Introduction aux méthodes à intervalles pour la résolution de systèmes d'équations non linéaires

Gilles Trombettoni

Université de Nice–Sophia, Equipe COPRIN, INRIA, Sophia Antipolis, France

RAIM, 27 octobre 2009

Outline

Outline

Outline

Qu'est-ce qu'un intervalle ?

Intervalle

Un **intervalle** $[a, b]$ décrit l'ensemble des réels x tels que : $a \leq x \leq b$.
a et b sont des nombres représentables sur un ordinateur (**flottants** ou rationnels).
$\mathbb{IR}$ désigne l'ensemble de tous les intervalles.

Boîte

On désigne par **boîte** un produit cartésien d'intervalles.
En pratique, c'est un vecteur d'intervalles qui définit un espace de recherche dans lequel se trouvent les valeurs des inconnues.

Notations

- $[x]$ désigne l'intervalle associé à la variable x.
- $\underline{x}$ désigne la borne inférieure de $[x]$.
- $\overline{x}$ désigne la borne supérieure de $[x]$.
- $Diam([B]) = \overline{x} - \underline{x}$ est le **diamètre** de la boîte $[B]$. C'est la taille de la plus grande dimension $[x]$ de $[B]$.
- `Mid`$([x])$ retourne le point milieu de l'intervalle $[x]$ (en pratique, un flottant proche...).
- L'opérateur `Hull` d'*enveloppe* permet d'approximer un ensemble S de boîtes par la plus petite boîte (dite **extérieure**) contenant S.

Outline

Extension aux intervalles d'une fonction

L' **extension** aux intervalles d'une fonction ($[f] : \mathbb{IR}^n \to \mathbb{IR}$) permet un calcul conservatif de l'image (évaluation) d'une fonction sur intervalles :

$$[f]([B]) \supseteq \{f(B) \;\; t.q. \;\; B \in [B]\}$$

- Pour les fonctions élémentaires ('+', '$\times$', 'power', 'sinus', 'exp',...) l' **arithmétique des intervalles** calcule rapidement l'image (aux arrondis près).
  Exemples : $[a,b] + [c,d] = [a+c, b+d]$; $sqr([-4,2]) = [0,16]$
- Extension (aux intervalles) **naturelle** $[f]_N$ d'une fonction f composée : l'arithmétique des intervalles est utilisée en chaque opérateur.

  Exemple : en considérant $f(x) = x^3 - 3x^2 + x$, l'image $[y]$ de $[3,4]$ par la fonction $[f]_N$ est : $[y] = [3,4]^3 - 3 \times [3,4]^2 + [3,4]$ $= [27,64] - 3 \times [9,16] + [3,4] = [-18,41]$

Extension aux intervalles d'une fonction (suite)

- Extension de Taylor (ordre 1 ou 2) basée sur un calcul de dérivées...
- Extension de Horner (forme factorisée)
- Extension basée sur la monotonie :

  Exemple :

  1. $f'(x) = 3x^2 - 6x + 1$ d'où $[f']_N([3,4]) = [3,30]$.
  2. Comme $0 \notin [3,30]$, f est monotone croissante par rapport x sur $[x] = [3,4]$.
  3. L'image calculée par monotonie $[f]_M([3,4]) = [f(3), f(4)] = [3,20]$.
- Autres : arithmétique affine, polynômes de Bernstein,...

Causes de non optimalité du calcul de l'image

Toutes les extensions aux intervalles des fonctions produisent en général une **surestimation** de l'image. Calculer l'image optimale d'un système polynomial est un problème NP-difficile. Trois causes de non optimalité :

- les **arrondis** liés aux calculs sur les flottants ;
- le manque de **continuité** des fonctions. Exemple :
  $f(x) = (\frac{1}{x})^2$, avec $[x] = [-1, 1]$
  $\frac{1}{[-1,1]} = Hull([-\infty, -1] \cup [1, +\infty]) = [-\infty, +\infty]$
  D'où : $(\frac{1}{[-1,1]})^2 = [0, +\infty]$
  Or, l'image optimale $[1, +\infty]$ peut s'obtenir en faisant un point de choix sur $[-\infty, -1]$ et $[1, +\infty]$.
- les variables qui apparaissent **plusieurs fois** dans l'expression.

Problème des occurrences multiples de variables

Exemple : $[x] - [x] = [a - b, b - a] \neq [0, 0]$

$(\mathbb{IR}, +)$ et $(\mathbb{IR} \setminus \{[0, 0]\}, \times)$ ne sont pas des groupes !

Si f est monotone par rapport à x dans une boîte donnée, alors le problème de dépendance lié à cette variable disparaît.

Outline

Schéma combinatoire (naïf) de résolution : bissection + évaluation

$$\begin{aligned} f: \quad & \mathbb{R}^2 \longrightarrow \mathbb{R}^2 \\ & x \mapsto (x_1^2 + x_2^2 - 4^2,\ (x_1 - 4)^2 + x_2^2 - 6^2)^T \end{aligned}$$

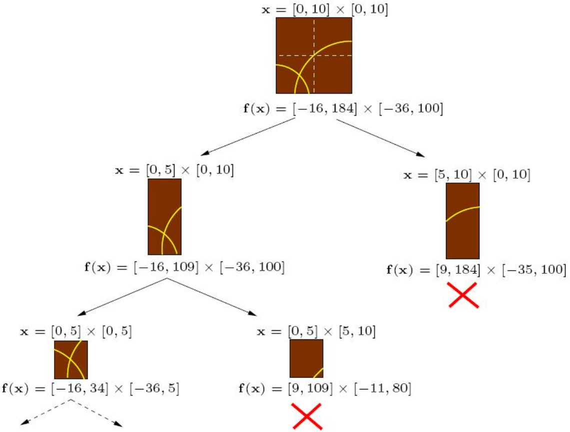


Schéma général de résolution

Les méthodes à intervalles permettent de trouver **toutes** les solutions d'un système de contraintes sur les réels.

Schéma général : contraction + bissection

1. $L \leftarrow \{[X]\}$ /* On part d'une boîte initiale $[X]$ */
2. Tant que L est non vide faire :
    1. Choisir la première boîte $[B]$ de L (et l'enlever de L).
    2. **Contracter** $[B]$ et essayer de **garantir** l'existence et/ou de l'unicité d'une solution dans $[B]$.
    3. Si $[B] = \emptyset$, alors pas de solution dans $[B]$.
    4. Si $[B] \neq \emptyset$ et $Diam([B]) < \epsilon$, alors $[B]$ est “solution” (garantie ou non).
    5. Si $[B] \neq \emptyset$ et $Diam([b]) > \epsilon$, alors **Bissection**: $[B]$ est découpée sur une dimension (variable) en deux sous-boîtes $[B_g]$ et $[B_d]$.
    6. $L \leftarrow L \cup \{[B_g]\} \cup \{[B_d]\}$

Schéma général de résolution (suite)

Remarques

- Définition de **contraction** : réduction de la boîte courante sur les bornes sans perte de solution.
- Toutes les stratégies de résolution basées sur les intervalles trouvent un **sur-ensemble** des solutions (en utilisant des algorithmes de contraction fiables).

Trois types de contracteurs

- Relaxation linéaire ou convexe (optimisation globale robuste).
- Extension aux intervalles des algorithmes d'analyse numérique (analyse par intervalles).
- Propagation de contraintes et consistance forte (programmation par contraintes).

Schéma pour l'optimisation globale sous contraintes

Schéma complexifié pour minimiser une fonction objectif sous contraintes.
En plus des contracteurs, l'algorithme de **branch and bound** met à jour au cours de la recherche un minorant l et un majorant u de l'objectif jusqu'à ce que $u - l < \epsilon$:

- Majorant souvent obtenu par recherche locale. Certification coûteuse (analyse par intervalles).
- Minorant obtenu par approximation linéaire (ou convexe) conservative de la fonction objectif : la meilleure solution de l'approximation fournit une borne inférieure de l'objectif.

Schéma général de résolution (exemple)

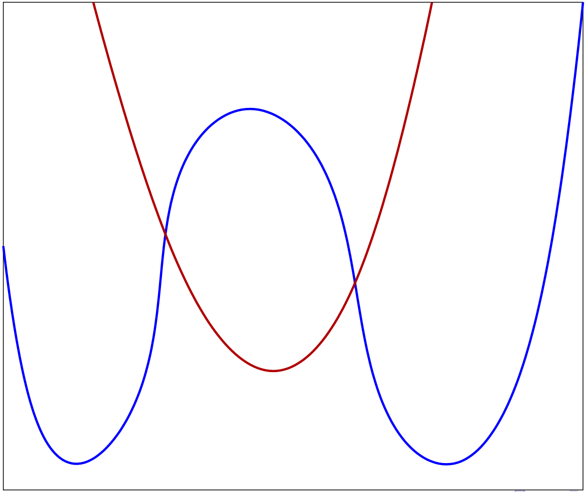


Schéma général de résolution (exemple)

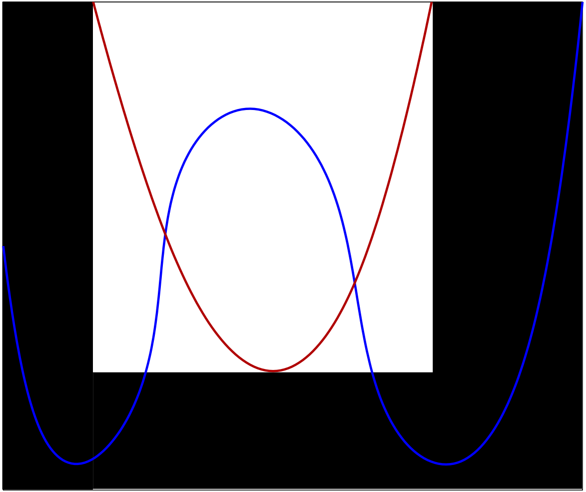


Schéma général de résolution (exemple)

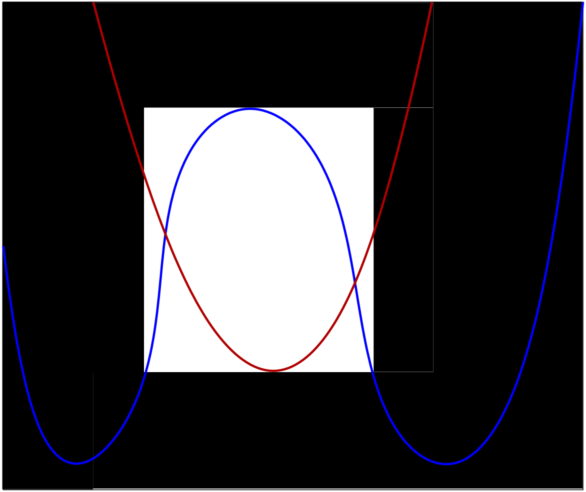


Schéma général de résolution (exemple)

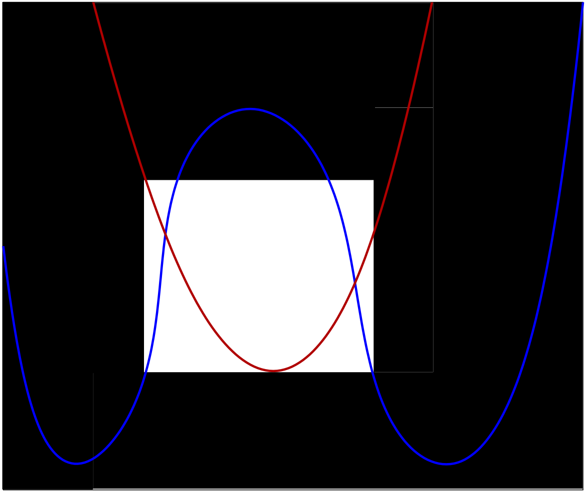


Schéma général de résolution (exemple)

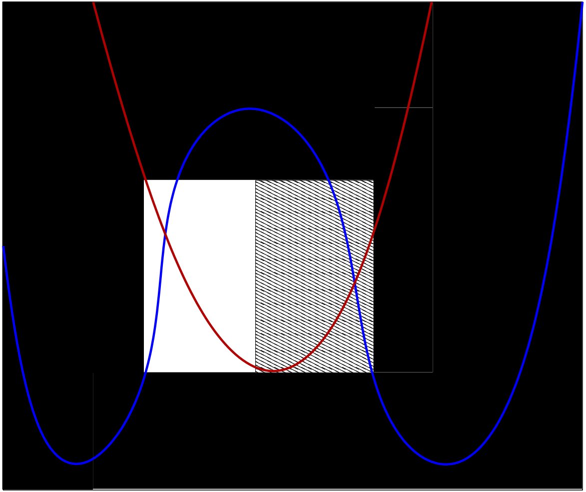


Schéma général de résolution (exemple)

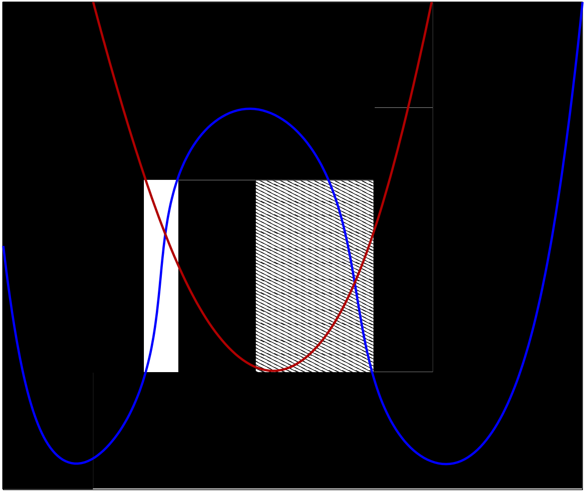


Schéma général de résolution (exemple)

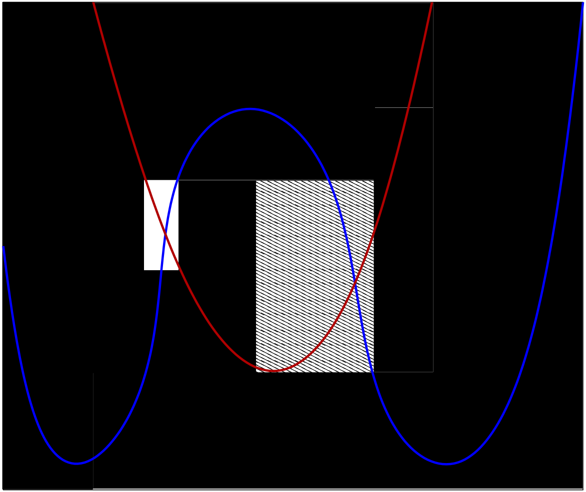


Schéma général de résolution (exemple)

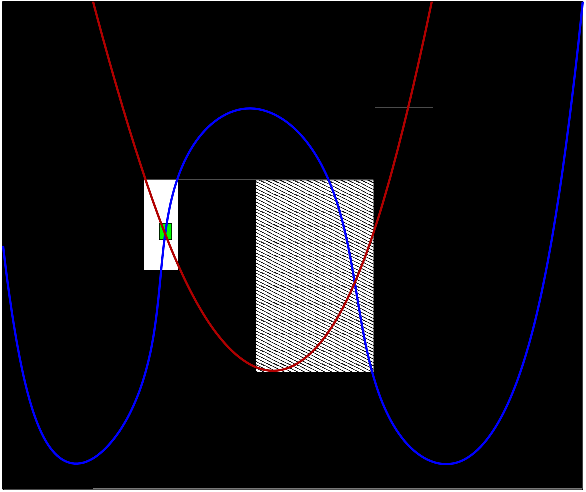


Schéma général de résolution (exemple)

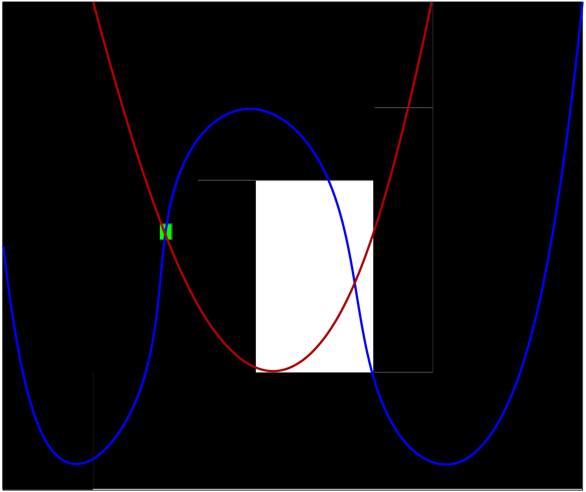
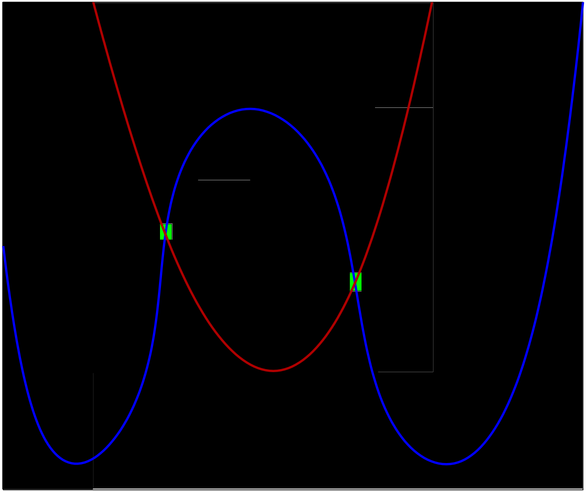


Schéma général de résolution (exemple)



Outline

Outline

Principales heuristiques de bissection

Dans l'arbre de recherche, le choix de la prochaine variable à bissecter est très important. Trois heuristiques sont très employées :

- à **tour de rôle** ! (ce qui souligne l'importance de n'oublier aucune variable) ;
- le **plus large intervalle** d'abord (ce qui souligne encore l'importance de n'oublier personne) ;
- la fonction **smear** (Kearfott 1990) :
  - pour chaque couple (f, x), dans la boîte $[B]$ courante : calculer $smear(f, x) = |\frac{\partial f}{\partial x}([B])| \times Diam([x])$;
  - pour une variable x donnée : $smear(x) = \sum_j (smear(f_j, x))$ (ou $Max_j(smear(f_j, x))$) ;
  - bissecter la variable de plus grand impact.

Outline

Opérateurs de l'analyse par intervalles

Adaptation aux intervalles des algos d'analyse numérique (Newton).

Principe

- On part d'une boîte initiale $[X]$.
- On calcule jusqu'à point-fixe : $[X] \leftarrow [X] \cap N([X])$.
- Si $N([X]) \subset [X]$, alors garantie de solution unique dans $[X]$.

On définit $N([X]) := \dot{X} + [Y_s]$, où :

- $\dot{X}$ est un point de $[X]$;
- $[Y_s]$ est la boîte obtenue par résolution du système linéarisé $[A][Y] + [B] = 0$.
- Par exemple, $[A]$ est la matrice jacobienne du système F (chaque élément est l'intervalle $\frac{\partial [f_j]}{\partial x_i}$) ;
  $\dot{X} = Mid([X])$; $[B] = F(\dot{X})$; $[Y] = [X] - \dot{X}$.

Opérateurs de l'analyse par intervalles (suite)

Newton intervalles comprend en fait un grand ensemble de variantes qui diffèrent par :

- l'utilisation de dérivées ou de pentes ;
- la matrice utilisée : jacobienne, de Hansen, etc ;
- l'utilisation de calcul formel sur la matrice ;
- la linéarisation : Newton, Krawczyck, Borsuc, Kantorovitch (dérivées secondes), etc ;
- la résolution du système linéaire : inversion de matrice, Gauss-seidel, Hansen-Bliek, LU, etc ;
- le préconditionnement pour rendre la contraction effective.

Outline

Principe de la relaxation linéaire

Principe : Approximer les contraintes non-linéaires (et la fonction objectif) par un ensemble d'inégalités linéaires englobantes traitées généralement par un algorithme du simplexe.

Relaxation

- Le système linéarisé contient plus de solutions que le système non-linéaire initial.
- Relaxer linéairement l'objectif à minimiser permet de trouver un minorant (un point plus bas) à la valeur de l'optimum global.

Relaxation rigoureuse

- La linéarisation doit être conservative (arrondis sur des nombres flottants) pour ne pas perdre de solution (ou l'optimum global).
- Les algorithmes du simplexe disponibles sont non rigoureux. Correction par un post-traitement (Neumaier, Shcherbina 2004).

Relaxation du système : exemples

L'algorithme `Quad` (Lebbah, Rueher, Michel) :

- les termes bilinéaires et quadratiques sont linéarisés ;
- Le sous-système linéaire résultant est contracté par du “LP narrowing” :
  Chaque borne de chaque variable est modifiée par un appel à un simplexe.
- Propagation de contraintes sur l'ensemble du système (incluant les contraintes non polynomiales).

Autre linéarisation existante : implanter les fonctions élémentaires à l'aide de l'*arithmétique affine*.

Outline

Deux grands types de contracteurs de PPCI

- **Propagation de contraintes**:
  La boîte est contractée vis à vis de chaque contrainte **individuellement** (procédure *revise*) :
  - HC4-Revise ;
  - Box-Revise (BoxNarrow) ;
  - ...

  L'intersection entre boîtes individuelles est mise en œuvre incrémentalement par une procédure de **propagation de contraintes** de type AC3.
- **Contracteurs plus puissants** : algorithme de 3B-consistance.

Propagation de contraintes

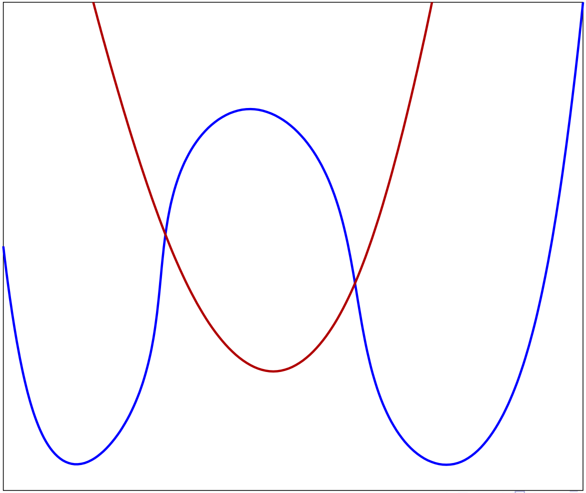
Un système $(\mathcal{C}, \mathcal{X}, [\mathcal{B}])$ est **2B-cohérent** si la boîte $[\mathcal{B}]$ est optimale pour chaque contrainte de $\mathcal{C}$ prise individuellement.

But des algorithmes de **propagation de contraintes** : contracter la boîte courante vis à vis d'une seule contrainte, à tour de rôle.

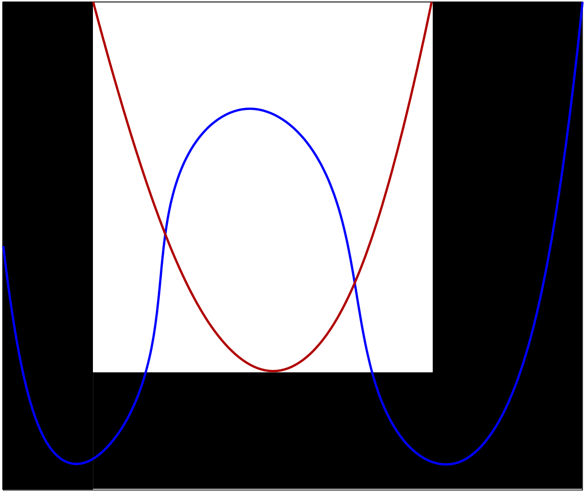
Principe : Au départ, on empile toutes les contraintes dans une *queue de propagation* Q. Jusqu'à ce que Q devienne vide :

1. sélection dans Q d'une contrainte c (et élimination) ;
2. application sur c d'une procédure de contraction/révision qui réduit (optimalement ?) les intervalles des variables V de c ;
3. propagation : $\forall v \in V$, si $[v]$ est (suffisamment) réduit, on ajoute dans Q chaque contrainte c' impliquant v ($c' \neq c$) (si c' ne se trouve pas déjà dans Q).

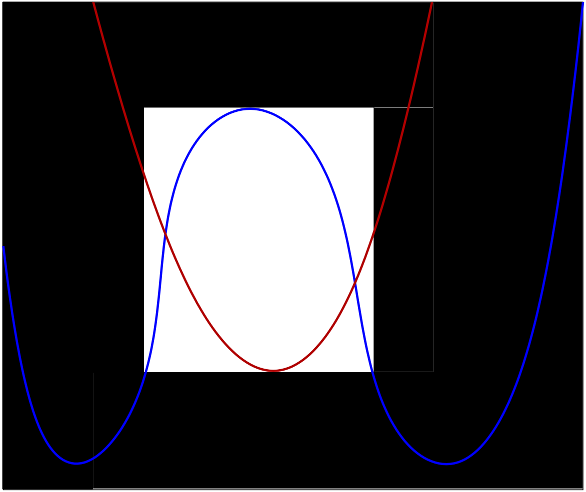
Exemple de propagation de contraintes



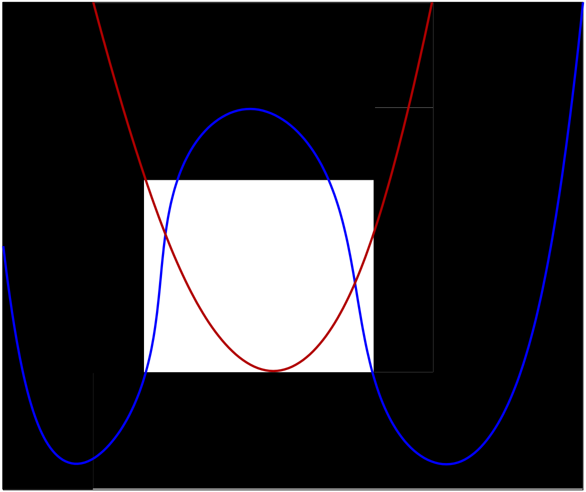
Exemple de propagation de contraintes



Exemple de propagation de contraintes



Exemple de propagation de contraintes



Algorithme `HC4-Revise`

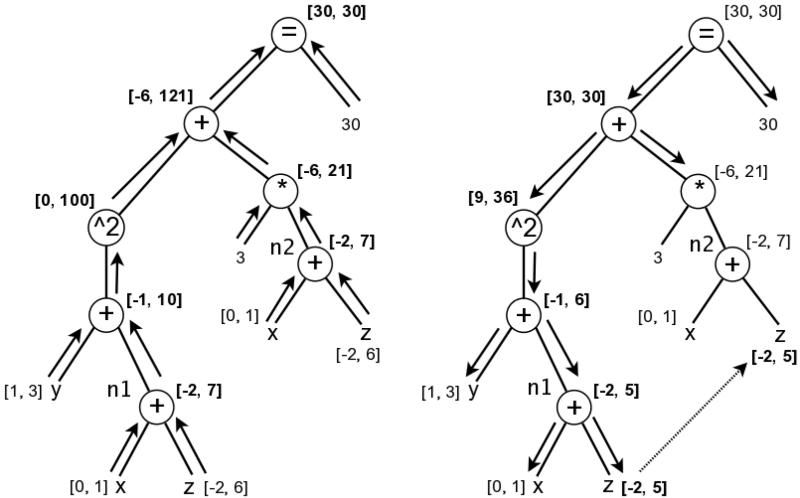
L'algorithme `HC4-Revise` contracte la boîte courante vis à vis d'une contrainte *c*.

Principe de `HC4-Revise` : contracter chaque **occurrence** de variable en l'isolant dans *c*.

Implantation de `HC4-Revise`

- Double-parcours de l'arbre de syntaxe représentant *c*.
- Synthèse : **évaluation** (sur intervalles) en chaque noeud de l'arbre.
- Héritage : **projection élémentaire** en chaque noeud de l'arbre.

Exemple : $(x + y + z)^2 + 3(x + z) = 30$



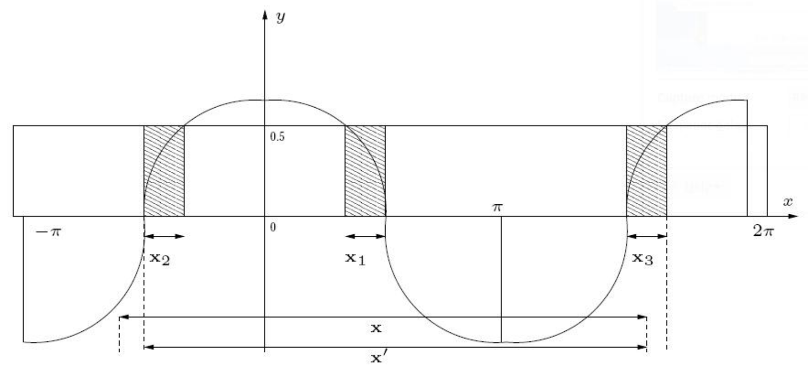
Fonctions de projection élémentaires

- Toute contrainte s'obtient par composition de fonctions **élémentaires**.
- Contraintes élémentaires :

$y = x$	$y = \exp(x)$	$y = \cos(x)$	$y = \arccos(x)$
$y = x + z$	$y = \ln(x)$	$y = \sin(x)$	$y = \arcsin(x)$
$y = x - z$	$y = \log(x)$	$y = \tan(x)$	$y = \arctan(x)$
$y = x \times z$	$y = \mathrm{sqr}(x)$	$y = \cosh(x)$	$y = \mathrm{arccosh}\,(x)$
$y = x/z$	$y = \sqrt{x}$	$y = \sinh(x)$	$y = \mathrm{arcsinh}\,(x)$
$y = x^a$	$y = 1/x$	$y = \tanh(x)$	$y = \mathrm{arctanh}\,(x)$

- Pour les contraintes élémentaires, on sait calculer la boîte optimale en utilisant les **fonctions de projection** (appelées aussi **fonctions inverses**).

Fonctions de projection élémentaires (ex : $y = cos(x)$)

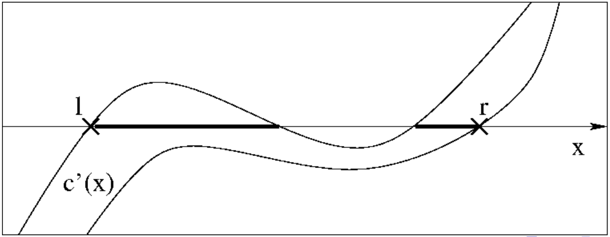


Les fonctions de projection primitives sont toutes strictement monotones par morceaux sur leur domaine de définition, et sont inversibles par morceaux.
$\Rightarrow$ On peut implémenter les procédures correspondantes en utilisant les bornes des opérandes, en étudiant les changements de sens de variation et en gérant les bornes flottantes.

Algorithme Box-Revise

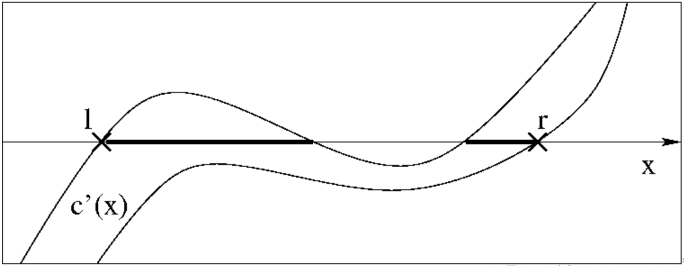
Principe :

- Considérer la contrainte univariée “épaisse” $c'(x) = f(x, [y_1], ..., [y_k]) = 0$: toutes les variables sauf x sont remplacées par leur intervalle courant.
- Réduire $[x]$ en calculant le zéro de c' le plus petit (l) et le plus grand (r).



Algorithme Box-Revise

- Découper $[x]$ de gauche à droite en bandes/tranches $[x_i]$ de largeur ϵ.
- Si $0 \notin [f]([x_i], [y_1], ..., [y_k])$, alors bande $[x_i]$ éliminée et on passe à la bande suivante.
- Optimisations : découpage dichotomique et utilisation d'un Newton intervalles univarié.



Propagation de contraintes optimale ?

Souci : Pour **une** contrainte polynomiale $c = f(X) = 0$, calculer la boîte optimale (2B-cohérente) est NP-difficile.
Causes de non optimalité : arrondis sur les flottants, non continuité des fonctions et occurrences multiples de variables.
En pratique :

- HC4-Revise (une variante en fait...) calcule la boîte optimale (aux arrondis près) quand c ne contient **aucune** occurrence multiple de variables.
- Box-Revise est plus coûteux mais contracte mieux que HC4-Revise quand c contient **une** seule variable qui apparaît plusieurs fois.
- Un nouvel algorithme Mohc-Revise est optimal quand **toutes** les variables $x \in X$ qui apparaissent plusieurs fois sont **monotones** (f est monotone par rapport à x).

Remplacement des sous-expressions communes (CS)

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

Le nouveau système est :

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

$$\begin{aligned}
v_1 &= x^2 & \qquad v_4 &= v_1 + cos(y) \\
v_2 &= y + v_1 & \qquad v_5 &= v_2 + y^3 - 1 \\
v_3 &= v_1 + y^3 & \qquad v_5 &= -1 + y + v_3
\end{aligned}$$

Remplacement des sous-expressions communes (CS)

Remplacer des CS par une même variable auxiliaire revient à l'ajout de contraintes redondantes et donc apporte de la **contraction additionnelle** avec de la propagation de contraintes.

Vrai seulement quand :

- l'opérateur correspondant est non monotone ou non continu : x^a (a entier positif pair), $\mathtt{sinus}$, $\frac{1}{x}$, etc ;
- l'opérateur est n-aire : $+$, $\times$.

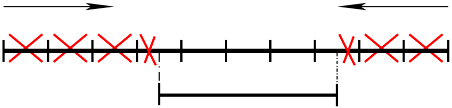
Gains de performance possibles de plusieurs ordres de grandeur.

Outline

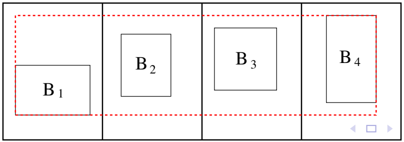
Algorihme 3B-consistance

But principal : réduire l'intervalle d'une variable x_i

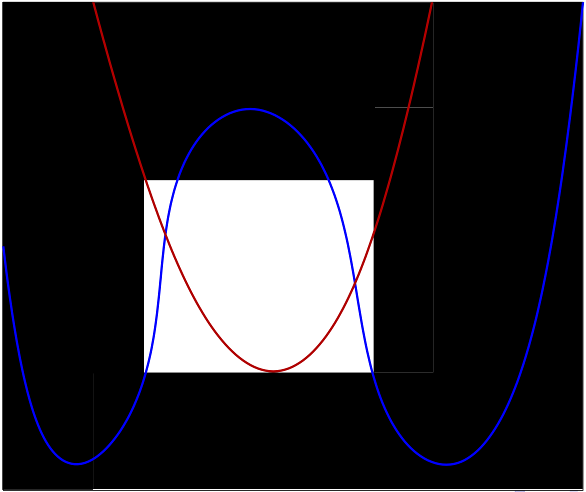
L'intervalle $[x_i]$ est 3B-consistant (c'est-à-dire, ne peut être plus réduit) si on ne peut pas éliminer sa borne gauche ni sa borne droite avec `HC4`.
L'algorithme de 3B-consistance effectue des "points de choix polynomiaux".



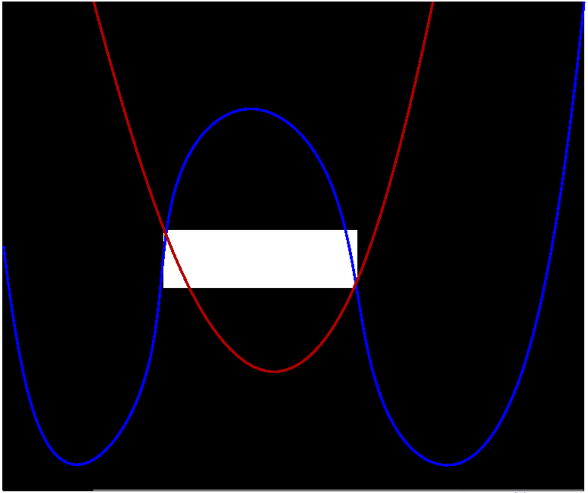
Nous avons apporté plusieurs améliorations à cet algorithme, dont :



Exemple de 3B-consistance



Exemple de 3B-consistance



Outline

Problèmes avec continua de solutions

Il existe des systèmes avec une infinité de solutions :

- Systèmes d'inégalités ;
- **AE-systèmes** (*All-Exist systems*) qui contiennent des fonctions paramétrées (par U et V) :
  $(\forall U \in [U])\ (\exists V \in [V]) \quad f(X, U, V) = 0$

Deux ingrédients essentiels pour aborder ces problèmes :

- **Arbres de recherche et/ou** pour la bissection des paramètres.
- Identifier des **boîtes intérieures** qui ne contiennent que des solutions :
  $(\forall f \in F)(\forall X \in [X])(\forall U \in [U])\ (\exists V \in [V]) \quad f(X, U, V) = 0$

Identification de boîtes intérieures

Il s'agit de l'opération duale de la contraction (de boîtes *extérieures*). Plusieurs approches selon des conditions :

- Système classique d'inégalités ($c_1 \wedge ... \wedge c_n$) : Si une boîte $[B]$ est éliminée à la fois par $\neg c_1$, $\neg c_2$,... et $\neg c_n$, alors $[B]$ est intérieure.
- Système d'inégalités quantifiées universellement ($\forall U \in [U] f(X, U) \leq 0$) : Heuristique de bissection des paramètres U et test de contraintes individuelles.
- Isolation possible de paramètre existentiel dans chaque équation ($f(X, v) \equiv g(X) = v$) : Si $G([X]) \subseteq [V]$, alors $[X]$ est intérieure.
- Nombre de paramètres existentiels égal au nombre de variables : Newton intervalles multivarié (épais) sur une permutation variable-paramètre.
- Système linéaire en les paramètres existentiels : "résolution" du système linéaire.

Identification de boîtes intérieures (suite)

- Si chaque composante de V n'apparaît au plus qu'une fois dans F :
  Si $0 \subseteq F(dual[X], dual[U], V)$, alors $[X]$ est intérieure (intervalles généralisés).

- Approximation extérieure (contraction) : Newton intervalles généralisé.

Utilité des intervalles généralisés ?

Une généralisation des intervalles sont les intervalles modaux – $(\forall, [x])$ ou $(\exists, [x])$ – et les intervalles généralisés (ex : $[2, 1]$ dual de $[1, 2]$) qui corrigent des défauts algébriques de $\mathbb{IR}$.

Intervalles modaux et généralisés ne semblent pas utiles pour la résolution de systèmes quantifiés **non-linéaires**.

Espoir pour systèmes quantifiés linéaires ?

Exemple : méthode du pivot de Gauss (LU) généralisée : faire apparaître un zéro dans le coefficient $[a_{12}]$ de la matrice quand $[a_{11}]$ est le pivot :

$$[a'_{12}] = [a_{12}] - \frac{dual[a_{12}] \times dual[a_{11}]}{[a_{11}]} = 0$$

Bibliographie

- Ramon E. Moore, R. Baker Kearfott, Michael J. Cloud, *"Introduction to Interval Analysis"*, SIAM, 2009
- Eldon Hansen, *"Global Optimization using Interval Analysis"*,
- Arnold Neumaier, *"Interval Methods for Systems of Equations"*, Cambridge University Press, 1990
- Pascal Van Hentenryck, Laurent Michel, Yves Deville, *"Numerica: A Modeling Language for Global Optimization"*, MIT Press, 1997.
- Frédéric Benhamou et Laurent Granvilliers, *"Continuous and Interval Constraints"*, chapitre 16 du livre *"Handbook of Constraint Programming"*, Elsevier, 2006.
- **Luc Jaulin, Michel Kieffer, Olivier Didrit, Eric Walter, "Applied Interval Analysis", Springer, 2001.**

Différentes communautés travaillant sur les intervalles

Plus de 200 chercheurs s'intéressent aux intervalles, mais une minorité se préoccupe des systèmes non convexes. De plus, ils sont éparpillés dans différentes communautés scientifiques.

- arithmétique sur intervalles (et flottants) ;
- analyse par intervalles (analyse numérique) ;
- optimisation globale robuste (programmation mathématique, recherche opérationnelle) ;
- programmation par contraintes (informatique, RO) ;
- “neo applicatifs” : Chercheurs (en France, notamment de la 61^e section !) appliquant les méthodes à intervalles dans divers domaines comme la robotique, l'automatique (robuste), le traitement du signal, chimie, etc ;

  Communauté émergente !