPS
- ▶ Robotique (robotique mobile, Simultaneous Localization And Mapping, conception de robots parallèles)
- ▶ Identification (estimation de paramètres), étalonnage
- ▶ Preuve de propriétés mathématiques
- ▶ ...

Exemple : assemblage de vin (avec P. Vismara et R. Coletta)

Assemblage = mélange de plusieurs cuvées pour concevoir un vin cible



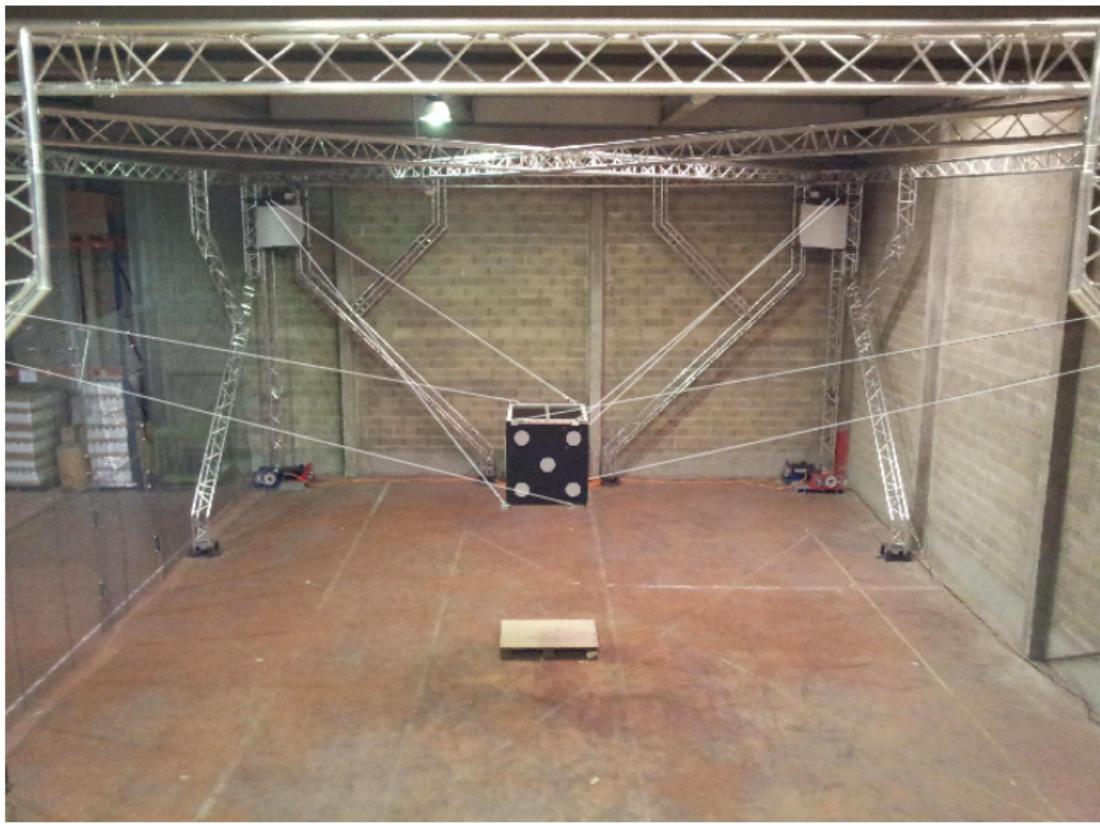
Minimisation de l'écart sur les critères aromatiques : une centaine d'équations

Exemple en robotique parallèle

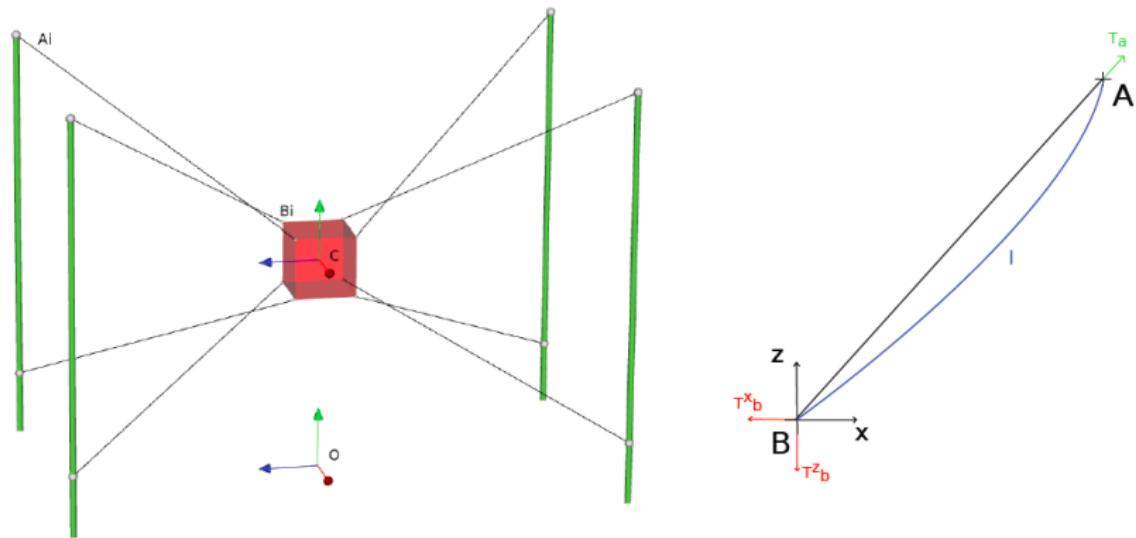
Robot Cogiro (projet ANR piloté par M. Gouttefarde)



Exemple en robotique parallèle



Exemple en robotique parallèle



Prouver un seuil d'erreur maximum

(avec J. Alexandre-dit-Sandretto et D. Daney)

$$\forall A \in [0, 4] \times [0, 3], \forall F_a \in [40, 150] : L_0 - AB < \text{seuil}$$

avec AB tel que :

$$A_x = \frac{F_x L_0}{k} + \frac{|F_x|}{mg} [\sinh^{-1}\left(\frac{F_z}{F_x}\right) - \sinh^{-1}\left(\frac{F_z - mgL_0}{F_x}\right)]$$

$$A_z = \frac{mgL_0^2}{k} \left(\frac{F_z}{mgL_0} - \frac{1}{2} \right) + \frac{1}{mg} [\sqrt{F_x^2 + F_z^2} - \sqrt{F_x^2 + (F_z - mgL_0)^2}]$$

$$0 = F_a - \sqrt{F_x^2 + (F_z - mgL_0)^2}$$

$$\{\exists A \in [0, 4] \times [0, 3], \exists F_a \in [40, 150] : L_0 - AB \geq \text{seuil}\} = \emptyset$$

Plan

Introduction

Exemples d'applications

Intervalles, boîtes, contracteurs, identification à intervalles

Optimisation globale

Intervalles, contracteurs

Intervalle $[x] = [\underline{x}, \bar{x}]$	$\{x \in \mathbb{R}, \underline{x} \leq x \leq \bar{x}\}$
<u>x</u> et \bar{x}	Bornes flottantes
\mathbb{IR}	Ensemble de tous les intervalles
$m([x])$	Point milieu de $[x]$
Boîte $[x]$	$[x_1] \times \dots \times [x_i] \times \dots \times [x_n]$
Contraction	Réduction d'une boîte sans perte de solution
Contracteur	Algorithme de contraction

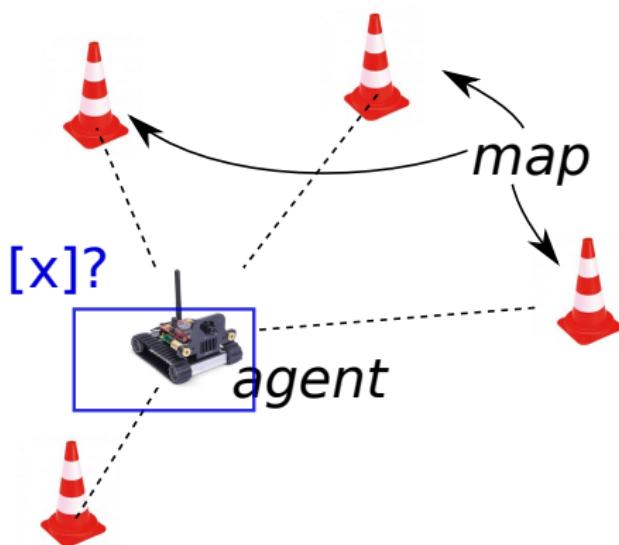
Arithmétique d'intervalles

Etend à \mathbb{IR} les opérateurs mathématiques sur \mathbb{R} :

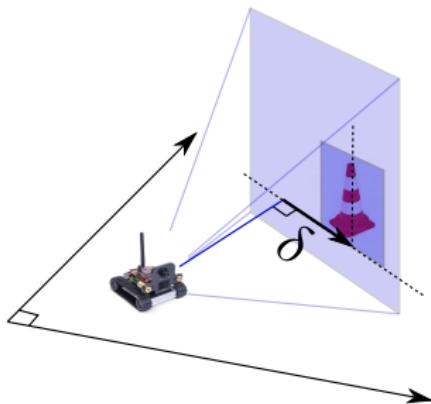
- ▶ $sqr([-4, 2]) = [0, 16]$
- ▶ $[x] + [y] = [\underline{x} + \underline{y}, \bar{x} + \bar{y}]$
- ▶ $[x] \cdot [y] = [\min\{\underline{x}\underline{y}, \underline{x}\bar{y}, \bar{x}\underline{y}, \bar{x}\bar{y}\}, \max\{\underline{x}\underline{y}, \underline{x}\bar{y}, \bar{x}\underline{y}, \bar{x}\bar{y}\}]$
- ▶ $[x].([x] + 1)$ avec $[x] = [-1, +1]$ donne
 $[-1, 1].[0, 2] = [-2, 2]$.
- ▶ L'arithmétique d'intervalles permet un calcul “conservatif”...

Exemple jouet d'identification

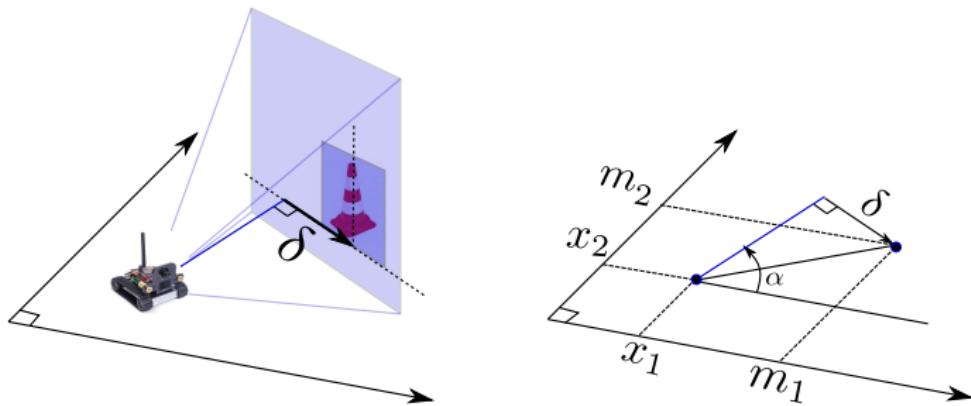
Un robot autonome (mais immobile) doit se géolocaliser (approximer sa position (x_1, x_2) par une petite boîte) à partir d'observations de l'environnement.



Contracteur (l)image : Chaque photo permet de contracter la distance $[\underline{\delta}, \bar{\delta}]$ à une balise reconnue.



Contracteur (I)mage : Chaque photo permet de contracter la distance $[\delta] = [\underline{\delta}, \bar{\delta}]$ à une balise reconnue.

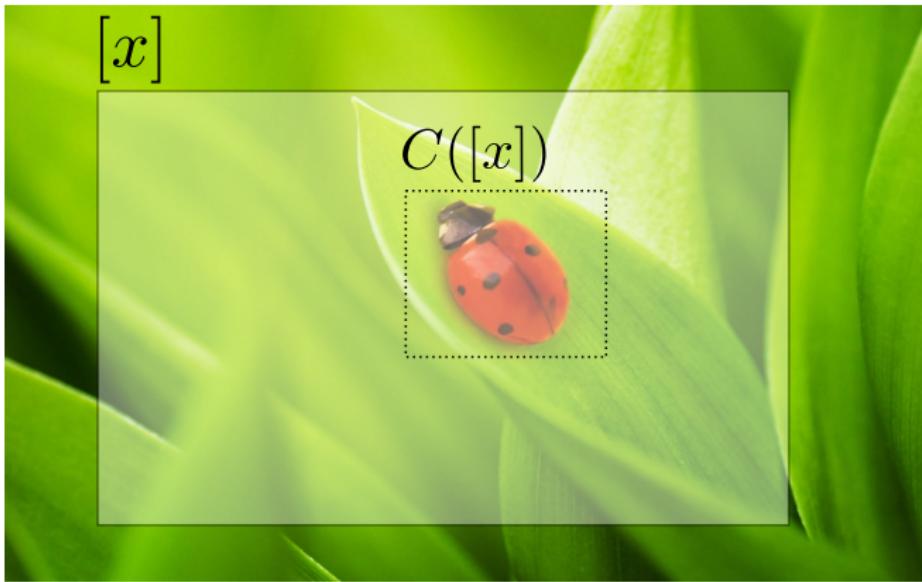


Contracteur (E)quation : L'équation suivante permet de contracter la position $([x_1], [x_2])$ du robot. (Relation entre l'orientation α de la caméra, les coordonnées (m_1, m_2) de la balise.)

$$\sin \left(\alpha - \text{atan} \left[\frac{m_2 - x_2}{m_1 - x_1} \right] \right) = \delta.$$

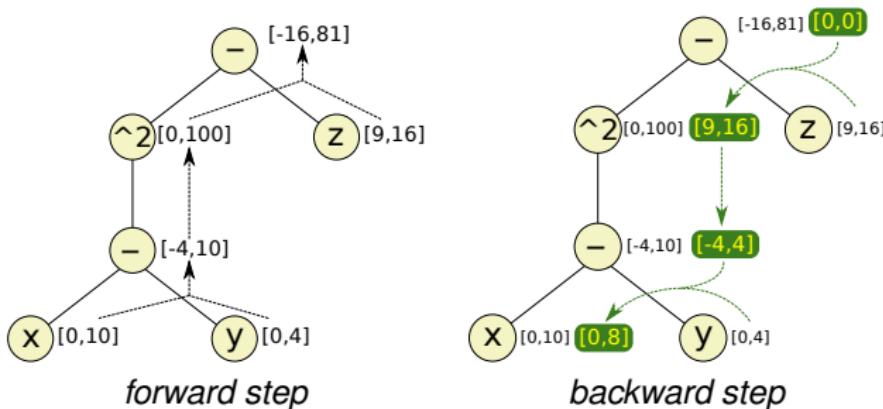
Contracteur / (région d'intérêt)

Il existe des algorithmes capables d'extraire d'une photo des *régions d'intérêt*.



Contracteur E (pour une contrainte)

- ▶ Forward-backward (ou HC4REVISE) est un contracteur classique pour une contrainte, e.g. $(x - y)^2 - z = 0$



- ▶ Mohc-Revise (conçu avec I. Araya et B. Neveu) exploite les monotonies de la contrainte/fonction.

Contracteur composé

Contracteur sans mesures aberrantes (*outliers*)

$$C := \text{fix} \left((E_1 \circ I_1) \circ \dots \circ (E_n \circ I_n) \right)$$

Contracteur composé

Contracteur sans mesures aberrantes (*outliers*)

$$C := \text{fix} \left((E_1 \circ I_1) \circ \dots \circ (E_n \circ I_n) \right)$$

Contracteurs avec mesures aberrantes

Les n contracteurs sont combinés par un opérateur de q -intersection.

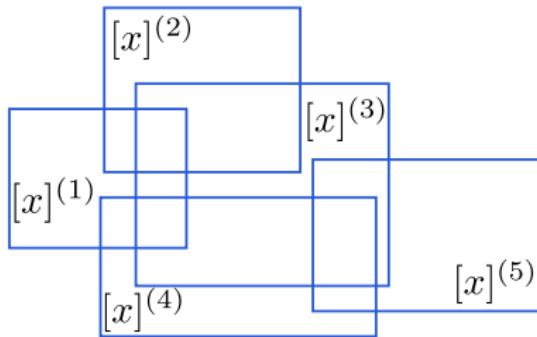
Le contracteur résultant devient :

$$C := \text{fix} \left(\text{qinter}(q, E_1 \circ I_1, \dots, E_n \circ I_n) \right)$$

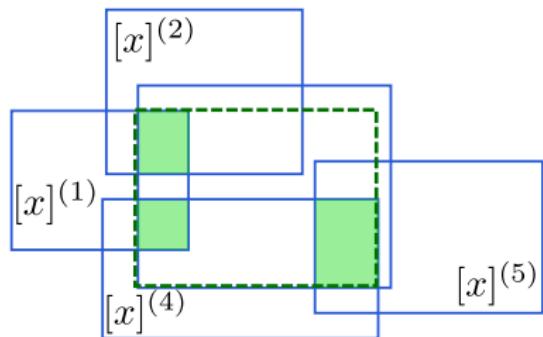
Q-intersection

Définition

La **q-intersection** d'un ensemble de boîtes S est la plus petite boîte incluant tous les points qui appartiennent à au moins q boîtes de S .



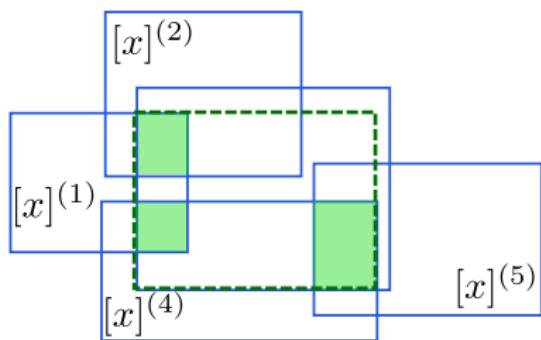
Boîtes (S)



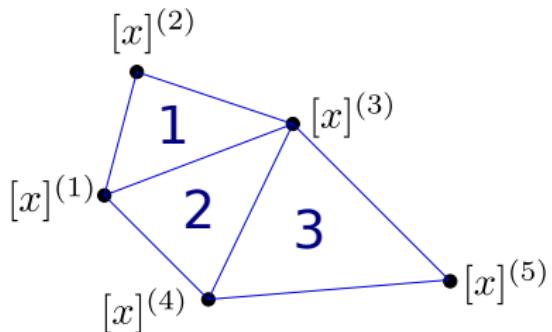
3-intersection

Q-intersection

La q-intersection peut se calculer à partir du **graphe d'intersection** (avec C. Carbonnel, P. Vismara, G. Chabert)



Boîtes



Graphe d'intersection

Extensions possibles

1. **Le robot bouge.**

Mouvement décrit par une équation différentielle ordinaire.
Des techniques d'intégration rigoureuse peuvent être
encapsulées dans un contracteur.

2. **Les marques ont des positions inconnues (*SLAM*).**

Plan

Introduction

Exemples d'applications

Intervalles, boîtes, contracteurs, identification à intervalles

Optimisation globale

Schéma Brancher & Contracter pour la résolution

Résoudre $\{f_1(x_1, x_2) = 0, f_2(x_1, x_2) = 0\}$

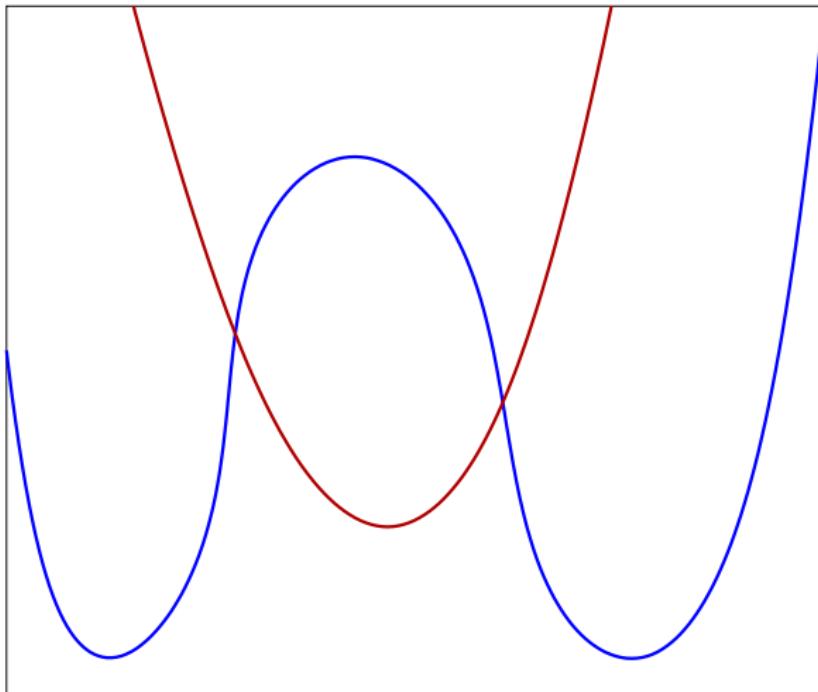


Schéma Brancher & Contracter pour la résolution

Résoudre $\{f_1(x_1, x_2) = 0, f_2(x_1, x_2) = 0\}$

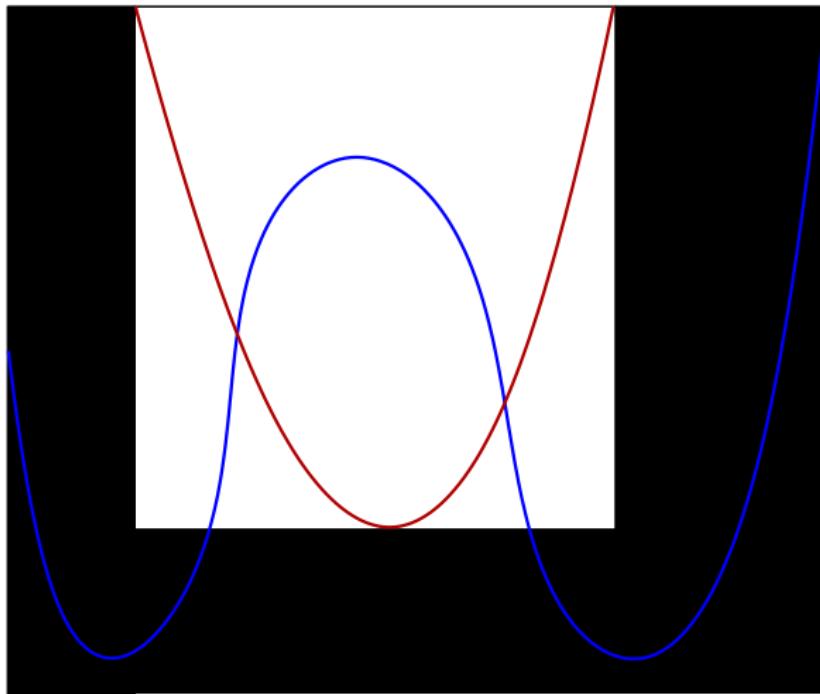


Schéma Brancher & Contracter pour la résolution

Résoudre $\{f_1(x_1, x_2) = 0, f_2(x_1, x_2) = 0\}$

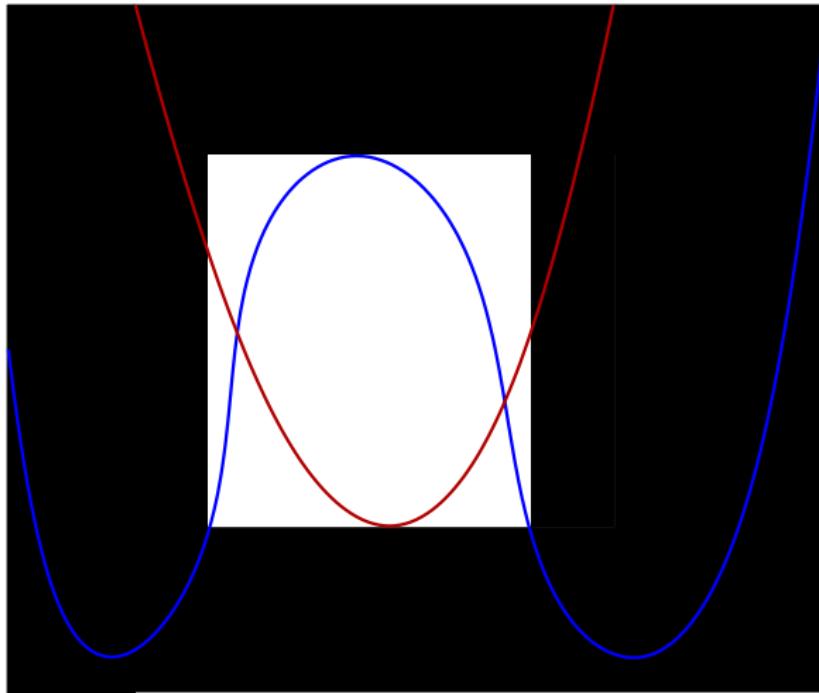


Schéma Brancher & Contracter pour la résolution

Résoudre $\{f_1(x_1, x_2) = 0, f_2(x_1, x_2) = 0\}$

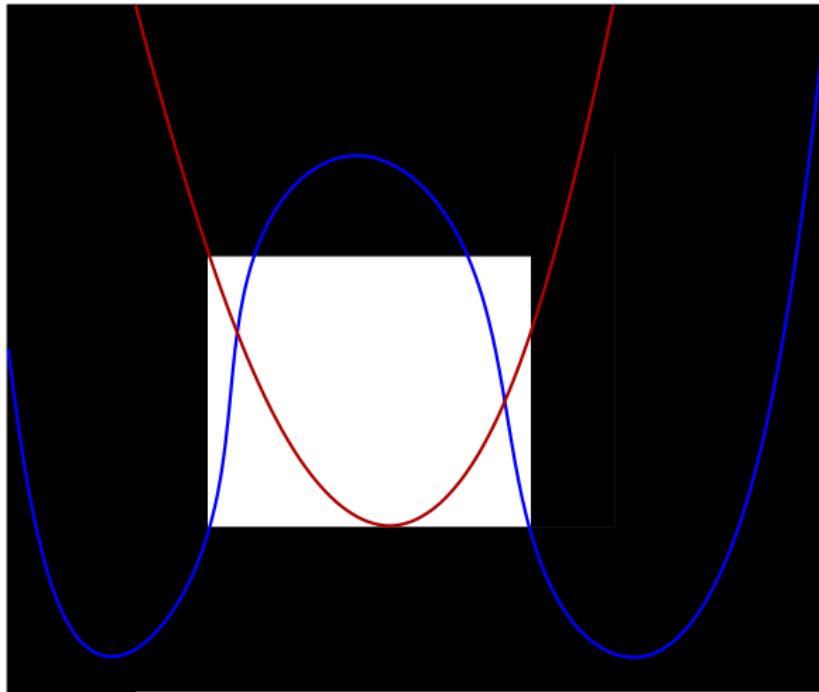


Schéma Brancher & Contracter pour la résolution

Résoudre $\{f_1(x_1, x_2) = 0, f_2(x_1, x_2) = 0\}$

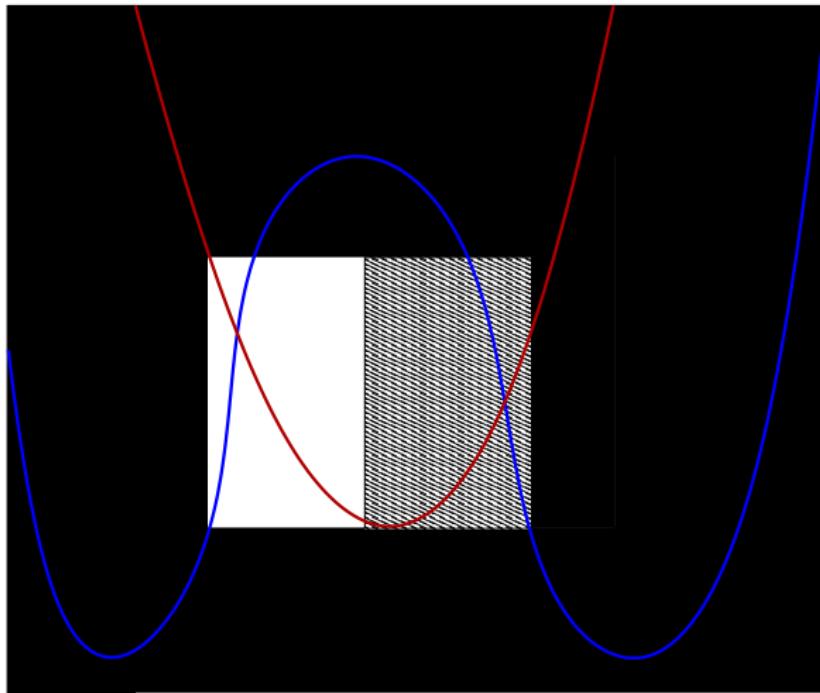


Schéma Brancher & Contracter pour la résolution

Résoudre $\{f_1(x_1, x_2) = 0, f_2(x_1, x_2) = 0\}$

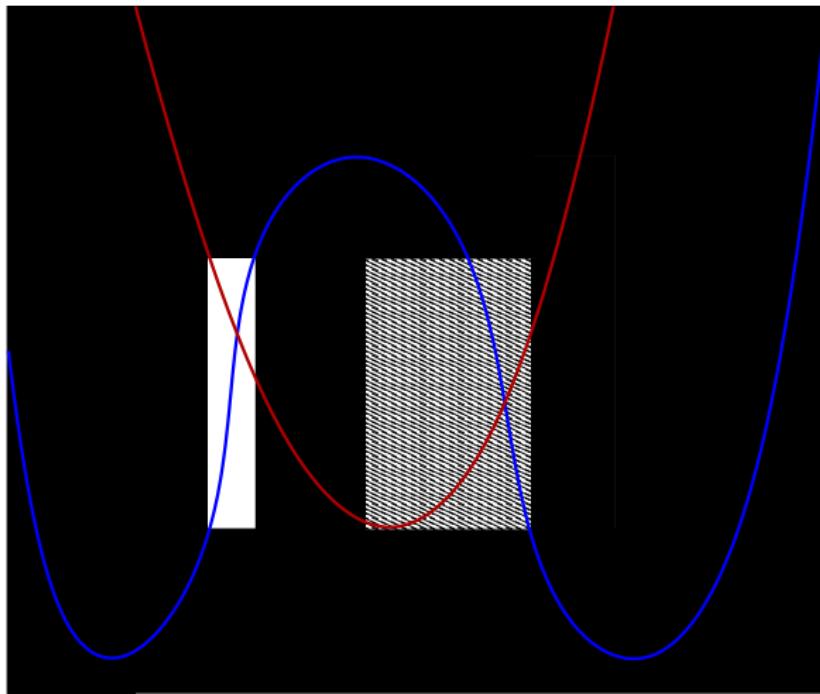


Schéma Brancher & Contracter pour la résolution

Résoudre $\{f_1(x_1, x_2) = 0, f_2(x_1, x_2) = 0\}$

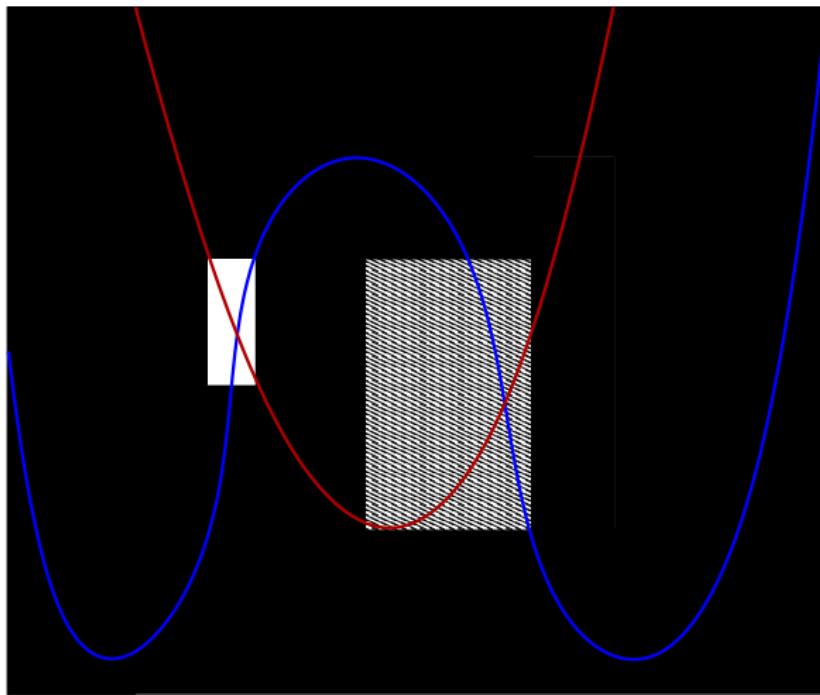


Schéma Brancher & Contracter pour la résolution

Résoudre $\{f_1(x_1, x_2) = 0, f_2(x_1, x_2) = 0\}$

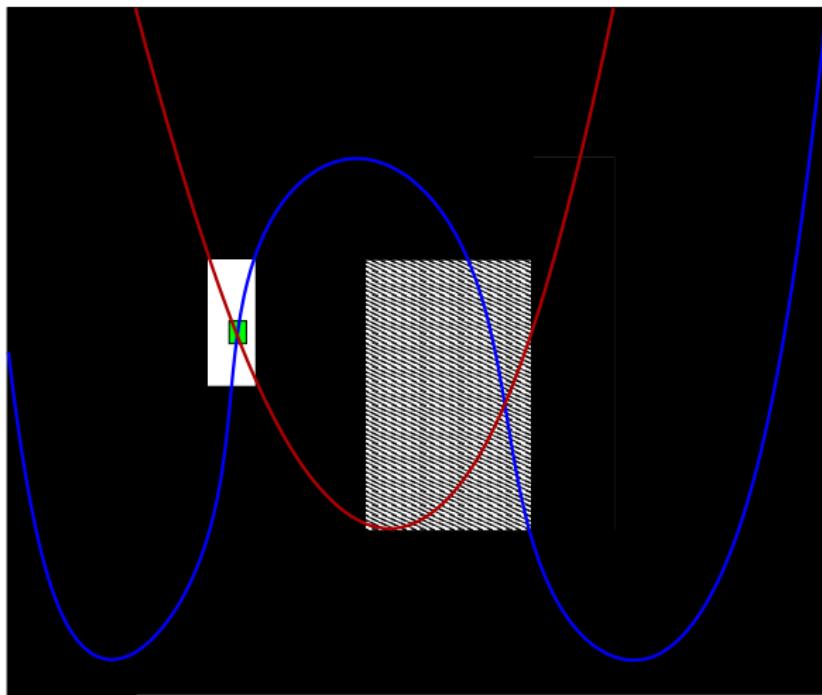


Schéma Brancher & Contracter pour la résolution

Résoudre $\{f_1(x_1, x_2) = 0, f_2(x_1, x_2) = 0\}$

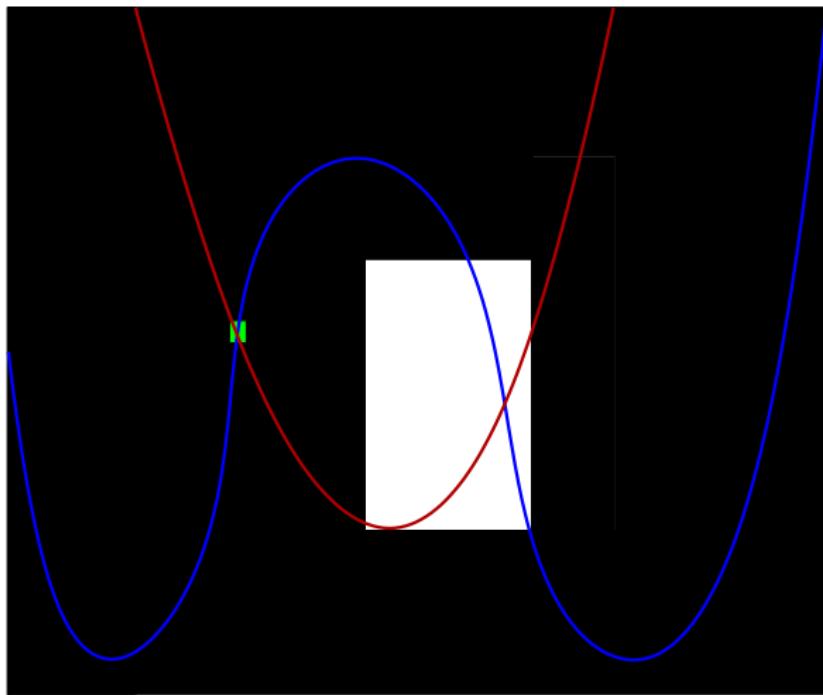
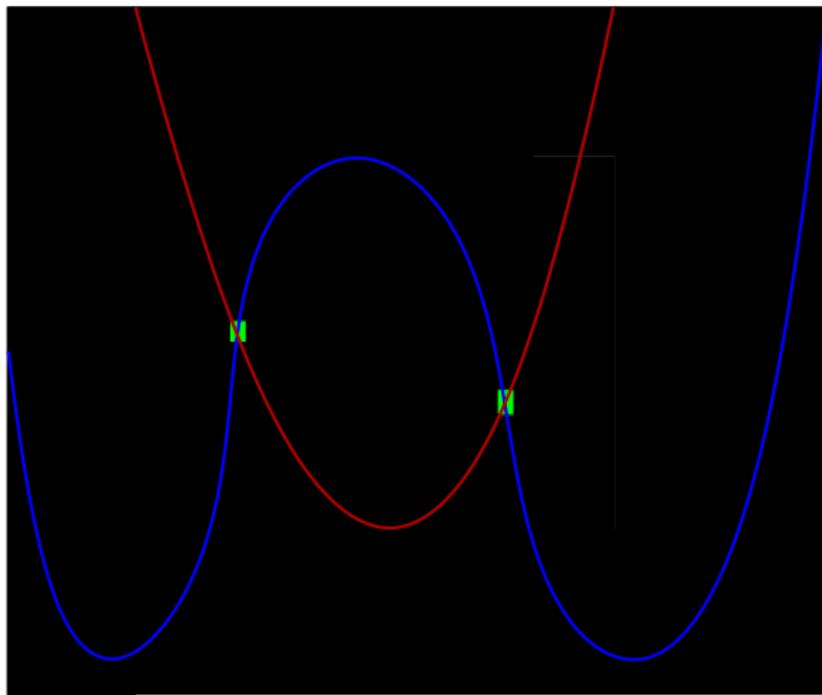


Schéma Brancher & Contracter pour la résolution

Résoudre $\{f_1(x_1, x_2) = 0, f_2(x_1, x_2) = 0\}$



Principaux ingrédients

- ▶ Pré-traitement : remplacement de sous-expressions communes dans le système de contraintes
- ▶ A chacun des (millions de) noeuds : contraction avec ACID(Mohc) suivi de X-Taylor

Remplacement de sous-expressions communes (CS)

$$\begin{aligned}x^2 + y + (y + x^2 + y^3 - 1)^3 + x^3 &= 2 \\ \frac{(y^3 + x^2)(x^2 + \cos(y)) + 14}{x^2 + \cos(y)} &= 8\end{aligned}$$

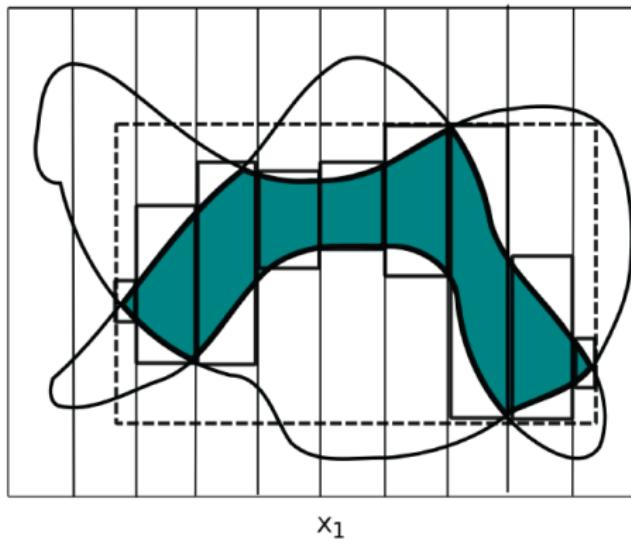
Le nouveau système est :

$$\begin{aligned}v_2 + v_5^3 + x^3 - 2 &= 0 \\ \frac{v_3 v_4 + 14}{v_4} - 8 &= 0\end{aligned}$$

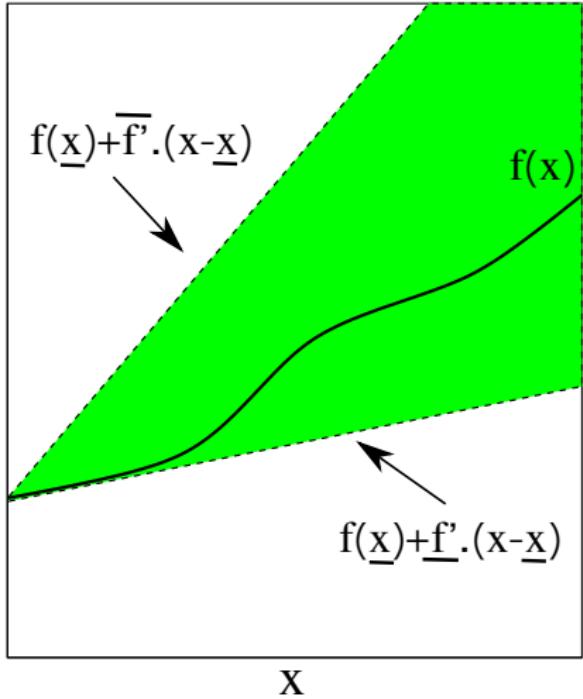
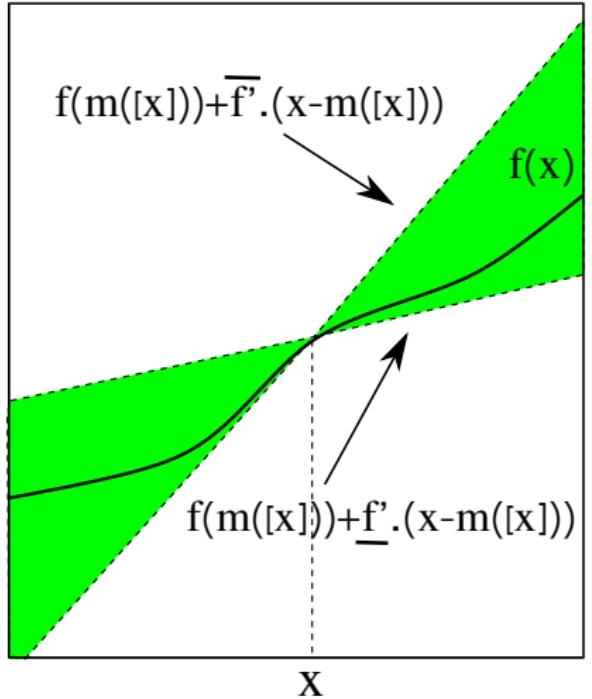
$$\begin{aligned}v_1 &= x^2 \\ v_2 &= y + v_1 \\ v_3 &= v_1 + y^3\end{aligned}$$

$$\begin{aligned}v_4 &= v_1 + \cos(y) \\ v_5 &= v_2 + y^3 - 1 \\ v_5 &= -1 + y + v_3\end{aligned}$$

Sur un “certain nombre” de variables, faire :



Nombre appris dynamiquement pendant la recherche arborescente



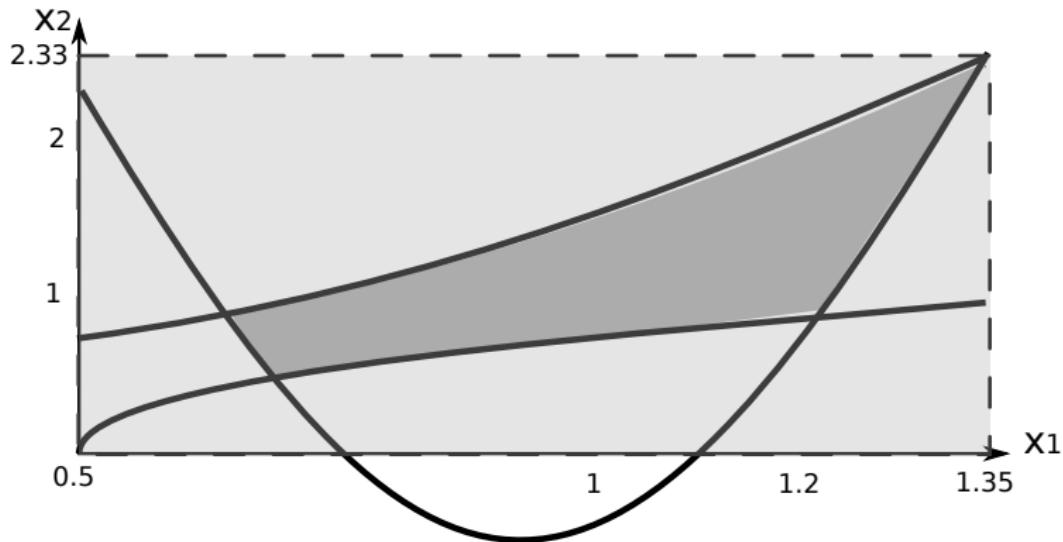
XTaylor

$$x_1^2 - x_2 + 0.5 \leq 0$$

$$-2.5 \sin(4x_1 + 1) + x_2 \leq 2$$

$$\sqrt{x_1 - 0.5} - x_2 \leq 0$$

Domaine = $[0.5, 1.35] \times [0, 2.33]$



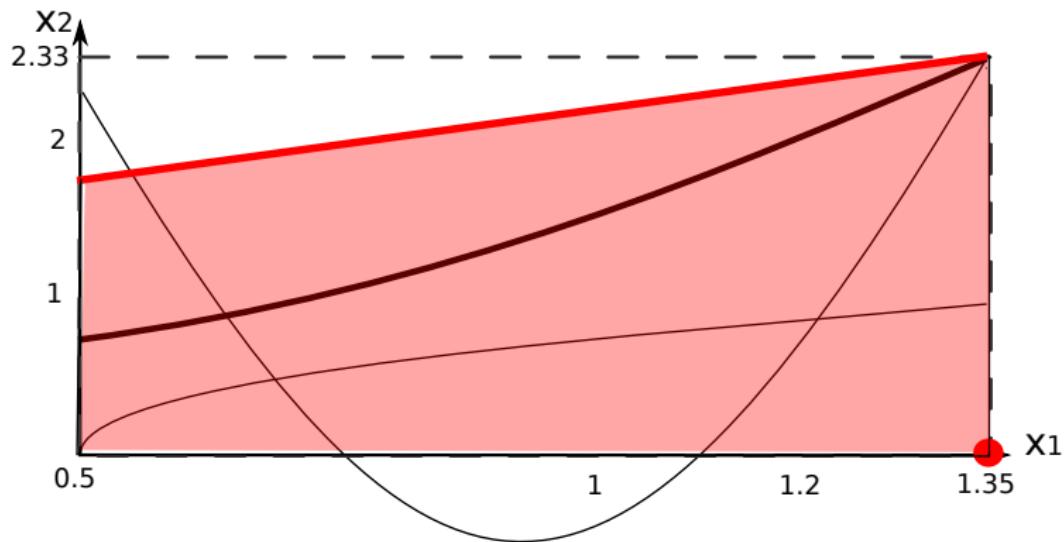
XTaylor

$$x_1^2 - x_2 + 0.5 \leq 0$$

$$-2.5 \sin(4x_1 + 1) + x_2 \leq 2$$

$$\sqrt{x_1 - 0.5} - x_2 \leq 0$$

Domaine = $[0.5, 1.35] \times [0, 2.33]$



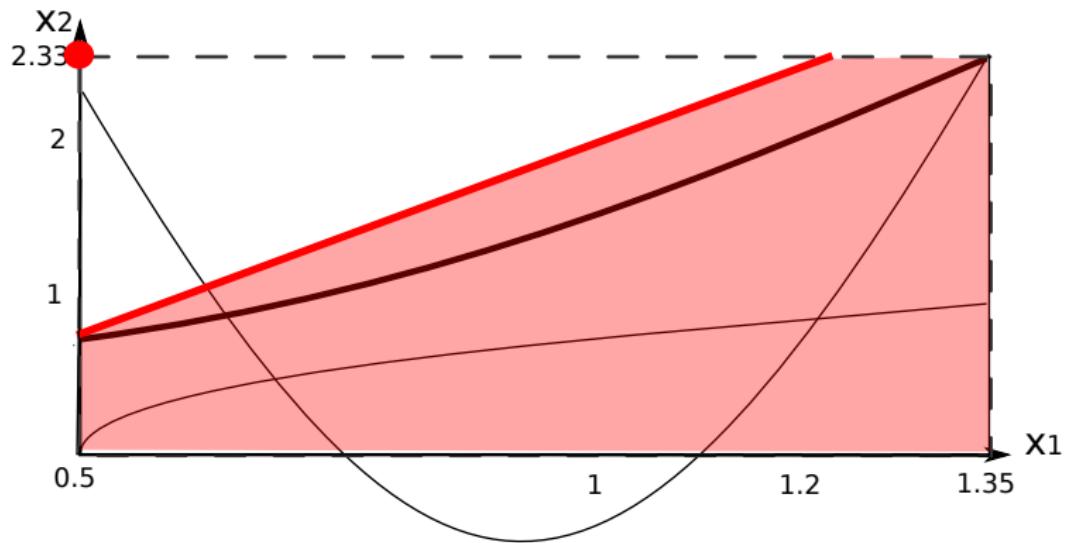
XTaylor

$$x_1^2 - x_2 + 0.5 \leq 0$$

$$-2.5 \sin(4x_1 + 1) + x_2 \leq 2$$

$$\sqrt{x_1 - 0.5} - x_2 \leq 0$$

Domaine = $[0.5, 1.35] \times [0, 2.33]$



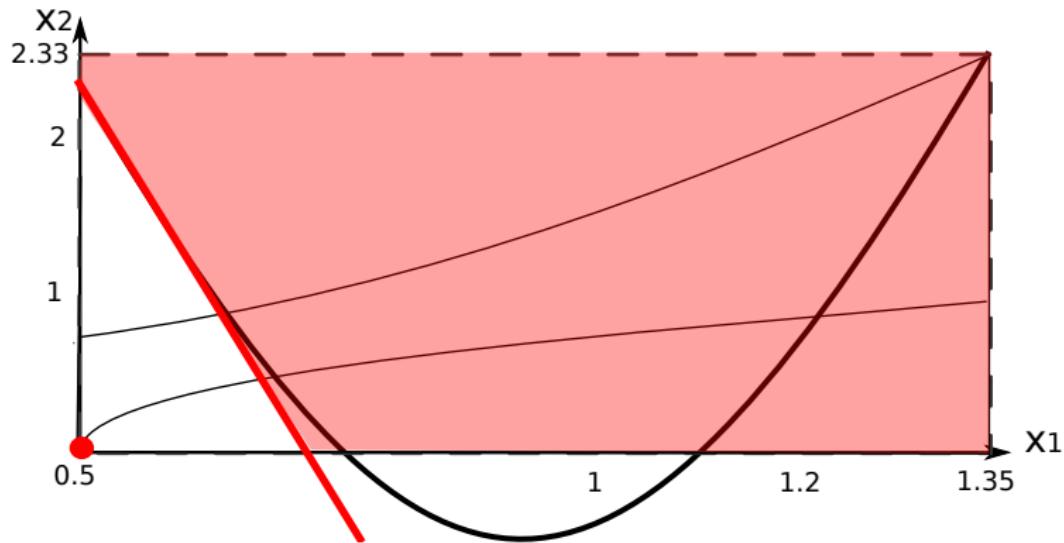
XTaylor

$$x_1^2 - x_2 + 0.5 \leq 0$$

$$-2.5 \sin(4x_1 + 1) + x_2 \leq 2$$

$$\sqrt{x_1 - 0.5} - x_2 \leq 0$$

Domaine = $[0.5, 1.35] \times [0, 2.33]$



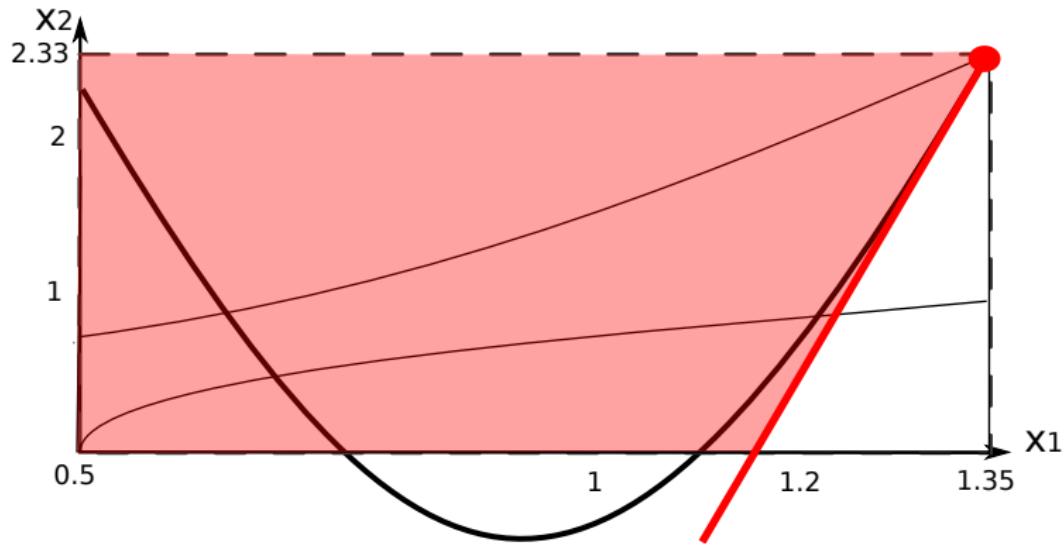
XTaylor

$$x_1^2 - x_2 + 0.5 \leq 0$$

$$-2.5 \sin(4x_1 + 1) + x_2 \leq 2$$

$$\sqrt{x_1 - 0.5} - x_2 \leq 0$$

Domaine = $[0.5, 1.35] \times [0, 2.33]$



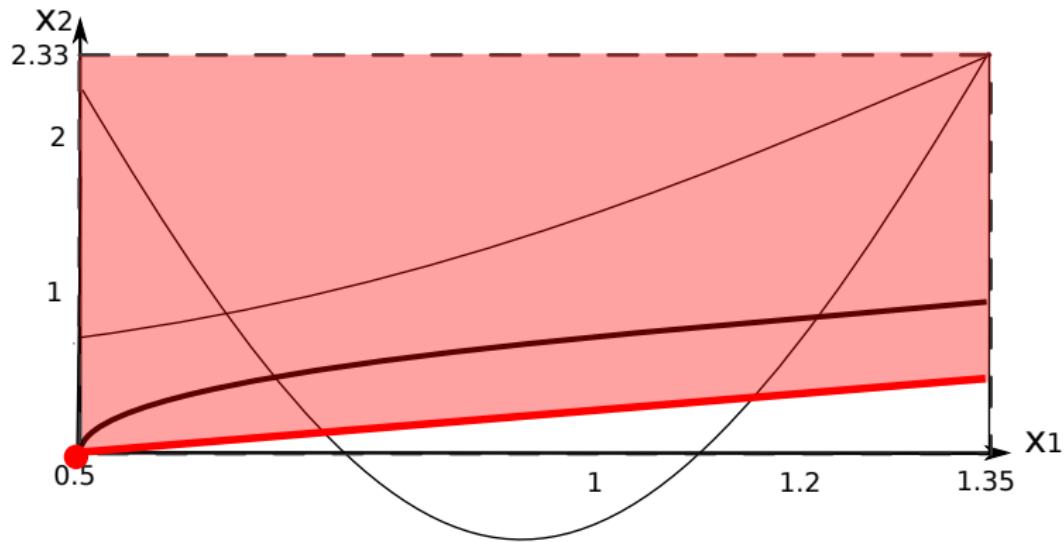
XTaylor

$$x_1^2 - x_2 + 0.5 \leq 0$$

$$-2.5 \sin(4x_1 + 1) + x_2 \leq 2$$

$$\sqrt{x_1 - 0.5} - x_2 \leq 0$$

Domaine = $[0.5, 1.35] \times [0, 2.33]$



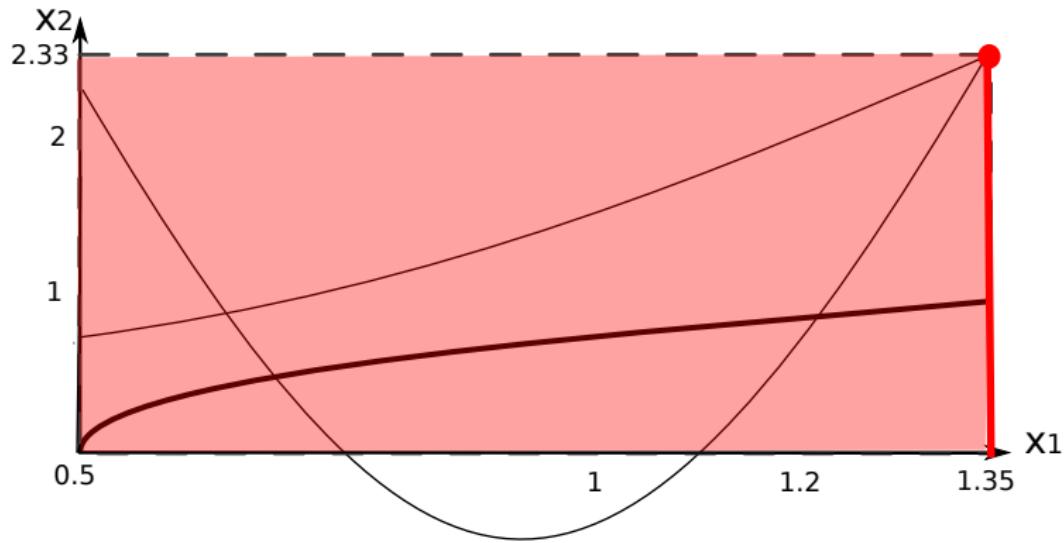
XTaylor

$$x_1^2 - x_2 + 0.5 \leq 0$$

$$-2.5 \sin(4x_1 + 1) + x_2 \leq 2$$

$$\sqrt{x_1 - 0.5} - x_2 \leq 0$$

Domaine = $[0.5, 1.35] \times [0, 2.33]$



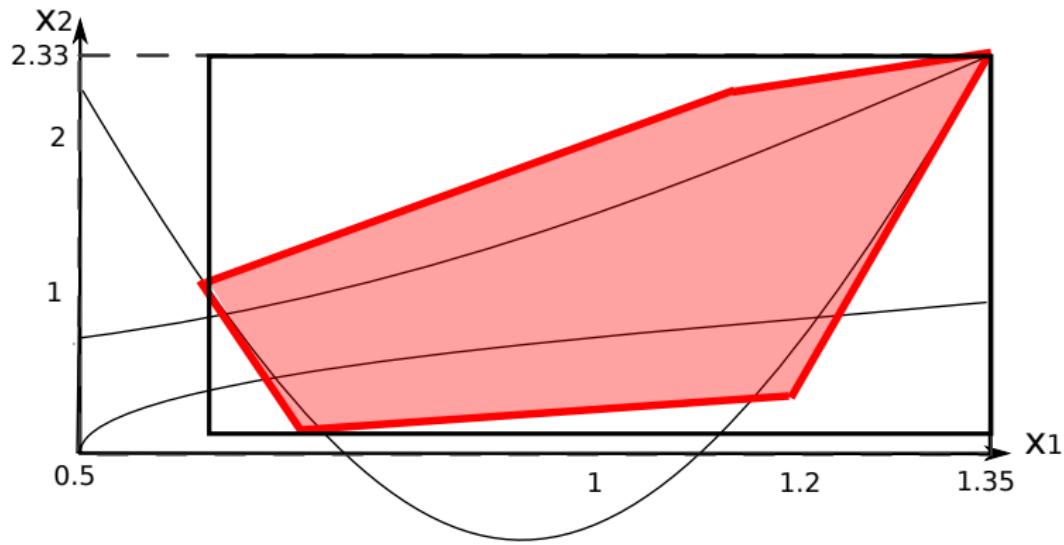
XTaylor

$$x_1^2 - x_2 + 0.5 \leq 0$$

$$-2.5 \sin(4x_1 + 1) + x_2 \leq 2$$

$$\sqrt{x_1 - 0.5} - x_2 \leq 0$$

Domaine = $[0.5, 1.35] \times [0, 2.33]$



Ingrédients supplémentaires pour l'optimisation globale sous contraintes

- ▶ Output : un point flottant x qui ϵ -minimise $f(x)$ s.c. $g(x) \leq 0 \wedge -\epsilon_{eq} \leq h(x) \leq \epsilon_{eq}$.
- ▶ Résolu par un Branch & Bound à intervalles
- ▶ Ajout de deux étapes : lowerbounding et upperbounding (du coût)
- ▶ Lower bounding : ajouter une variable de coût $x_{obj} = f(x)$. Amélioration de \underline{x}_{obj} par contraction \Rightarrow lowerbounding.
- ▶ Upper bounding : chercher un bon point réalisable
 - ▶ sans technique de programmation mathématique classique (relaxation lagrangienne)
 - ▶ en utilisant des algorithmes nouveaux “d'inflation” (extraction de régions intérieures où tous les points sont solution/réalisables)

Résultats

- ▶ IbexOpt est le premier optimiseur à intervalles compétitif avec les meilleurs outils de programmation mathématique (Baron, Antigone, Lindo, Scip, Couenne).

- ▶ Pas de caractérisation de la solution offerte par les outils de programmation mathématique.

- ▶ IbexOpt parfois le meilleur sur les systèmes non polynomiaux (log, sin, division).

Petit exemple où Baron (CMU) est battu

Variables

```
x1                      in [100,10000];  
x2, x3                  in [1000,10000];  
x4, x5, x6, x7, x8 in [10,1000];
```

Minimize x1 + x2 + x3;

Subject to

```
833.33252*x4/x1/x6 + 100/x6 - 83333.333/(x1*x6) <= 1;  
1250*x5/x2/x7 + x4/x7 - 1250*x4/x2/x7 <= 1;  
1250000/(x3*x8) + x5/x8 - 2500*x5/x3/x8 <= 1;
```

```
0.0025*x4 + 0.0025*x6 <= 1;  
-0.0025*x4 + 0.0025*x5 + 0.0025*x7 <= 1;  
-0.01*x5 + 0.01*x8 <= 1;
```

Conclusion

- ▶ Travailler sur des nouveaux algos et enrichir Ibex (pistes diverses et variées)
- ▶ Renforcer la stratégie d'optimisation (hybridation avec méthodes de programmation mathématique)
- ▶ Stratégie d'identification à intervalles : RANSAC++ (INTSAC) : passer de la taille (nombre de paramètres à identifier) 3 à 10 (de 5 à 100 si peu de mesures aberrantes).
- ▶ Avec Coconut : applications et allers-retours discret-continu
- ▶ Modèle hybride : MINLP puis modèle plus ambitieux (CSP+MINLP) avec Coconut
- ▶ Recherches sur les méthodes à intervalles : choix de raison, démarche scientifique

Merci !



Questions ?