

Certified linear algebra Introduction to Interval Arithmetic

Cours de recherche master informatique

19 October 2009

References for today's lecture

- ▶ R. Moore: *Interval Analysis*, Prentice Hall, Englewood Cliffs, 1966.
- ▶ A. Neumaier: *Interval methods for systems of equations*, CUP, 1990.
- ▶ E. Hansen and W. Walster: *Global optimization using interval analysis*, MIT Press, 2004.
- ▶ R.B. Kearfott: *Rigorous global search: continuous problems*, Kluwer, 1996.
- ▶ V. Kreinovich, A. Lakeyev, J. Rohn, P. Kahl: *Computational Complexity and Feasibility of Data Processing and Interval Computations*, Dordrecht, 1997.
- ▶ L.H. Figueiredo, J. Stolfi: *Affine arithmetic* <http://www.ic.unicamp.br/~stolfi/EXPORT/projects/affine-arith/>.
- ▶ *Taylor models arith.:* M. Berz and K. Makino, N. Nedialkov,

Historical remarks

Who invented Interval Arithmetic?

- ▶ Ramon Moore in 1962 - 1966 ?
- ▶ T. Sunaga in 1958 ?
- ▶ M. Warmus in 1956?
- ▶ P.S. Dwyer in 1951 (closed intervals only)?
- ▶ Rosalind Cecil Young in 1931 ?
- ▶ ...
- ▶ Archimedes: 3rd century BC, to enclose π !

Cf. <http://www.cs.utep.edu/interval-comp/>, click on *Early papers*.

Historical remarks

Who invented Interval Arithmetic?

- ▶ Ramon Moore in 1962 - 1966 ?
- ▶ T. Sunaga in 1958 ?
- ▶ M. Warmus in 1956?
- ▶ P.S. Dwyer in 1951 (closed intervals only)?
- ▶ Rosalind Cecil Young in 1931 ?
- ▶ ...
- ▶ Archimedes: 3rd century BC, to enclose π !

Cf. <http://www.cs.utep.edu/interval-comp/>, click on *Early papers*.

Historical remarks

Who invented Interval Arithmetic?

- ▶ Ramon Moore in 1962 - 1966 ?
- ▶ T. Sunaga in 1958 ?
- ▶ M. Warmus in 1956?
- ▶ P.S. Dwyer in 1951 (closed intervals only)?
- ▶ Rosalind Cecil Young in 1931 ?
- ▶ ...
- ▶ Archimedes: 3rd century BC, to enclose π !

Cf. <http://www.cs.utep.edu/interval-comp/>, click on *Early papers*.

Historical remarks

Who invented Interval Arithmetic?

- ▶ Ramon Moore in 1962 - 1966 ?
- ▶ T. Sunaga in 1958 ?
- ▶ M. Warmus in 1956?
- ▶ P.S. Dwyer in 1951 (closed intervals only)?
- ▶ Rosalind Cecil Young in 1931 ?
- ▶ ...
- ▶ Archimedes: 3rd century BC, to enclose π !

Cf. <http://www.cs.utep.edu/interval-comp/>, click on *Early papers*.

Historical remarks

Who invented Interval Arithmetic?

- ▶ Ramon Moore in 1962 - 1966 ?
- ▶ T. Sunaga in 1958 ?
- ▶ M. Warmus in 1956?
- ▶ P.S. Dwyer in 1951 (closed intervals only)?
- ▶ Rosalind Cecil Young in 1931 ?
- ▶ ...
- ▶ Archimedes: 3rd century BC, to enclose π !

Cf. <http://www.cs.utep.edu/interval-comp/>, click on *Early papers*.

Historical remarks

Who invented Interval Arithmetic?

- ▶ Ramon Moore in 1962 - 1966 ?
- ▶ T. Sunaga in 1958 ?
- ▶ M. Warmus in 1956?
- ▶ P.S. Dwyer in 1951 (closed intervals only)?
- ▶ Rosalind Cecil Young in 1931 ?
- ▶ ...
- ▶ Archimedes: 3rd century BC, to enclose π !

Cf. <http://www.cs.utep.edu/interval-comp/>, click on *Early papers*.

Historical remarks

Who invented Interval Arithmetic?

- ▶ Ramon Moore in 1962 - 1966 ?
- ▶ T. Sunaga in 1958 ?
- ▶ M. Warmus in 1956?
- ▶ P.S. Dwyer in 1951 (closed intervals only)?
- ▶ Rosalind Cecil Young in 1931 ?
- ▶ ...
- ▶ Archimedes: 3rd century BC, to enclose π !

Cf. <http://www.cs.utep.edu/interval-comp/>, click on *Early papers*.

Historical remarks

Childhood until the seventies.

Popularization in the 1980, German school (U. Kulisch).

IEEE-754 standard for floating-point arithmetic in 1985:
directed roundings are standardized and available (?).

Since the nineties: interval **algorithms**.

IEEE-1788 standard for interval arithmetic in 2012?

I hope so. . .

Agenda

Introduction

Definitions of interval arithmetic

operations, function extensions

Cons and pros

Cons: overestimation, complexity

Pros: contractant iterations, Brouwer's theorem

Some algorithms

solving linear systems

Newton

global optimization wo/with constraints

constraints programming

Variants

affine arithmetic, Taylor models arithmetic

Conclusions

A brief introduction

Interval arithmetic: replace numbers by intervals and compute.

Fundamental theorem of interval arithmetic:
(or “Thou shalt not lie”):

the exact result (number or set) is contained in the computed interval.

No result is lost, the computed interval is guaranteed to contain every possible result.

A brief introduction

Interval Arithmetic and certified scientific computing:

two directions

1. replace floating-point arithmetic by interval arithmetic to bound from above roundoff errors;
2. replace floating-point arithmetic and algorithms by interval ones to compute guaranteed enclosures.

A brief introduction

Interval arithmetic: replace numbers by intervals and compute.
Initially: introduced to take into account roundoff errors (Moore 1966)

and also uncertainties (on the physical data...).

Later: computations “in the large”, computations with sets.

Interval analysis: develop algorithms for **reliable (or verified, or guaranteed, or certified) computing**,
that are suited for interval arithmetic,
i.e. different from the algorithms from classical numerical analysis.

A brief introduction: examples of applications

- ▶ control the roundoff errors, cf. computational geometry
- ▶ solve several problems with certified solutions: linear and nonlinear systems of equations and inequalities, constraints satisfaction, (non/convex, un/constrained) global optimization, integrate ODEs e.g. particles trajectories...
- ▶ mathematical proofs: cf. Hales' proof of the Kepler's conjecture
or Tucker's proof that the Lorenz system is chaotic.

Cf. <http://www.cs.utep.edu/interval-comp/>

Agenda

Introduction

Definitions of interval arithmetic

operations, function extensions

Cons and pros

Cons: overestimation, complexity

Pros: contractant iterations, Brouwer's theorem

Some algorithms

solving linear systems

Newton

global optimization wo/with constraints

constraints programming

Variants

affine arithmetic, Taylor models arithmetic

Conclusions

Definitions: intervals

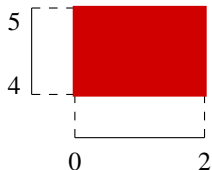
Objects:

- ▶ intervals of real numbers = closed connected sets of \mathbf{R}
 - ▶ interval for π : $[3.14159, 3.14160]$
 - ▶ data d measured with an absolute error less than $\pm\varepsilon$:
 $[d - \varepsilon, d + \varepsilon]$
- ▶ interval vector: components = intervals; also called *box*

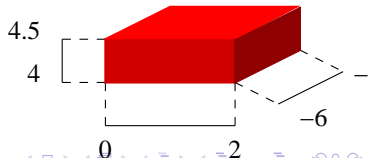
$[0 ; 2]$



$\begin{pmatrix} [0 ; 2] \\ [4 ; 5] \end{pmatrix}$



$\begin{pmatrix} [0;2] \\ [4 ; 4.5] \\ [-6 ; -5] \end{pmatrix}$



Definitions: operations

$$\mathbf{x} \diamond \mathbf{y} = \mathbf{Hull}\{x \diamond y : x \in \mathbf{x}, y \in \mathbf{y}\}$$

Arithmetic and algebraic operations: use the monotony

$$\begin{aligned} [\underline{x}, \bar{x}] + [\underline{y}, \bar{y}] &= [\underline{x} + \underline{y}, \bar{x} + \bar{y}] \\ [\underline{x}, \bar{x}] - [\underline{y}, \bar{y}] &= [\underline{x} - \bar{y}, \bar{x} - \underline{y}] \\ [\underline{x}, \bar{x}] \times [\underline{y}, \bar{y}] &= [\min(\underline{x} \times \underline{y}, \underline{x} \times \bar{y}, \bar{x} \times \underline{y}, \bar{x} \times \bar{y}), \max(\text{ibid.})] \\ [\underline{x}, \bar{x}]^2 &= [\min(\underline{x}^2, \bar{x}^2), \max(\underline{x}^2, \bar{x}^2)] \text{ if } 0 \notin [\underline{x}, \bar{x}] \\ &= [0, \max(\underline{x}^2, \bar{x}^2)] \text{ otherwise} \\ 1 / [\underline{y}, \bar{y}] &= [\min(1/\underline{y}, 1/\bar{y}), \max(1/\underline{y}, 1/\bar{y})] \text{ if } 0 \notin [\underline{y}, \bar{y}] \\ [\underline{x}, \bar{x}] / [\underline{y}, \bar{y}] &= [\underline{x}, \bar{x}] \times (1 / [\underline{y}, \bar{y}]) \text{ if } 0 \notin [\underline{y}, \bar{y}] \\ \sqrt{[\underline{x}, \bar{x}]} &= [\sqrt{\underline{x}}, \sqrt{\bar{x}}] \text{ if } 0 \leq \underline{x}, [0, \sqrt{\bar{x}}] \text{ otherwise} \end{aligned}$$

Definitions: operations

Algebraic properties: associativity, commutativity hold, some are lost:

- ▶ subtraction is not the inverse of addition, in particular $\mathbf{x} - \mathbf{x} \neq [0]$
- ▶ division is not the inverse of multiplication
- ▶ squaring is tighter than multiplication by oneself
- ▶ multiplication is only sub-distributive wrt addition

Definitions: functions

Definition:

an interval extension f of a function f satisfies

$$\forall x, f(x) \subset \mathbf{f}(x), \text{ and } \forall x, f(\{x\}) = \mathbf{f}(\{x\}).$$

Elementary functions: again, use the monotony.

$$\begin{aligned} \exp x &= [\exp \underline{x}, \exp \bar{x}] \\ \log x &= [\log \underline{x}, \log \bar{x}] \text{ if } \underline{x} \geq 0, [-\infty, \log \bar{x}] \text{ if } \bar{x} > 0 \\ \sin[\pi/6, 2\pi/3] &= [1/2, 1] \\ \dots & \end{aligned}$$

Definitions: function extension

Example: $f(x) = x^2 - x + 1$ with $x \in [-2, 1]$.

$$[-2, 1]^2 - [-2, 1] + 1 = [0, 4] + [-1, 2] + 1 = [0, 7].$$

Since $x^2 - x + 1 = x(x - 1) + 1$, we get $[-2, 1] \cdot ([-2, 1] - 1) + 1 = [-2, 1] \cdot [-3, 0] + 1 = [-3, 6] + 1 = [-2, 7]$.

Since $x^2 - x + 1 = (x - 1/2)^2 + 3/4$, we get $([-2, 1] - 1/2)^2 + 3/4 = [-5/2, 1/2]^2 + 3/4 = [0, 25/4] + 3/4 = [3/4, 7] = f([-2, 1])$.

Problem with this definition: infinitely many interval extensions, syntactic use (instead of semantic).

How to choose the best extension? How to choose a good one?

Definitions: function extension

Mean value theorem of order 1 (Taylor expansion of order 1):

$$\forall x, \forall y, \exists \xi_{x,y} \in (x, y) : f(y) = f(x) + (y - x) \cdot f'(\xi_{x,y})$$

Interval interpretation:

$$\forall y \in \mathbf{x}, \forall \tilde{x} \in \mathbf{x}, f(y) \in f(\tilde{x}) + (y - \tilde{x}) \cdot \mathbf{f}'(\mathbf{x})$$

$$\Rightarrow f(\mathbf{x}) \subset f(\tilde{x}) + (\mathbf{x} - \tilde{x}) \cdot \mathbf{f}'(\mathbf{x})$$

Mean value theorem of order 2 (Taylor expansion of order 2):

$$\forall x, \forall y, \exists \xi_{x,y} \in (x, y) : f(y) = f(x) + (y - x) \cdot f'(x) + \frac{(y - x)^2}{2} \cdot f''(\xi_{x,y})$$

Interval interpretation:

$$\forall y \in \mathbf{x}, \forall \tilde{x} \in \mathbf{x}, f(y) \in f(\tilde{x}) + (y - \tilde{x}) \cdot \mathbf{f}'(\tilde{x}) + \frac{(y - \tilde{x})^2}{2} \cdot \mathbf{f}''(\mathbf{x})$$

$$\Rightarrow f(\mathbf{x}) \subset f(\tilde{x}) + (\mathbf{x} - \tilde{x}) \cdot \mathbf{f}'(\tilde{x}) + \frac{(\mathbf{x} - \tilde{x})^2}{2} \cdot \mathbf{f}''(\mathbf{x})$$

Definitions: function extension

No need to go further:

- ▶ it is difficult to compute (automatically) the derivatives of higher order, especially for multivariate functions;
- ▶ there is no (theoretical) gain in quality.

Theorem:

- ▶ for the natural extension \mathbf{f} of f , it holds $d(f(\mathbf{x}), \mathbf{f}(\mathbf{x})) \leq \mathcal{O}(w(\mathbf{x}))$
- ▶ for the first order Taylor extension \mathbf{f}_{T_1} of f , it holds $d(f(\mathbf{x}), \mathbf{f}_{T_1}(\mathbf{x})) \leq \mathcal{O}(w(\mathbf{x})^2)$
- ▶ getting an order higher than 3 is impossible without the squaring operation, is difficult even with it...

Agenda

Introduction

Definitions of interval arithmetic

operations, function extensions

Cons and pros

Cons: overestimation, complexity

Pros: contractant iterations, Brouwer's theorem

Some algorithms

solving linear systems

Newton

global optimization wo/with constraints

constraints programming

Variants

affine arithmetic, Taylor models arithmetic

Conclusions

Cons: overestimation (1/2)

The result encloses the true result, but it is too large:
overestimation phenomenon.

Two main sources: variable dependency and wrapping effect.

(Loss of) Variable dependency:

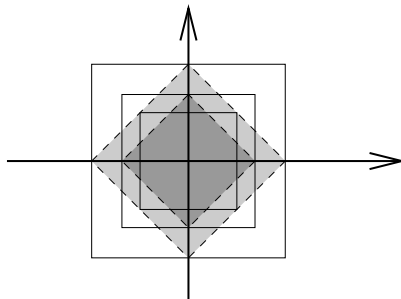
$$\mathbf{x} - \mathbf{x} = \{x - y : x \in \mathbf{x}, y \in \mathbf{x}\} \neq \{x - x : x \in \mathbf{x}\} = \{0\}.$$

Cons: overestimation (2/2)

Wrapping effect



image of $f(x)$
with $f : \mathbf{R}^2 \rightarrow \mathbf{R}^2$



2 successive rotations of $\pi/4$
of the little central square

Cons: Complexity: almost every problem is NP-hard

Gaganov 1982, Rohn 1994 ff, Kreinovich...

- ▶ evaluate a function on a box (cartesian product of intervals)
- ▶ evaluate a function on a box up to ε
- ▶ solve a linear system
- ▶ solve a linear system up to $1/4n^4$ ($n = \text{dim. of the system}$)
- ▶ determine if the solution of a linear system is bounded
- ▶ compute the matrix norm $\|\mathbf{A}\|_{\infty,1}$
- ▶ determine if an interval matrix (= a matrix with interval coefficients) is regular, i.e. if every possible punctual matrix in it is regular
- ▶ ...

Cons: Complexity: Gaganov 1982

evaluation of a multivariate polynomial with rational coeff. on a box is NP-hard

Idea: reduce polynomially CNF-3 to this problem.

On n boolean variables q_1, \dots, q_n , a formula f in CNF-3 is defined by

$$f = \bigwedge_{i=1}^m f_i \text{ with } f_i = \bigvee_{j=1}^{1,2 \text{ or } 3} r_{i,j}$$

with $r_{i,j} = q_{k_{i,j}}$ or $r_{i,j} = \neg q_{k_{i,j}}$.

Cons: Complexity: Gaganov 1982

evaluation of a multivariate polynomial with rational coeff. on a box is NP-hard

To each boolean variable q_i , let us associate a real variable $x_i \in [0, 1]$.

Meaning: $x_i = 0$ if $q_i = F$ and $x_i = 1$ if $q_i = T$.

Goal: get a polynomial which takes only values in $[0, 1]$
i.e. allow only product of terms or of $(1 - \text{term})$.

A product corresponds to a conjunction and $1 - x$ to a negation

\Rightarrow express f and the f_i using conjunctions and negations

\Rightarrow express the f_i as $\neg \bigwedge_{j=1}^{1,2\text{or}3} \neg r_{i,j}$.

Cons: Complexity: Gaganov 1982

evaluation of a multivariate polynomial with rational coeff. on a box is NP-hard

To each boolean variable q_i , let us associate a real variable $x_i \in [0, 1]$.

Meaning: $x_i = 0$ if $q_i = F$ and $x_i = 1$ if $q_i = T$.

Goal: get a polynomial which takes only values in $[0, 1]$
i.e. allow only product of terms or of $(1 - \text{term})$.

A product corresponds to a conjunction and $1 - x$ to a negation

\Rightarrow express f and the f_i using conjunctions and negations

\Rightarrow express the f_i as $\neg \bigwedge_{j=1}^{1,2\text{or}3} \neg r_{i,j}$.

Cons: Complexity: Gaganov 1982

evaluation of a multivariate polynomial with rational coeff. on a box is NP-hard

To each boolean variable q_i , let us associate a real variable $x_i \in [0, 1]$.

Meaning: $x_i = 0$ if $q_i = F$ and $x_i = 1$ if $q_i = T$.

Goal: get a polynomial which takes only values in $[0, 1]$
i.e. allow only product of terms or of $(1 - \text{term})$.

A product corresponds to a conjunction and $1 - x$ to a negation

\Rightarrow express f and the f_i using conjunctions and negations

\Rightarrow express the f_i as $\neg \bigwedge_{j=1}^{1,2 \text{ or } 3} \neg r_{i,j}$.

Cons: Complexity: Gaganov 1982

evaluation of a multivariate polynomial with rational coeff. on a box is NP-hard

To each boolean variable q_i , let us associate a real variable $x_i \in [0, 1]$.

Meaning: $x_i = 0$ if $q_i = F$ and $x_i = 1$ if $q_i = T$.

Goal: get a polynomial which takes only values in $[0, 1]$
i.e. allow only product of terms or of $(1 - \text{term})$.

A product corresponds to a conjunction and $1 - x$ to a negation

\Rightarrow express f and the f_i using conjunctions and negations

\Rightarrow express the f_i as $\neg \bigwedge_{j=1}^{1,2\text{or}3} \neg r_{i,j}$.

Cons: Complexity: Gaganov 1982

evaluation of a multivariate polynomial with rational coeff. on a box is NP-hard

More precisely:

1. to each $r_{i,j}$ let us associate a polynomial $y_{i,j}$ (corresponding to the negation of $r_{i,j}$) defined by

$$r_{i,j} = q_{k_{i,j}} \rightarrow y_{i,j}(x) = 1 - x_{k_{i,j}}$$

$$r_{i,j} = \neg q_{k_{i,j}} \rightarrow y_{i,j}(x) = x_{k_{i,j}}$$

2. to each f_i , let us associate a polynomial p_i (corresponding to the negation of f_i) defined by $f_i = \bigwedge r_{i,j} \rightarrow p_i(x) = \prod y_{i,j}(x)$.
3. to f , let us associate the polynomial p defined by $f = \bigwedge_{i=1}^m f_i \rightarrow p(x) = \prod_{i=1}^m (1 - p_i(x))$.

Cons: Complexity: Gaganov 1982

evaluation of a multivariate polynomial with rational coeff. on a box is NP-hard

More precisely:

1. to each $r_{i,j}$ let us associate a polynomial $y_{i,j}$ (corresponding to the negation of $r_{i,j}$) defined by

$$r_{i,j} = q_{k_{i,j}} \rightarrow y_{i,j}(x) = 1 - x_{k_{i,j}}$$

$$r_{i,j} = \neg q_{k_{i,j}} \rightarrow y_{i,j}(x) = x_{k_{i,j}}$$

2. to each f_i , let us associate a polynomial p_i (corresponding to the negation of f_i) defined by $f_i = \bigwedge r_{i,j} \rightarrow p_i(x) = \prod y_{i,j}(x)$.

3. to f , let us associate the polynomial p defined by
 $f = \bigwedge_{i=1}^m f_i \rightarrow p(x) = \prod_{i=1}^m (1 - p_i(x))$.

Cons: Complexity: Gaganov 1982

evaluation of a multivariate polynomial with rational coeff. on a box is NP-hard

More precisely:

1. to each $r_{i,j}$ let us associate a polynomial $y_{i,j}$ (corresponding to the negation of $r_{i,j}$) defined by

$$r_{i,j} = q_{k_{i,j}} \rightarrow y_{i,j}(x) = 1 - x_{k_{i,j}}$$

$$r_{i,j} = \neg q_{k_{i,j}} \rightarrow y_{i,j}(x) = x_{k_{i,j}}$$

2. to each f_i , let us associate a polynomial p_i (corresponding to the negation of f_i) defined by $f_i = \bigwedge r_{i,j} \rightarrow p_i(x) = \prod y_{i,j}(x)$.

3. to f , let us associate the polynomial p defined by $f = \bigwedge_{i=1}^m f_i \rightarrow p(x) = \prod_{i=1}^m (1 - p_i(x))$.

Cons: Complexity: Gaganov 1982

evaluation of a multivariate polynomial with rational coeff. on a box is NP-hard

Lemma:

1. $\forall x \in [0, 1], p(x) \in [0, 1]$.
2. if α is a boolean vector and β is the associated 0 – 1 vector, then

$$\begin{aligned} f(\alpha) = T &\Rightarrow p(\beta) = 1 \\ f(\alpha) = F &\Rightarrow p(\beta) = 0. \end{aligned}$$

3. if f is not feasible, then $\forall x \in [0, 1]^n, p(x) \leq 7/8$.

Cons: Complexity: Gaganov 1982

evaluation of a multivariate polynomial with rational coeff. on a box is NP-hard

Proof of (3): (proving (1) and (2) is easy).

$\forall x \in [0, 1]^n$, let us consider β the 0-1 vector obtained by rounding x to the nearest.

Since f is not feasible, $p(\beta) = 0$.

Since $p(x) = \prod_{i=1}^m (1 - p_i(x))$, $\exists i_0$ such that $1 - p_{i_0}(\beta) = 0$.

One can prove that $p_{i_0}(x) \geq 1/8$, using the fact that it is the product of at most three terms, each of them $\geq 1/2$, using the fact that β is the rounding to nearest of x . Thus $1 - p_{i_0}(x) \leq 7/8$.

The remaining factors $1 - p_j(x)$ are less or equal to 1.

Thus $p(x) = \prod_{i=1}^m (1 - p_i(x)) \leq 7/8$.

Consequence: since checking the feasibility of a CNF-3 formula is NP-hard, evaluating a multivariate polynomial (up to a small ε) is

Agenda

Introduction

Definitions of interval arithmetic

operations, function extensions

Cons and pros

Cons: overestimation, complexity

Pros: contractant iterations, Brouwer's theorem

Some algorithms

solving linear systems

Newton

global optimization wo/with constraints

constraints programming

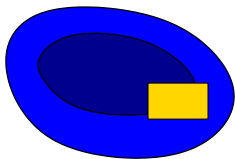
Variants

affine arithmetic, Taylor models arithmetic

Conclusions

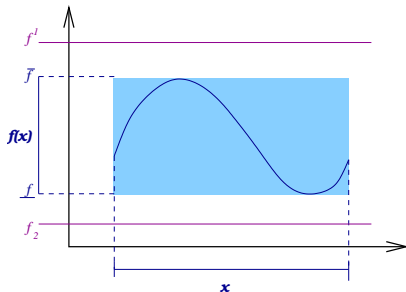
Pros: set computing

Behaviour safe?
controllable? dangerous?



always controllable.

On x , are the extrema of the function f
 $> f^1, < f_2$?

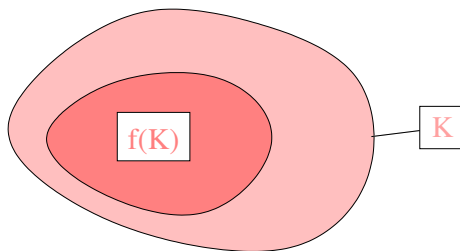


No if $f(x) = [\underline{f}, \bar{f}] \subset [f_2, f^1]$.

Pros: Brouwer-Schauder theorem

A function f which is continuous on the unit ball B and which satisfies $f(B) \subset B$ has a fixed point on B .

Furthermore, if $f(B) \subset \text{int}B$ then f has a unique fixed point on B .



The theorem remains valid if B is replaced by a compact K and in particular an interval.

Agenda

Introduction

Definitions of interval arithmetic

operations, function extensions

Cons and pros

Cons: overestimation, complexity

Pros: contractant iterations, Brouwer's theorem

Some algorithms

solving linear systems

Newton

global optimization wo/with constraints

constraints programming

Variants

affine arithmetic, Taylor models arithmetic

Conclusions

Algorithm: linear systems solving (Hansen-Sengupta)

Problem: solve $Ax = b$, i.e. determine

$\text{Hull}(\Sigma_{\exists\exists}(A, b)) = \text{Hull}(\{x : \exists A \in A, \exists b \in b, Ax = b\})$.

Pre-processing:

multiply the system by an approximate $\text{mid}(A)^{-1}$.

New system = $\text{mid}(A)^{-1}Ax = b$.

Hope: contracting iteration.

Algorithm: apply Gauss-Seidel iteration

while convergence not reached loop

for $i = 1$ to n do

$$x_i := \left(b_i - \sum_{j \neq i} A_{i,j} x_j \right) / A_{i,i}$$

Agenda

Introduction

Definitions of interval arithmetic

operations, function extensions

Cons and pros

Cons: overestimation, complexity

Pros: contractant iterations, Brouwer's theorem

Some algorithms

solving linear systems

Newton

global optimization wo/with constraints

constraints programming

Variants

affine arithmetic, Taylor models arithmetic

Conclusions

Algorithm: solving a nonlinear system: Newton

Why a specific iteration for interval computations?

Usual formula:

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$$

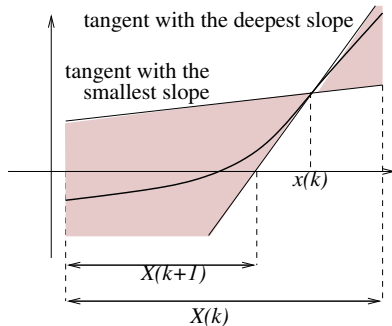
Direct interval transposition:

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$$
$$w(x_{k+1}) = w(x_k) + w\left(\frac{f(x_k)}{f'(x_k)}\right) > w(x_k)$$

divergence!

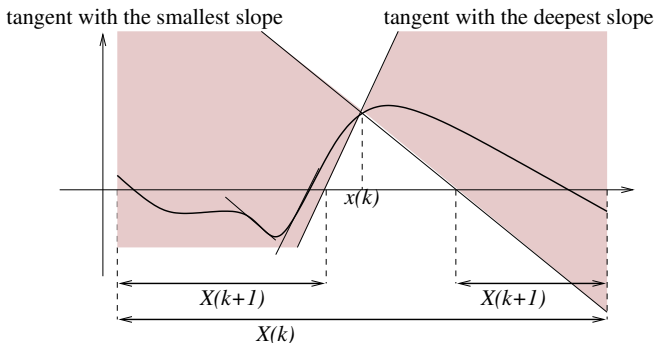
Algorithm: interval Newton principle of an iteration

(Hansen-Greenberg 83, Baker Kearfott 95-97, Mayer 95, van Hentenryck et al. 97)



$$x_1 := \left(x - \frac{F(\{x\})}{F'(x)} \right) \cap x$$

Algorithm: interval Newton principle of an iteration



$$(\mathbf{x}_1, \mathbf{x}_2) := \left(x - \frac{F(\{x\})}{F'(\mathbf{x})} \right) \cap x$$

Algorithm: interval Newton

Input: F, F', x_0 // x_0 initial search interval
Initialization: $\mathcal{L} = \{x_0\}, \alpha = 0.75$ // any value in $]0.5, 1[$
 is suitable
Loop: while $\mathcal{L} \neq \emptyset$
 Suppress (x, \mathcal{L})
 $x := \text{mid}(x)$
 $(x_1, x_2) := \left(x - \frac{F(\{x\})}{F'(x)}\right) \cap x$ // x_1 and x_2 can be
 empty
 if $w(x_1) > \alpha w(x)$ or $w(x_2) > \alpha w(x)$ then $(x_1, x_2) := \text{bisect}(x)$
 if $x_1 \neq \emptyset$ and $F(x_1) \ni 0$ then
 if $w(x_1)/|\text{mid}(x_1)| \leq \varepsilon_x$ or $w(F(x_1)) \leq \varepsilon_Y$ then Insert x_1 in Res
 else Insert x_1 in \mathcal{L}
 same handling of x_2
Output: Res, a list of intervals that may contain the roots.

Algorithm: interval Newton

properties

Existence and uniqueness of a root are proven:

if there is no hole and if the new iterate (before \cap) is contained in the interior of the previous one.

Existence of a root is proven:

- ▶ using the mean value theorem:
OK if $f(\inf(x))$ and $f(\sup(x))$ have opposite signs.
(Miranda theorem in higher dimensions).
- ▶ using Schauder theorem: if the new iterate (before \cap) in contained in the previous one.

Agenda

Introduction

Definitions of interval arithmetic

operations, function extensions

Cons and pros

Cons: overestimation, complexity

Pros: contractant iterations, Brouwer's theorem

Some algorithms

solving linear systems

Newton

global optimization wo/with constraints

constraints programming

Variants

affine arithmetic, Taylor models arithmetic

Conclusions

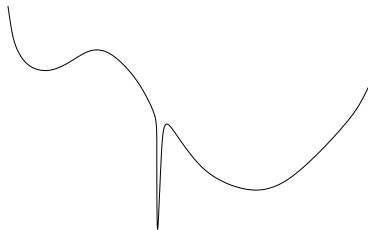
Algorithm: optimize a continuous function

Problem: $f : \mathbf{R}^n \rightarrow \mathbf{R}$, determine x^* and f^* that verify

$$f^* = f(x^*) = \min_x f(x)$$

Assumptions:

- ▶ search within a box x_0
- ▶ $x^* \in$ in the interior of (x_0) , not at the boundary
- ▶ f continuous enough: \mathcal{C}^2



Algorithm: optimize a continuous function

(Ratschek and Rokne 1988, Hansen 1992, Kearfott 1996...)

Goal: find the minimum of f , continuous function on a box x_0 .

x_0 current box

\bar{f} current upper bound of f^*

while there is a box in the waiting list

if $f(x) > \bar{f}$ then

reject x

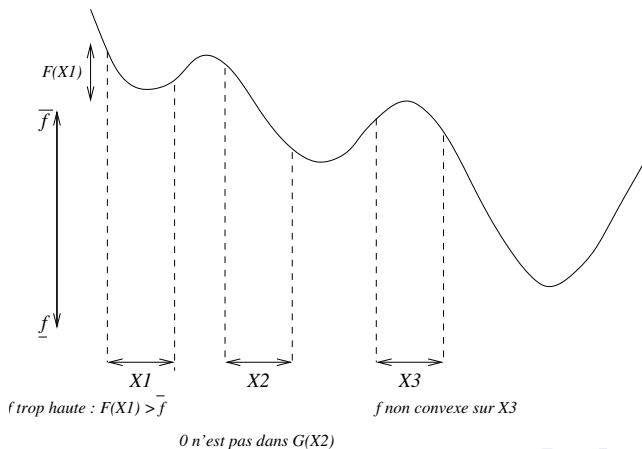
otherwise

update \bar{f} : if $f(\text{mid}(x)) < \bar{f}$ then $\bar{f} = f(\text{mid}(x))$

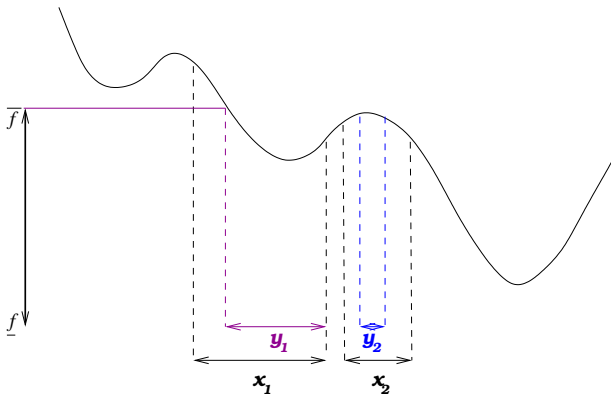
bisect x into x_1 and x_2

examine x_1 and x_2

Algorithm: optimize a continuous function the rejection procedure



Algorithm: optimize a continuous function the reduction procedure



Algorithm: optimize a continuous function

Hansen algorithm Hansen 1992

\mathcal{L} = list of not yet examined boxes := $\{x_0\}$

while $\mathcal{L} \neq \emptyset$ **loop**

remove x from \mathcal{L}

reject x ?

yes if $f(x) > \bar{f}$

yes if $\text{Grad}f(x) \not\approx 0$

yes if $Hf(x)$ has its diagonal non > 0

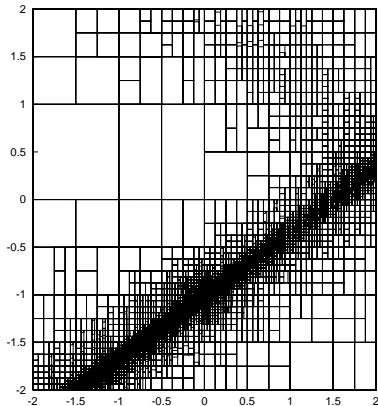
reduce x

Newton applied to the gradient

solve $y \subset x$ such that $f(y) \leq \bar{f}$

bisect y : insert the resulting y_1 and y_2 in \mathcal{L}

Exemple de découpage d'un pavé $[-2, 2]^2$



Algorithm: constrained optimization

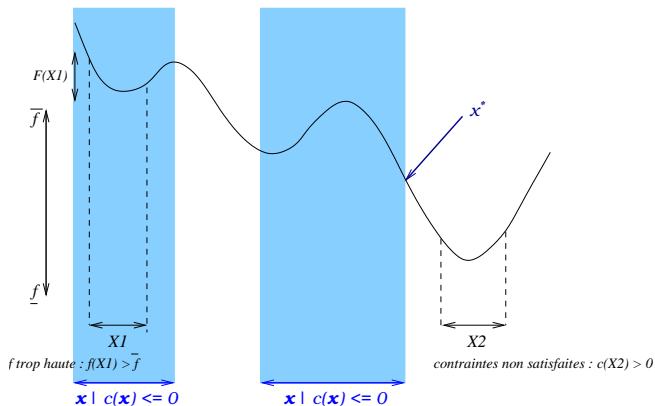
Problem: $f : \mathbf{R}^n \rightarrow \mathbf{R}$ and $c : \mathbf{R}^n \rightarrow \mathbf{R}^m$,
determine x^* and f^* that verify

$$f^* = f(x^*) = \min_{\{x | c(x) \leq 0\}} f(x)$$

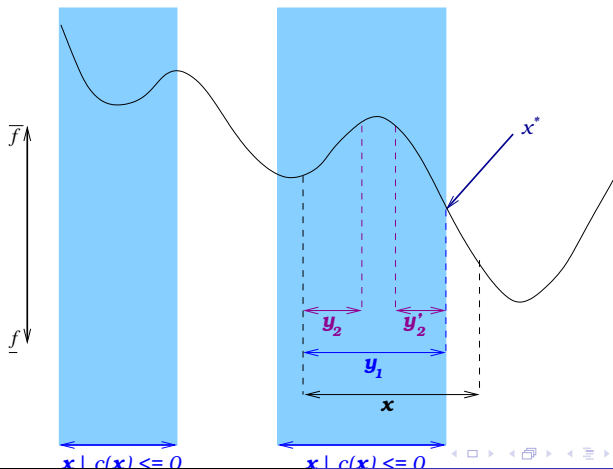
Assumptions:

- ▶ search within a box x_0
- ▶ f continuous enough: \mathcal{C}^2
- ▶ c continuous enough: \mathcal{C}^1

Algorithm: constrained optimization $c(x) \leq 0$ the rejection procedure



Algorithm: constrained optimization $c(x) \leq 0$ the reduction procedure



Algorithm: constrained optimization $c(x) \leq 0$

$\mathcal{L} := \{x_0\}$
while $\mathcal{L} \neq \emptyset$ **loop**
 remove x from \mathcal{L}
 reject x ?
 yes if $f(x) > \bar{f}$
 yes if $\text{Grad}f(x) \neq 0$
 yes if f not convex on x
 reduce x
 solve $y \subset x \mid f(y) \leq \bar{f}$
 Newton applied to the gradient
 bisect y into y_1 and y_2
 insert y_1 and y_2 in \mathcal{L}

$\mathcal{L} := \{x_0\}$
while $\mathcal{L} \neq \emptyset$ **loop**
 remove x from \mathcal{L}
 reject x ?
 yes if $f(x) > \bar{f}$
 yes if $c(x) > 0$
 reduce x
 solve $y \subset x$ such that $c(y) \leq 0$
 Newton applied to the Lagrangian
 bisect y into y_1 and y_2
 insert y_1 and y_2 in \mathcal{L}

Agenda

Introduction

Definitions of interval arithmetic

operations, function extensions

Cons and pros

Cons: overestimation, complexity

Pros: contractant iterations, Brouwer's theorem

Some algorithms

solving linear systems

Newton

global optimization wo/with constraints

constraints programming

Variants

affine arithmetic, Taylor models arithmetic

Conclusions

Algorithm: constraints programming

Cleary 1987, Benhamou et al. 1999, Jaulin et al. 2001

Problem:

$$\begin{cases} c_1(x_1, \dots, x_n) = 0 \\ \vdots \\ c_p(x_1, \dots, x_n) = 0 \end{cases}$$

expressed as:

$$\begin{aligned} y_i &= x_i && \text{for } 1 \leq i \leq n \\ y_k &= y_i \diamond y_j && \text{for } n+1 \leq k \leq m \text{ and } i, j < k \\ &&& y_k \text{ auxiliary variable} \end{aligned}$$

$$\text{where } y_k = \varphi(y_i) \quad \text{for } n+1 \leq k \leq m \text{ and } i < k$$

Algorithm: constraints programming

Initializations: $y_1 := x_1, \dots, y_n := x_n$

Propagation: forward mode

for $k = n + 1$ to m loop

$$y_k := y_i \diamond y_j \text{ or } y_k := \varphi(y_i)$$

Propagation: backward mode

for $k = m$ to n loop

if y_k is defined as $y_i \diamond y_j$ then

$$y_i := (y_k \diamond^{-r} y_j) \cap y_i$$

$$y_j := (y_i \diamond^{-l} y_k) \cap y_j$$

else if y_k is defined as $\varphi(y_i)$ then

$$y_i := \varphi^{-1}(y_k) \cap y_i$$

Algorithm: constraints programming:

$x_1 = [0, 2\pi/3], x_2 = [-1, 1], x_3 = [-1/2, 3]$

iter. 1 : forward

$y_4 = y_2^2$

$y_5 = y_1 y_4$

$y_6 = 2y_3$

$y_7 = y_5 - y_6$

$y_8 = \cos y_1$

$y_9 = y_8 + y_3$

backward

$y_9 = y_8 + y_3$

$y_8 = \cos y_1$

$y_7 = y_5 - y_6$

$y_6 = 2y_3$

$y_5 = y_1 y_4$

$y_4 = y_2^2$

$$\left\{ \begin{array}{l} x_1 x_2^2 - 2x_3 = 0 \\ \cos x_1 + x_3 = 0 \\ y_1 = [0, 2\pi/3], y_2 = [-1, 1], y_3 = [-1/2, 3] \\ y_4 = [0, 1] \\ y_5 = [0, 2\pi/3] \\ y_6 = [-1, 6] \\ y_7 = [-6, 1 + 2\pi/3] \ni 0 \\ y_8 = [-1/2, 1] \\ y_9 = [-1, 4] \ni 0 \\ \left\{ \begin{array}{l} y_8 = (y_9 - y_3) \cap y_8 = [-1/2, 1/2] \\ y_3 = (y_9 - y_8) \cap y_3 = [-1/2, 1/2] \end{array} \right. \\ y_1 = \cos^{-1} y_8 \cap y_1 = [\pi/3, 2\pi/3] \\ \left\{ \begin{array}{l} y_5 = (y_7 + y_6) \cap y_5 = [0, 2\pi/3] \\ y_6 = (y_5 - y_7) \cap y_6 = [0, 2\pi/3] \end{array} \right. \\ y_3 = (1/2 y_6) \cap y_3 = [0, 1/2] \\ \left\{ \begin{array}{l} y_1 = (y_5 / y_4) \cap y_1 = [\pi/3, 2\pi/3] \\ y_4 = (y_5 / y_1) \cap y_4 = [0, 1] \end{array} \right. \\ y_2 = \pm \sqrt{y_4} \cap y_2 = [-1, 1] \end{array} \right.$$

Algorithm: constraints programming: $\begin{cases} x_1 x_2^2 - 2x_3 = 0 \\ 2\cos x_1 + x_3 - 1 = 0 \end{cases}$

$$\mathbf{y}_1 = \left[\frac{\pi}{3}, \frac{2\pi}{3}\right], \mathbf{y}_2 = [-1, 1], \mathbf{y}_3 = \left[0, \frac{1}{2}\right]$$

$$\mathbf{y}_4 = [0, 1], \mathbf{y}_5 = [0, 2\pi/3], \mathbf{y}_6 = [0, 1]$$

$$\mathbf{y}_7 = 0, \mathbf{y}_8 = [-1/2, 1/2], \mathbf{y}_9 = 0$$

iter. 2: forward

$$\mathbf{y}_4 = \mathbf{y}_2^2$$

$$\mathbf{y}_5 = \mathbf{y}_1 \mathbf{y}_4$$

$$\mathbf{y}_6 = 2\mathbf{y}_3$$

$$\mathbf{y}_8 = \cos \mathbf{y}_1$$

backward

$$\mathbf{y}_9 = \mathbf{y}_8 + \mathbf{y}_3$$

$$\mathbf{y}_8 = \cos \mathbf{y}_1$$

$$\mathbf{y}_7 = \mathbf{y}_5 - \mathbf{y}_6$$

$$\mathbf{y}_6 = 2\mathbf{y}_3$$

$$\mathbf{y}_5 = \mathbf{y}_1 \mathbf{y}_4$$

$$\mathbf{y}_4 = \mathbf{y}_2^2$$

$$\mathbf{y}_4 = [0, 1]$$

$$\mathbf{y}_5 = [0, 2\pi/3]$$

$$\mathbf{y}_6 = [0, 1]$$

$$\mathbf{y}_8 = [-1/2, 1/2]$$

$$\begin{cases} \mathbf{y}_8 = (\mathbf{y}_9 - \mathbf{y}_3) \cap \mathbf{y}_8 = [-1/2, 0] \\ \mathbf{y}_3 = (\mathbf{y}_9 - \mathbf{y}_8) \cap \mathbf{y}_3 = [0, 1/2] \end{cases}$$

$$\mathbf{y}_1 = \cos^{-1} \mathbf{y}_8 \cap \mathbf{y}_1 = [\pi/2, 2\pi/3]$$

$$\begin{cases} \mathbf{y}_5 = (\mathbf{y}_7 + \mathbf{y}_6) \cap \mathbf{y}_5 = [0, 1] \\ \mathbf{y}_6 = (\mathbf{y}_5 - \mathbf{y}_7) \cap \mathbf{y}_6 = [0, 1] \end{cases}$$

$$\mathbf{y}_3 = (1/2 \mathbf{y}_6) \cap \mathbf{y}_3 = [0, 1/2]$$

$$\begin{cases} \mathbf{y}_1 = (\mathbf{y}_5 / \mathbf{y}_4) \cap \mathbf{y}_1 = [\pi/2, 2\pi/3] \\ \mathbf{y}_4 = (\mathbf{y}_5 / \mathbf{y}_1) \cap \mathbf{y}_4 = [0, 2/\pi] \end{cases}$$

$$\mathbf{y}_2 = \pm \sqrt{\mathbf{y}_4} \cap \mathbf{y}_2 = \left[-\sqrt{2/\pi}, \sqrt{2/\pi}\right]$$

Algorithm: constraints programming:
$$\begin{cases} x_1 x_2^2 - 2x_3 = 0 \\ \cos x_1 + x_3 = 0 \end{cases}$$

Problem:
$$\begin{cases} x_1 x_2^2 - 2x_3 = 0 \\ \cos x_1 + x_3 = 0 \end{cases}$$

with $\mathbf{x}_1 = [0, \frac{2\pi}{3}]$, $\mathbf{x}_2 = [-1, 1]$, $\mathbf{x}_3 = [-\frac{1}{2}, 3]$.

Optimal solution obtained after two iterations:

$\mathbf{x}_1 = [\frac{\pi}{2}, \frac{2\pi}{3}]$, $\mathbf{x}_2 = [-\sqrt{\frac{2}{\pi}}, \sqrt{\frac{2}{\pi}}]$, $\mathbf{x}_3 = [0, \frac{1}{2}]$.

Agenda

Introduction

Definitions of interval arithmetic

operations, function extensions

Cons and pros

Cons: overestimation, complexity

Pros: contractant iterations, Brouwer's theorem

Some algorithms

solving linear systems

Newton

global optimization wo/with constraints

constraints programming

Variants

affine arithmetic, Taylor models arithmetic

Conclusions

Affine arithmetic

Comba, Stolfi and Figueiredo (1993, 2004)

Definition: each input or computed quantity x is represented by

$$x = x_0 + \alpha_1 \varepsilon_1 + \alpha_2 \varepsilon_2 + \dots + \alpha_n \varepsilon_n$$

where $x_0, \alpha_1, \dots, \alpha_n$ are known real / floating-point numbers,
and $\varepsilon_1, \varepsilon_2 \dots \varepsilon_n$ are symbolic variables $\in [-1, +1]$.

Example: $x \in [3, 7]$ is represented by $x = 5 + 2\varepsilon$.

Operations:

$$(x + \sum_k \alpha_k \varepsilon_k) + (y + \sum_k \beta_k \varepsilon_k) = (x + y) + \sum_k (\alpha_k + \beta_k) \varepsilon_k.$$

$(x + \sum_k \alpha_k \varepsilon_k) \times (y + \sum_k \beta_k \varepsilon_k) = (x \times y) + \dots + \gamma_l \varepsilon_l$ with ε_l a new variable.

Roundoff errors: compute δ_l an upper bound of all roundoff errors and add it to γ_l .

Taylor models

Berz, Hoefkens and Makino 1998, Nedialkov, Neher

Principle: represent a function $f(x)$ for $x \in [-1, 1]$ by a polynomial part $p(x)$ and a reminder part (a big bin) I such that $\forall x \in [-1, 1], f(x) \in p(x) + I$.

Operations:

- ▶ affine operations: straightforward;
- ▶ non-affine operations: enclose the nonlinear terms and add this enclosure to the reminder.

Roundoff errors: determine an upper bound b on the roundoff errors and add $[-b, b]$ to the reminder.

Agenda

Introduction

Definitions of interval arithmetic

operations, function extensions

Cons and pros

Cons: overestimation, complexity

Pros: contractant iterations, Brouwer's theorem

Some algorithms

solving linear systems

Newton

global optimization wo/with constraints

constraints programming

Variants

affine arithmetic, Taylor models arithmetic

Conclusions

Conclusions

Interval algorithms

- ▶ can solve problems that other techniques are not able to solve
- ▶ is a simple version of set computing
- ▶ give effective versions of theorems which did not seem to be effective (Brouwer)
- ▶ can determine all zeros or all extrema of a continuous function
- ▶ overestimate the result
- ▶ is less efficient than floating-point arithmetic (theoretical factor: 4, practical factor: 20)
⇒ solve “small” problems.

Philosophical conclusion

Morale

- ▶ forget one's biases:
 - ▶ do not use without thinking algorithms which are supposed to be good ones (Newton)
 - ▶ do not reject without thinking algorithm which are supposed to be bad ones (Gauss-Seidel)
- ▶ prefer contracting iterations whenever possible