Introduction to Interval Computation

Milan Hladík\(^1\) Michal Černý\(^2\)

\(^1\) Faculty of Mathematics and Physics, Charles University in Prague, Czech Republic
http://kam.mff.cuni.cz/~hladik/

\(^2\) Faculty of Computer Science and Statistics, University of Economics, Prague, Czech Republic
http://nb.vse.cz/~cernym/

Workshop on Interval Programming
7th International Conference of Iranian Operation Research Society
Semnan, Iran, May 12–13, 2014
Outline

1. Motivation
2. Interval Computations
3. Interval Functions
4. Algorithmic Issues
1 Motivation

2 Interval Computations

3 Interval Functions

4 Algorithmic Issues
Interval computation = solving problems with interval data.

Where interval data do appear

1. numerical analysis (handling rounding errors)
2. computer-assisted proofs
3. global optimization
4. modelling uncertainty (an alternative to fuzzy and stochastic programming)
Example (Rump, 1988)

Consider the expression

\[ f = 333.75b^6 + a^2(11a^2b^2 - b^6 - 121b^4 - 2) + 5.5b^8 + \frac{a}{2b}, \]

with

\[ a = 77617, \quad b = 33096. \]

Calculations from 80s gave

- single precision \( f \approx 1.172603 \ldots \)
- double precision \( f \approx 1.1726039400531 \ldots \)
- extended precision \( f \approx 1.172603940053178 \ldots \)
- the true value \( f = -0.827386 \ldots \)
Computer-Assisted Proofs

**Kepler conjecture**

What is the densest packing of balls? (Kepler, 1611)

That one how the oranges are stacked in a shop.

The conjecture was proved by T.C. Hales (2005).

**Double bubble problem**

What is the minimal surface of two given volumes?

Two pieces of spheres meeting at an angle of $120^\circ$.

Global Optimization

Rastrigin’s function \( f(x) = 20 + x_1^2 + x_2^2 - 10(\cos(2\pi x_1) + \cos(2\pi x_2)) \)
Further Sources of Intervals

- Mass number of chemical elements (due to several stable isotopes)
  - [12.0096, 12.0116] for the carbon
- Physical constants
  - [9.78, 9.82] \(ms^{-2}\) for the gravitational acceleration
- Mathematical constants
  - \(\pi \in [3.1415926535897932384, 3.1415926535897932385]\).
- Measurement errors
  - Temperature measured 23°C ± 1°C
- Discretization
  - Time is split in days
  - Temperature during the day in [18, 29]°C for Semnan in May
- Missing data
  - What was the temperature in Semnan on May 12, 1999?
    - Very probably in [10, 40]°C.
- Processing a state space
  - Find robot singularities, where it may breakdown
  - Check joint angles [0, 180]°.
1 Motivation

2 Interval Computations

3 Interval Functions

4 Algorithmic Issues
Interval Computations

**Notation**

An interval matrix

\[ A := [A, \bar{A}] = \{A \in \mathbb{R}^{m \times n} | A \leq A \leq \bar{A}\}. \]

The center and radius matrices

\[ A^c := \frac{1}{2}(\bar{A} + A), \quad A^\Delta := \frac{1}{2}(\bar{A} - A). \]

The set of all \( m \times n \) interval matrices: \( \mathbb{I}\mathbb{R}^{m \times n} \).

**Main Problem**

Let \( f : \mathbb{R}^n \mapsto \mathbb{R}^m \) and \( x \in \mathbb{I}\mathbb{R}^n \). Determine the image

\[ f(x) = \{f(x) : x \in x\}. \]
### Interval Arithmetic

\[
\begin{align*}
\mathbf{a} + \mathbf{b} &= [a + b, \bar{a} + b], \\
\mathbf{a} - \mathbf{b} &= [a - \bar{b}, \bar{a} - b], \\
\mathbf{a} \cdot \mathbf{b} &= [\min(ab, \bar{a}b, \bar{a}b, \bar{a}b), \max(ab, \bar{a}b, \bar{a}b, \bar{a}b)], \\
\mathbf{a} / \mathbf{b} &= [\min(a/b, \bar{a}/b, \bar{a}/b, \bar{a}/b), \max(a/b, \bar{a}/b, \bar{a}/b, \bar{a}/b)], & 0 \notin \mathbf{b}.
\end{align*}
\]

### Theorem (Basic properties of interval arithmetic)

- Interval addition and multiplication is commutative and associative.
- It is not distributive in general, but sub-distributive instead,
  \[
  \forall \mathbf{a}, \mathbf{b}, \mathbf{c} \in \mathbb{IR} : \mathbf{a}(\mathbf{b} + \mathbf{c}) \subseteq \mathbf{a}\mathbf{b} + \mathbf{a}\mathbf{c}.
  \]

### Example \((\mathbf{a} = [1, 2], \mathbf{b} = 1, \mathbf{c} = -1)\)

\[
\begin{align*}
\mathbf{a}(\mathbf{b} + \mathbf{c}) &= [1, 2] \cdot (1 - 1) = [1, 2] \cdot 0 = 0, \\
\mathbf{a}\mathbf{b} + \mathbf{a}\mathbf{c} &= [1, 2] \cdot 1 + [1, 2] \cdot (-1) = [1, 2] - [1, 2] = [-1, 1].
\end{align*}
\]
Exercises for YOU

Prove or Disprove

For $a, b, c \in \mathbb{IR}$ and $d \in \mathbb{R}$:

1. $a + b = a + c \Rightarrow b = c$.
2. $(b + c)d = bd + cd$.
3. $(a \geq 0$ and $ab = ac) \Rightarrow b = c$.
4. $a \subseteq b \iff |a^c - b^c| + a^\Delta \leq b^\Delta$.
5. $a \cap b \neq \emptyset \iff |a^c - b^c| \leq a^\Delta + b^\Delta$. 
### Images of Functions

#### Monotone Functions

If \( f : x \rightarrow \mathbb{R} \) is non-decreasing, then \( f(x) = [f(\underline{x}), f(\overline{x})] \).

#### Example

\[
\exp(x) = [\exp(\underline{x}), \exp(\overline{x})], \quad \log(x) = [\log(\underline{x}), \log(\overline{x})], \ldots
\]

#### Some Basic Functions

Images \( x^2, \sin(x), \ldots \), are easily calculated, too.

\[
x^2 = \begin{cases} 
   [\min(x^2, \overline{x}^2), \max(x^2, \overline{x}^2)] & \text{if } 0 \not\in x, \\
   [0, \max(x^2, \overline{x}^2)] & \text{otherwise}
\end{cases}
\]

#### But...

...what to do for more complex functions?
Images of Functions

Notice

\( f(x) \) need not be an interval (neither closed nor connected).

Interval Hull \( \Box f(x) \)

Compute the interval hull instead

\[
\Box f(x) = \bigcap_{v \in \mathbb{IR}^n : f(x) \subseteq v} v.
\]

Bad News

Computing \( \Box f(x) \) is still very difficult.

Interval Enclosure

Compute as tight as possible \( v \in \mathbb{IR}^n : f(x) \subseteq v \).
Interval Functions

**Definition (Inclusion Isotonicity)**

\( f : \mathbb{IR}^n \mapsto \mathbb{IR} \) is *inclusion isotonic* if for every \( x, y \in \mathbb{IR}^n \):

\[
x \subseteq y \Rightarrow f(x) \subseteq f(y).
\]

**Definition (Interval Extension)**

\( f : \mathbb{IR}^n \mapsto \mathbb{IR} \) is an *interval extension* of \( f : \mathbb{R}^n \mapsto \mathbb{R} \) if for every \( x \in \mathbb{R}^n \):

\[
f(x) = f(x).
\]

**Theorem (Fundamental Theorem of Interval Analysis)**

*If* \( f : \mathbb{IR}^n \mapsto \mathbb{IR} \) *satisfies both properties, then*

\[
f(x) \subseteq f(x), \quad \forall x \in \mathbb{IR}^n.
\]

**Proof.**

For every \( x \in x \), one has by interval extension and inclusion isotonicity that \( f(x) = f(x) \subseteq f(x) \), whence \( f(x) \subseteq f(x) \).
Natural Interval Extension

Definition (Natural Interval Extension)

Let $f : \mathbb{R}^n \mapsto \mathbb{R}$ be a function given by an arithmetic expression. The corresponding **natural interval extension** $\mathbf{f}$ of $f$ is defined by that expression when replacing real arithmetic by the interval one.

Theorem

**Natural interval extension of an arithmetic expression is both an interval extension and inclusion isotonic.**

Proof.

It is easy to see that interval arithmetic is both an interval extension and inclusion isotonic. Next, proceed by mathematical induction.
Natural Interval Extension

Example

\[ f(x) = x^2 - x, \quad x \in x = [-1, 2]. \]

Then

\[ x^2 - x = [-1, 2]^2 - [-1, 2] = [-2, 5], \]
\[ x(x - 1) = [-1, 2]([-1, 2] - 1) = [-4, 2], \]

Best one? \( (x - \frac{1}{2})^2 - \frac{1}{4} = ([-1, 2] - \frac{1}{2})^2 - \frac{1}{4} = [-\frac{1}{4}, 2]. \)

Theorem

Suppose that in an expression of \( f : \mathbb{R}^n \mapsto \mathbb{R} \) each variable \( x_1, \ldots, x_n \) appears at most once. The corresponding natural interval extension \( f(x) \) satisfies for every \( x \in \mathbb{I}\mathbb{R}^n : f(x) = f(x). \)

Proof.

Inclusion “\( \subseteq \)” by the previous theorems.

Inclusion “\( \supseteq \)” by induction and exactness of interval arithmetic.
Software

Matlab libraries

- **Intlab** (by S.M. Rump),
  interval arithmetic and elementary functions
  http://www.ti3.tu-harburg.de/~rump/intlab/

- **Versoft** (by J. Rohn),
  verification software written in Intlab
  http://uivtx.cs.cas.cz/~rohn/matlab/

- **Lime** (by M. Hladík, J. Horáček et al.),
  interval methods written in Intlab, under development

Other languages libraries

- **Int4Sci Toolbox** (by Coprin team, INRIA),
  A Scilab Interface for Interval Analysis
  http://www-sop.inria.fr/coprin/logiciels/Int4Sci/

- **C++ libraries**: C-XSC, PROFIL/BIAS, BOOST interval, FILIB++,...

- **many others**: for Fortran, Pascal, Lisp, Maple, Mathematica,...
G. Alefeld and J. Herzberger.  
*Introduction to Interval Computations.*  

L. Jaulin, M. Kieffer, O. Didrit, and É. Walter.  
*Aplied Interval Analysis.*  

R. E. Moore.  
*Interval Analysis.*  

R. E. Moore, R. B. Kearfott, and M. J. Cloud.  
*Introduction to Interval Analysis.*  

A. Neumaier.  
*Interval Methods for Systems of Equations.*  
1 Motivation

2 Interval Computations

3 Interval Functions

4 Algorithmic Issues
Motivation

- Interval Analysis is not only an exciting theory, but is should be **useful in practice**
- **Useful in practice** = efficient algorithms
- Therefore, theory of algorithms plays an important role

In IntAnal, we meet many notions from Recursion Theory and Complexity Theory. For example:

- non-recursivity (= algorithmic unsolvability)
- NP-completeness, coNP-completeness
- weak and strong polynomiality
- Turing model and real-number computation model
In mathematics, there are many problems which are nonrecursive = not algorithmically solvable at all. Three examples:

### Diophantine equations (Matiyasevich’s Theorem, 1970; Hilbert’s Tenth Problem, 1900)

- **Input:** a polynomial $p(x_1, \ldots, x_9)$ with integer coefficients.
- **Task:** decide whether there exist $x_1^*, \ldots, x_9^* \in \mathbb{Z}$ such that $p(x_1^*, \ldots, x_9^*) = 0$.

### Provability (Gödel’s Theorem, 1931)

- **Input:** a claim (= closed formula in the set-theoretic language) $\varphi$.
- **Task:** decide whether $\varphi$ is provable in Set Theory (say, ZFC).

### Randomness of a coin toss

- **Input:** a finite 0-1 sequence $\gamma$ and a number $K$.
- **Task:** decide whether Kolmogorov complexity of $\gamma$ is greater than $K$. 
The core **nonrecursive problem of Interval Analysis:**

- **Input:** a function $f : \mathbb{R}^n \rightarrow \mathbb{R}$, intervals $x_1, \ldots, x_n$ and $\xi \in \mathbb{R}$.
- **Task:** decide whether $\xi \in f(x_1, \ldots, x_n)$.

**Negative results**

- In general, we cannot determine the range of a function over intervals.
- In general, we cannot determine the interval hull $\Box f(x_1, \ldots, x_n)$.
- In general, we cannot determine an approximation of $\Box f(x_1, \ldots, x_n)$ with a prescribed precision.

**Positive research motivation**

- Find special classes of functions for which determination or approximation of $\Box f(x_1, \ldots, x_n)$ is algorithmically solvable.
- Find special classes of functions for which $\Box f(x_1, \ldots, x_n)$ is **efficiently** computable, i.e. in polynomial computation time.
Proof-idea that “\( \xi \in \text{range}(f) \)” is nonrecursive

- By Matiyasevich we know that given a polynomial \( p(x_1, \ldots, x_9) \), it is nonrecursive to decide whether \( p \) has a integer-valued root.
- Let \( p(x_1, \ldots, x_9) \) be given and consider the function

\[
f(x_1, \ldots, x_n) = p(x_1, \ldots, x_9)^2 + \sum_{i=1}^{9} \sin^2(\pi x_i).
\]

- Now \( 0 \in \text{range}(f) \) iff \( p(x_1, \ldots, x_9) \) has an integer-valued root.

The proof showed an example of a reduction of one problem to another. This is the proof-method for hardness-of-computation results.

We reasoned as follows: if somebody designed an algorithm for the question “\( \xi \in \text{range}(f) \)”, then she would have solved the question “does \( p \) have an integer-valued root?”. But the latter is impossible.
Algorithm: An example of recursive, but “inefficient” problem

Problem formulation
Let $p(x_1, \ldots, x_n)$ be a polynomial over given intervals $x_1, \ldots, x_n$. By continuity we have

\[
\text{range}(p) = \square p = \max\{p(x_1, \ldots, x_n) : x_1 \in x_1, \ldots, x_n \in x_n\},
\]

\[
\text{range}(p) = \Box p = \min\{p(x_1, \ldots, x_n) : x_1 \in x_1, \ldots, x_n \in x_n\}.
\]

Is computation of $\square p$, $\Box p$ recursive?

- Yes.
- Proof-idea: use Tarski’s Quantifier Elimination Method (completeness of the theory of Real Closed Fields).
- But: computation time can be double-exponential.
- So the problem is recursive, but inefficient for practical purposes.
Polynomiality

- **Consensus.** Efficient algorithm = algorithm running in time $p(L)$, where $p$ is a polynomial and $L$ is the bit-size of input.
- **Example.** Interval arithmetic runs in polynomial time.
- **Example.** Linear programming runs is polynomial time (e.g. Ellipsoid Method, IPMs; but **not** the Simplex Method!).
- **Remark.** In numerical problems, the bit-size $L$ involves also *lengths of encodings of binary representations of rational numbers.*
  - Recall that this is a serious issue in linear programming: all known poly-time algorithms for LP are weakly polynomial.
  - So keep in mind: whenever we prove a polynomial-time result in Interval Analysis, which uses LP as a subroutine (which is a frequent case), we have a *weakly polynomial* result.
  - Weak polynomiality of LP is one of Smale’s Millenium Problems for 21st century.
### NP, coNP

- **NP** = a class of YES/NO problems s.t. a YES answer has a short and efficiently verifiable witness.
- **coNP** = a class of YES/NO problems s.t. a NO answer has a short and efficiently verifiable witness.

### Examples

- **CNFSAT:** Is a given boolean formula in conjunctive normal form satisfiable? (NP)
- **TAUT:** Is a given boolean formula tautology? (coNP)
- **TSP:** Given a graph $G$ with weighted edges and a number $K$, does $G$ have a Hamiltonian cycle with length $\leq K$? (NP)
- **KNAPSACK:** Does a given equation $a^T x = b$ with $a \geq 0$ have a 0-1 solution? (NP)
- **ILP:** Does a given inequality system $Ax \leq b$ have an integer solution? (NP, nontrivial proof)
Reductions

- **Informally:** When every instance of a problem $A$ can be written as a particular instance of a problem $B$, then we say that $A$ is *reducible* to $B$. We write $A \leq B$.

- **Example:** $\text{CNFSAT} \leq \text{ILP}$. To illustrate, the CNFSAT-instance
  $$(x_1 \lor \neg x_2 \lor \neg x_3) \land (\neg x_1 \lor x_4 \lor \neg x_5) \land (\neg x_1 \lor \neg x_2)$$
  can be written as the ILP-instance
  $$x_1 + (1 - x_2) + (1 - x_3) \geq 1,$$
  $$(1 - x_1) + x_4 + (1 - x_5) \geq 1,$$
  $$x_i \in \{0, 1\} \quad (\forall i).$$
### Completeness

- \( \mathcal{B} \) is **NP-hard** \iff (\( \forall A \in \text{NP} \)) \( A \leq \mathcal{B} \),
- \( \mathcal{B} \) is **coNP-hard** \iff (\( \forall A \in \text{coNP} \)) \( A \leq \mathcal{B} \),
- \( \mathcal{B} \) is **NP-complete** \iff \( \mathcal{B} \in \text{NP} \) and is NP-hard,
- \( \mathcal{B} \) is **coNP-complete** \iff \( \mathcal{B} \in \text{coNP} \) and is coNP-hard.

### Importance

- For (co)NP-hard (complete) problems we know only \( 2^n \)-algorithms or worse.

- Showing that a problem is (co)-NP hard (complete) is **bad news**: only small instances can be computed.

- Showing that a problem is (co)-NP hard (complete) is **good news for research**: inspect subproblems (special cases) which are tractable; or deal with approximate algorithms.
Generic problems

- Some well-known NP-complete problems: CNFSAT, ILP, TSP.
- Basic coNP-complete problem: TAUT.
- Following Jiří Rohn (our teacher, colleague and a celebrated personality in IntAnal), the following generic NP-complete problem is often used: \textit{given a matrix }$A$, \textit{decide whether there is }$x \in \mathbb{R}^n$ \textit{s.t.}
\[ |Ax| \leq e, \quad \|x\|_1 \geq 1. \]

- $2^n$-algorithm: \( \forall s \in \{-1, 1\}^n \) set \( T_s = \text{diag}(s) \) and solve the LP
\[ -e \leq Ax \leq e, \quad e^T T_s x \geq 1. \]

- Rohn’s NP-completeness result shows that this is “the best” algorithm we can expect.

- The $2^n$-algorithm inspects $\mathbb{R}^n$ orthant-by-orthant; we will meet this orthant decomposition method repeatedly. (REMEMBER THIS!)
**NP-hardness**

- We use “NP-hardness” also for other than YES/NO problems.
- Then we say that a problem \( \mathcal{A} \) is NP-hard if the following holds: *if \( \mathcal{A} \) is solvable in polynomial time, then CNFSAT is solvable in polynomial time (and thus \( P = NP \)).*
- **Example:** given a polynomial \( p(x_1, \ldots, x_n) \),
  - computation of \( p(x_1, \ldots, x_n) \) is NP-hard,
  - computation of \( \overline{p(x_1, \ldots, x_n)} \) is NP-hard.
To recall: the basic problem of Interval Analysis is: *given a function* \( f \) *and intervals* \( x_1, \ldots, x_n \), *determine* \( \square f(x_1, \ldots, x_n) \). Examples from statistics:

**Example: sample mean** \( f \equiv \mu := \frac{1}{n} \sum_{i=1}^{n} x_i \)

- Both \( \square \mu \) and \( \underline{\mu} \) can be computed in polynomial time by interval arithmetic.

**Example: sample variance** \( f \equiv \sigma^2 := \frac{1}{n} \sum_{i=1}^{n} (x_i - \mu)^2 \)

- \( \square \sigma^2 \): polynomial time.
- \( \underline{\sigma^2} \): NP-hard, computable in time \( 2^n \).
- *inapproximability result*: approximate computation of \( \square \sigma^2 \) with an arbitrary absolute error: NP-hard.

**Example: variation coefficient** \( f \equiv t := \frac{\mu}{\sigma} \)

- \( t \): NP-hard, computable in time \( 2^n \).
- *inapproximability result*: approximate computation of \( t \) with an arbitrary absolute error: NP-hard.
- \( \overline{t} \): computable in polynomial time.
Interval linear equations, part I.

Interval Programming 2

Milan Hladík ¹  Michal Černý ²

¹ Faculty of Mathematics and Physics,
  Charles University in Prague, Czech Republic
  http://kam.mff.cuni.cz/~hladik/

² Faculty of Computer Science and Statistics,
  University of Economics, Prague, Czech Republic
  http://nb.vse.cz/~cernym/

Workshop on Interval Programming
7th International Conference of Iranian Operation Research Society
Semnan, Iran, May 12–13, 2014
1. Interval Linear Equations – Solution Concept

2. Enclosure Methods

3. Application: Verification of Real Linear Equations

4. Algorithmic Issues
1 Interval Linear Equations – Solution Concept

2 Enclosure Methods

3 Application: Verification of Real Linear Equations

4 Algorithmic Issues
## Solution Set

### Interval Linear Equations

Let $A \in \mathbb{IR}^{m \times n}$ and $b \in \mathbb{IR}^m$. The family of systems

$$Ax = b, \quad A \in A, \ b \in b.$$ 

is called interval linear equations and abbreviated as $Ax = b$.

### Solution set

The solution set is defined

$$\Sigma := \{x \in \mathbb{R}^n : \exists A \in A \exists b \in b : Ax = b\}.$$

### Important Notice

We do not want to compute $x \in \mathbb{IR}^n$ such that $Ax = b$.

### Theorem (Oettli–Prager, 1964)

The solution set $\Sigma$ is a non-convex polyhedral set described by

$$|A^c x - b^c| \leq A^\Delta |x| + b^\Delta.$$
Proof of the Oettli–Prager Theorem

Let \( x \in \Sigma \), that is, \( Ax = b \) for some \( A \in A \) and \( b \in b \). Now,
\[
\left| A^c x - b^c \right| = \left| (A^c - A)x + (Ax - b) + (b - b^c) \right| = \left| (A^c - A)x + (b - b^c) \right|
\leq \left| A^c - A \right| \left| x \right| + \left| b - b^c \right| \leq A^\Delta \left| x \right| + b^\Delta.
\]
Conversely, let \( x \in \mathbb{R}^n \) satisfy the inequalities. Define \( y \in [-1, 1]^m \) as
\[
y_i = \begin{cases} \frac{(A^c x - b^c)_i}{(A^\Delta |x| + b^\Delta)_i} & \text{if } (A^\Delta |x| + b^\Delta)_i > 0, \\ 1 & \text{otherwise.} \end{cases}
\]
Now, we have \( (A^c x - b^c)_i = y_i (A^\Delta |x| + b^\Delta)_i \), or,
\[
A^c x - b^c = \text{diag}(y)(A^\Delta |x| + b^\Delta).
\]
Define \( z := \text{sgn}(x) \), then \( |x| = \text{diag}(z)x \) and we can write
\[
A^c x - b^c = \text{diag}(y)A^\Delta \text{ diag}(z)x + \text{diag}(y)b^\Delta,
\]
or
\[
(A^c - \text{diag}(y)A^\Delta \text{ diag}(z))x = b^c + \text{diag}(y)b^\Delta.
\]
Example of the Solution Set

Example (Barth & Nuding, 1974))

\[
\begin{pmatrix}
[−1, 2] & [2, 4]
\end{pmatrix}
\begin{pmatrix}
x_1 \\
x_2
\end{pmatrix}
=
\begin{pmatrix}
[−2, 2] \\
[−2, 2]
\end{pmatrix}
\]
Example of the Solution Set

$\begin{pmatrix}
[3, 5] & [1, 3] & -[0, 2] \\
-[0, 2] & [3, 5] & [0, 2] \\
[0, 2] & -[0, 2] & [3, 5]
\end{pmatrix}
\begin{pmatrix}
x_1 \\
x_2 \\
x_3
\end{pmatrix}
=
\begin{pmatrix}
[-1, 1] \\
[-1, 1] \\
[-1, 1]
\end{pmatrix}.$
Proposition

In each orthant, $\Sigma$ is either empty or a convex polyhedral set.

Proof.

Restriction to the orthant given by $s \in \{\pm 1\}^n$:

$$|A^c x - b^c| \leq A^\Delta |x| + b^\Delta, \text{ diag}(s)x \geq 0.$$ 

Since $|x| = \text{diag}(s)x$, we have

$$|A^c x - b^c| \leq A^\Delta \text{ diag}(s)x + b^\Delta, \text{ diag}(s)x \geq 0.$$ 

Using $|a| \leq b \iff a \leq b, -a \leq b$, we get

$$(A^c - A^\Delta \text{ diag}(s))x \leq \bar{b}, \quad (-A^c - A^\Delta \text{ diag}(s))x \leq -\bar{b}, \text{ diag}(s)x \geq 0. \quad \square$$

Corollary

The solutions of $Ax = b, x \geq 0$ is described by $Ax \leq \bar{b}, \bar{A}x \geq b, x \geq 0$. 


Interval Hull $\square \Sigma$

**Goal**

Seeing that $\Sigma$ is complicated, compute $\square \Sigma$ instead.

**First Idea**

Go through all $2^n$ orthants of $\mathbb{R}^n$, determine interval hull of restricted sets (by solving $2n$ linear programs), and then put together.

**Theorem**

If $A$ is regular (each $A \in A$ is nonsingular), $\Sigma$ is bounded and connected.

**Theorem (Jansson, 1997)**

When $\Sigma \neq \emptyset$, then exactly one of the following alternatives holds true:

1. $\Sigma$ is bounded and connected.
2. Each topologically connected component of $\Sigma$ is unbounded.

**Second Idea – Jansson’s Algorithm**

Check the orthant with $(A^c)^{-1}b^c$ and then all the topologically connected.
Prove or Disprove

1. $x \in \Sigma$ if and only if $0 \in Ax - b$,

2. $x \in \Sigma$ if and only if $Ax \cap b \neq \emptyset$. 
Two Basic Polynomial Cases

1. \( A^c = I_n \),
2. \( A \) is inverse nonnegative, i.e., \( A^{-1} \geq 0 \ \forall A \in A \).

Theorem (Kuttler, 1971)

\( A \in \mathbb{IR}^{n \times n} \) is inverse nonnegative if and only if \( A^{-1} \geq 0 \) and \( \overline{A}^{-1} \geq 0 \).

Theorem

Let \( A \in \mathbb{IR}^{n \times n} \) be inverse nonnegative. Then

1. \( \Box \Sigma = [\overline{A}^{-1} b, A^{-1} b] \) when \( b \geq 0 \),
2. \( \Box \Sigma = [A^{-1} b, \overline{A}^{-1} b] \) when \( b \leq 0 \),
3. \( \Box \Sigma = [A^{-1} b, A^{-1} b] \) when \( 0 \in \mathfrak{b} \).

Proof.

Let \( A \in A \) and \( b \in \mathfrak{b} \). Since \( \overline{b} \geq b \geq b \geq 0 \) and \( A^{-1} \geq A^{-1} \geq \overline{A}^{-1} \geq 0 \), we get \( \overline{A}^{-1} b \leq A^{-1} b \leq A^{-1} b \).
Preconditioning

Enclosure

Since \( \Sigma \) is hard to determine and deal with, we seek for enclosures \( x \in \mathbb{R}^n \) such that \( \Sigma \subseteq x \).

Many methods for enclosures exists, usually employ preconditioning.

Preconditioning (Hansen, 1965)

Let \( C \in \mathbb{R}^{n \times n} \). The preconditioned system of equations:

\[
(CA)x = Cb.
\]

Remark

- the solution set of the preconditioned systems contains \( \Sigma \)
- usually, we use \( C \approx (A^c)^{-1} \)
- then we can compute the best enclosure (Hansen, 1992, Bliek, 1992, Rohn, 1993)
Preconditioning

Example (Barth & Nuding, 1974))

\[
\begin{pmatrix}
2 & 4 \\
-1 & 2
\end{pmatrix}
\begin{pmatrix}
-2 & 1 \\
2 & 4
\end{pmatrix}
\begin{pmatrix}
x_1 \\
x_2
\end{pmatrix}
= \begin{pmatrix}
-2 & 2 \\
-2 & 2
\end{pmatrix}
\]

\[
\begin{pmatrix}
2, 4 \\
-1, 2
\end{pmatrix}
\begin{pmatrix}
-2, 1 \\
2, 4
\end{pmatrix}
\begin{pmatrix}
x_1 \\
x_2
\end{pmatrix}
= \begin{pmatrix}
-2, 2 \\
-2, 2
\end{pmatrix}
\]

\[
\begin{pmatrix}
-14 & -7 \\
7 & 14
\end{pmatrix}
\begin{pmatrix}
-2 & 2 \\
-2 & 2
\end{pmatrix}
\begin{pmatrix}
x_1 \\
x_2
\end{pmatrix}
= \begin{pmatrix}
-2, 2 \\
-2, 2
\end{pmatrix}
\]

\[
\begin{pmatrix}
14 & 14 \\
14 & -14
\end{pmatrix}
\begin{pmatrix}
-2 & 2 \\
-2 & 2
\end{pmatrix}
\begin{pmatrix}
x_1 \\
x_2
\end{pmatrix}
= \begin{pmatrix}
-2, 2 \\
-2, 2
\end{pmatrix}
\]
Preconditioning

Example (typical case)

\[
\begin{pmatrix}
[6, 7] & [2, 3] \\
[1, 2] & -[4, 5]
\end{pmatrix}
\begin{pmatrix}
x_1 \\
x_2
\end{pmatrix}
=
\begin{pmatrix}
[6, 8] \\
- [7, 9]
\end{pmatrix}
\]
Interval Gaussian elimination = Gaussian elimination + interval arithmetic.

Example (Barth & Nuding, 1974))

\[
\begin{pmatrix}
[2, 4] & [-2, 1] \\
[-1, 2] & [2, 4]
\end{pmatrix}
\begin{pmatrix}
x_1 \\
x_2
\end{pmatrix} =
\begin{pmatrix}
[-2, 2] \\
[-2, 2]
\end{pmatrix}
\]

Then we proceed as follows

\[
\begin{pmatrix}
[-1, 2] & [2, 4] & [-2, 2]
\end{pmatrix}
\begin{pmatrix}
[2, 4] \\
[-2, 1] \\
[-2, 2]
\end{pmatrix} \sim
\begin{pmatrix}
[2, 4] \\
0 \\
[1, 6]
\end{pmatrix}
\begin{pmatrix}
[-2, 2] \\
[-4, 4]
\end{pmatrix}.
\]

By back substitution, we compute

\[
x_2 = [-4, 4],
\]

\[
x_1 = \left( [-2, 2] - [-2, 1] \cdot [-4, 4] \right) / [2, 4] = [-5, 5].
\]
Interval Jacobi and Gauss-Seidel Iterations

Idea

From the $i$th equation of $Ax = b$ we get

$$x_i = \frac{1}{a_{ii}} \left( b_i - \sum_{j=1}^{i-1} a_{ij} x_j - \sum_{j=i+1}^{n} a_{ij} x_j \right).$$

If $x^0 \supseteq \Sigma$ is an initial enclosure, then

$$x_i \in \frac{1}{a_{ii}} \left( b_i - \sum_{j \neq i} a_{ij} x_j^0 \right), \quad \forall x \in \Sigma.$$ 

Thus, we can tighten the enclosure by iterations

Interval Jacobi / Gauss–Seidel Iterations ($k = 1, 2, \ldots$)

1: for $i = 1, \ldots, n$ do
2: \quad $x_i^k := \frac{1}{a_{ii}} \left( b_i - \sum_{j \neq i} a_{ij} x_j^{k-1} \right) \cap x_i^{k-1};$
3: end for
Krawczyk Iterations

Krawczyk operator

Krawczyk operator $K : \mathbb{IR}^n \rightarrow \mathbb{IR}^n$ reads

\[ K(x) := Cb + (I_n - CA)x \]

Proposition

If $x \in x \cap \Sigma$, then $x \in K(x)$.

Proof.

Let $x \in x \cap \Sigma$, so $Ax = b$ for some $A \in A$ and $b \in b$. Thus $CAx = Cb$, whence $x = Cb + (I_n - CA)x \in Cb + (I_n - CA)x = K(x)$. 

Krawczyk Iterations

Let $x^0 \supseteq \Sigma$ is an initial enclosure, and iterate ($k = 1, 2, \ldots$):

1: $x^k := K(x^{k-1}) \cap x^{k-1}$;
Theorem

Let \( x \in \mathbb{IR}^n \) and \( C \in \mathbb{R}^{n \times n} \). If

\[
K(x) = Cb + (I - CA)x \subseteq \text{int} x,
\]

then \( C \) is nonsingular, \( A \) is regular, and \( \Sigma \subseteq x \).

Proof.

Existence of a solution based on Brouwer’s fixed-point theorem. Nonsingularity and uniqueness based on the Perron–Frobenius theory.

Remark

- A reverse iteration method to the Krawczyk method.
- It starts with a small box around \((A^c)^{-1}b^c\), and then interatively inflates the box.
- Implemented in Intlab v. 6.
1 Interval Linear Equations – Solution Concept

2 Enclosure Methods

3 Application: Verification of Real Linear Equations

4 Algorithmic Issues
Verification of Real Linear Equations

Problem formulation
Given a real system $Ax = b$ and $x^*$ approximate solution, find $y \in \mathbb{R}^n$ such that $A^{-1}b \in x^* + y$.

Example

\[
\begin{align*}
\begin{array}{c}
\text{x}_1 \\
\text{x}_2 \\
\end{array} \\
\end{align*}
\]
Theorem

Let \( y \in \mathbb{IR}^n \) and \( C \in \mathbb{R}^{n \times n} \). If

\[
C(b - Ax^*) + (I - CA)y \subseteq \text{int } y,
\]

then \( C \) and \( A \) are nonsingular, and \( A^{-1}b \in x^* + y \).

Proof.

Substitute \( x := y + x^* \), and apply the \( \varepsilon \)-inflation method for the system

\[
Ay = b - Ax^*.
\]

\( \varepsilon \)-inflation method (Caprani and Madsen, 1978, Rump, 1980)

Repeat inflating \( y := [0.9, 1.1]x + 10^{-20}[-1, 1] \) and updating

\[
x := C(b - Ax^*) + (I - CA)y
\]

until \( x \subseteq \text{int } y \).

Then, \( \Sigma \subseteq x^* + x \).
Verification of Real Linear Equations

Example

Let $A$ be the Hilbert matrix of size 10 (i.e., $a_{ij} = \frac{1}{i+j-1}$), and $b := Ae$. Then $Ax = b$ has the solution $x = e = (1, \ldots, 1)^T$.

Approximate solution by Matlab:

<table>
<thead>
<tr>
<th>Approximate solution by Matlab:</th>
<th>Enclosing interval by $\varepsilon$-inflation method (2 iterations):</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.999999999235452</td>
<td>$[0.99999973843401, 1.00000026238575]$</td>
</tr>
<tr>
<td>1.000000065575364</td>
<td>$[0.99999843048508, 1.00000149895660]$</td>
</tr>
<tr>
<td>0.999998607887449</td>
<td>$[0.9997745481481, 1.00002404324710]$</td>
</tr>
<tr>
<td>1.000012638750021</td>
<td>$[0.9998166603900, 1.00020478046370]$</td>
</tr>
<tr>
<td>0.999939734980300</td>
<td>$[0.99902374408278, 1.00104070076742]$</td>
</tr>
<tr>
<td>1.000165704992114</td>
<td>$[0.99714060702796, 1.00268292103727]$</td>
</tr>
<tr>
<td>0.999727989024899</td>
<td>$[0.99559932282378, 1.00468935360003]$</td>
</tr>
<tr>
<td>1.000263042205847</td>
<td>$[0.99546972629357, 1.00425202249136]$</td>
</tr>
<tr>
<td>0.999861803020249</td>
<td>$[0.99776781605377, 1.00237789028988]$</td>
</tr>
<tr>
<td>1.000030414871015</td>
<td>$[0.99947719419921, 1.00049082925529]$</td>
</tr>
</tbody>
</table>
References

G. Alefeld and J. Herzberger.  
*Introduction to Interval Computations.*  

*Linear optimization problems with inexact data.*  

R. E. Moore, R. B. Kearfott, and M. J. Cloud.  
*Introduction to interval analysis.*  

A. Neumaier.  
*Interval methods for systems of equations.*  

J. Rohn.  
A handbook of results on interval linear problems.  

S. M. Rump.  
Verification methods: Rigorous results using floating-point arithmetic.  
1. Interval Linear Equations – Solution Concept

2. Enclosure Methods

3. Application: Verification of Real Linear Equations

4. Algorithmic Issues
Algorithmic Issues: Solvability of $Ax = b$

To recall:

- System $Ax = b$ is solvable iff $(\exists A \in A)(\exists b \in b)(\exists x \in \mathbb{R}^n) \ Ax = b$.
- Solution set is defined by
  \[ \Sigma(A, b) = \bigcup_{A \in A, \ b \in b} \{x \in \mathbb{R}^n : Ax = b\}. \]

Theorem

Checking solvability is an NP-complete problem.

Outline: we must prove (1) NP-hardness and (2) presence in NP.

Proof

Step 1. Proof of NP-hardness. We will show that Rohn’s generic problem of solvability of $-e \leq Ax \leq e, \|x\|_1 \geq 1$ is reducible to checking solvability of a particular system $Ax = b$.

(Informally: if somebody manages to design an efficient method for checking solvability, then she also managed to solve the Rohn’s generic problem; but it is impossible unless $P = NP$.)
Proof of NP-completeness of $\Sigma(A, b) \neq \emptyset$. Step 1 continued

**Claim:** Rohn’s system $-e \leq Ax \leq e, \|x\|_1 \geq 1$ is solvable iff

$$[A, A]x = [-e, e], \quad [-e^T, e^T]x = [1, 1] \quad (1)$$

is solvable. Thus, if we have an efficient method for (1), then we have an efficient method for Rohn’s system, which is NP-complete. This proves NP-hardness.

**Proof of claim**

- If $x$ solves Rohn’s system, then $x' := \frac{x}{\|x\|_1}$ solves (1). [*Proof.* $|Ax'| = \frac{1}{\|x\|_1} |Ax| \leq |Ax| \leq e$; thus $x'$ solves $Ax' = [-e, e]$. In addition, $\|x'\|_1 = 1$; thus $\text{sgn}(x')^T x' = 1$ and $\text{sgn}(x') \in [-e, e].$]

- If $x$ solves $Ax = b, c^T x = 1$ with $b \in [-e, e]$ and $c \in [-e, e]$, then $|Ax| = |b| \leq e$ and $\|x\|_1 = e^T |x| \geq |c|^T |x| \geq c^T x = 1$. QED
Proof of NP-completeness of $\Sigma(A, b) \not= \emptyset$. Step 2

Step 2. Proof that the problem is in NP.

- Intuitively, any pair $(A_0, b_0)$ s.t.
  \[
  A_0 \in A, \quad b_0 \in b, \quad \{x : A_0x = b_0\} \neq \emptyset
  \]
  (2)
  could serve as an NP-witness. (Observe that the conditions (2) can be verified in polynomial time.)

- However, there is a technical problem: the NP-witness must have polynomial size. In other words: we must prove that there exists a polynomial $p$ s.t. $\text{bitsize}(A_0, b_0) \leq p(\text{bitsize}(A, \overline{A}, b, \overline{b}))$.

- We proceed otherwise.
We use Oettli-Prager: we know that

\[ \Sigma(A, b) \cap \mathbb{R}^n_s = \{ x \in \mathbb{R}^n : -A^ΔT_s x - b^Δ \leq A^c x - b^c \leq A^ΔT_s x + b^Δ, \ T_s x \geq 0 \} \]

where \( s \in \{-1, 1\}^n \), \( T_s = \text{diag}(s) \) and \( \mathbb{R}^n_s = \{ x \in \mathbb{R}^n : T_s x \geq 0 \} \).

Given \( s \), nonemptiness of the polyhedron (\( \star \)) can be checked in polynomial time by LP.

Clearly, bitsize of \( s \) is bounded by \( \text{bitsize}(A, \overline{A}, b, \overline{b}) \).

Thus, \( s \) s.t. \( \Sigma \cap \mathbb{R}^n_s \neq \emptyset \) is a valid NP-witness for the fact \( \Sigma \neq \emptyset \). \( \square \)
To recall: the solution set is defined as

\[ \Sigma = \bigcup_{A \in \mathbf{A}, \ b \in \mathbf{b}} \{x \in \mathbb{R}^n : Ax = b\} . \]

**Theorem**

*Deciding whether \( \Sigma \) is bounded is a coNP-complete problem.*

**Proof idea of coNP-hardness.**

Consider the system \( \mathbf{A} x = 0 \). Then \( \Sigma = \Sigma(\mathbf{A}, [0, 0]) \) is unbounded iff

\[ \text{there is } A \in \mathbf{A} \text{ which is singular.} \quad (3) \]

Later we will show that deciding (3) is an NP-complete problem. So checking unboundedness of \( \Sigma \) is NP-hard, and checking boundedness is coNP-hard.
AlgoIss: Computation of $\square \Sigma$

Some consequences

- Every *exact* enclosure method (i.e. every method computing $\square \Sigma$ and $\square \Sigma$ exactly) must be implicitly able to detect (at least) the following “extreme” cases:
  - $\Sigma = \emptyset$,
  - $\Sigma$ is unbounded.

- Thus, any enclosure method *must* be able to solve two NP-complete problems. Thus $\square \Sigma$ is NP-hard.

- So we cannot expect that the $2^n$-method, based on orthant decomposition by Oettli-Prager, could be significantly improved.

Further results

- The basic results on hardness-of-computation of $\square \Sigma$ can be pushed further: it holds that even *approximate* computation of $\square \Sigma$ with a given absolute error or relative error is NP-hard.

- So, in theory, even “not too redundant” enclosures are hard to compute.
Interval linear equations, part II.

Interval Programming 3

Milan Hladík ¹  Michal Černý ²

¹ Faculty of Mathematics and Physics, Charles University in Prague, Czech Republic
   http://kam.mff.cuni.cz/~hladik/

² Faculty of Computer Science and Statistics, University of Economics, Prague, Czech Republic
   http://nb.vse.cz/~cernym/

Workshop on Interval Programming
7th International Conference of Iranian Operation Research Society
Semnan, Iran, May 12–13, 2014
1. Regularity of Interval Matrices
2. Parametric Interval Systems
3. AE Solution Set
4. Algorithmic Issues
1 Regularity of Interval Matrices

2 Parametric Interval Systems

3 AE Solution Set

4 Algorithmic Issues
Definition (Regularity)

A ∈ IR^{n×n} is regular if each A ∈ A is nonsingular.

Theorem

Checking regularity of an interval matrix is co-NP-hard.

Forty necessary and sufficient conditions for regularity of A by Rohn (2010):

1. The system |Acx| ≤ AΔ|x| has the only solution x = 0.
2. det(Ac − diag(y)AΔ diag(z)) is constantly either positive or negative for each y, z ∈ {±1}^n.
3. For each y ∈ {±1}^n, the system Acx − diag(y)AΔ|x| = y has a solution.
4. ...
Theorem (Beeck, 1975)

If $\rho((A^c)^{-1}|A^\Delta) < 1$, then $A$ is regular.

Proof.

Precondition $A$ by the midpoint inverse: $M := (A^c)^{-1}A$. Now,

$$M^c = I_n, \quad M^\Delta = |(A^c)^{-1}|A^\Delta,$$

and for each $M \in M$ we have

$$|M - M^c| = |M - I_n| \leq M^\Delta.$$

From the theory of eigenvalues of nonnegative matrices it follows

$$\rho(M - I_n) \leq \rho(M^\Delta) < 1,$$

so $M$ has no zero eigenvalue and is nonsingular.

Necessary Condition

If $0 \in Ax$ for some $0 \neq x \in \mathbb{R}^n$, then $A$ is not regular. (Try $x := (A^c)^{-1}_i$)
The following conditions are necessary for the regularity of $A$. Decide which of them are sufficient as well:

1. all matrices $A$ of the form $a_{ij} \in \{a_{ij}, \overline{a}_{ij}\}$ are nonsingular,
2. all matrices $A$ of the form $a_{ij} \in \{a_{ij}, \overline{a}_{ij}\}$, and $A^c$ are nonsingular.
1 Regularity of Interval Matrices

2 Parametric Interval Systems

3 AE Solution Set

4 Algorithmic Issues
Parametric Interval Systems

\[ A(p)x = b(p), \]
where the entries of \( A(p) \) and \( b(p) \) depend on parameters \( p_1 \in \mathbf{p}_1, \ldots, p_K \in \mathbf{p}_K. \)

Definition (Solution Set)

\[ \Sigma_p = \{ x \in \mathbb{R}^n : A(p)x = b(p) \text{ for some } p \in \mathbf{p} \}. \]

Relaxation

Compute (enclosures of) the ranges \( A := A(p) \) and \( b := b(p) \) and solve \( Ax = b. \)

May overestimate a lot!
Special Case: Parametric Linear Interval Systems

\[ A(p)x = b(p), \]

where

\[ A(p) = \sum_{k=1}^{K} A_k p_k, \quad b(p) = \sum_{k=1}^{K} b_k p_k \]

and \( p \in \mathbf{P} \) for some given interval vector \( \mathbf{p} \in \mathbb{R}^K \), matrices \( A_1, \ldots, A_K \in \mathbb{R}^{n \times n} \) and vectors \( b_1, \ldots, b_n \in \mathbb{R}^n \).

Remark

It covers many structured matrices: symmetric, skew-symmetric, Toeplitz or Hankel.
Example (Displacements of a truss structure (Skalna, 2006))

The 7-bar truss structure subject to downward force. The stiffnesses $s_{ij}$ of bars are uncertain. The displacements $d$ of the nodes, are solutions of the system $Kd = f$, where $f$ is the vector of forces.
Example (Displacements of a truss structure (Skalna, 2006))

The 7-bar truss structure subject to downward force.

The stiffnesses $s_{ij}$ of bars are uncertain.

The displacements $d$ of the nodes, are solutions of the system $Kd = f$, where $f$ is the vector of forces.

$$
K = \begin{pmatrix}
\frac{s_{12}}{2} + s_{13} & -\frac{s_{12}}{2} & -\frac{s_{12}}{2} & -s_{13} & 0 & 0 & 0 \\
-s_{21} & \frac{s_{21} + s_{23}}{2} + s_{24} & \frac{s_{21} - s_{23}}{2} & -\frac{s_{23}}{2} & -\frac{s_{23}}{2} & -s_{24} & 0 \\
2 & 2 & -s_{31} & -s_{32} & 2 & -s_{34} & 0 \\
-2 & 2 & -\frac{s_{32}}{2} & -s_{32} & 2 & -s_{34} & 0 \\
0 & 2 & 0 & 2 & -s_{42} & 0 & 0 \\
0 & 0 & 0 & 0 & -\frac{s_{43}}{2} & -\frac{s_{43}}{2} & -s_{45} \\
0 & 0 & 0 & 0 & 0 & -\frac{s_{43} + s_{45}}{2} & 0 \\
\end{pmatrix}
$$
Example

\[
\begin{pmatrix}
1 - 2p & 1 \\
2 & 4p - 1
\end{pmatrix}
\begin{pmatrix}
x_1 \\
x_2
\end{pmatrix}
= 
\begin{pmatrix}
7p - 9 \\
3 - 2p
\end{pmatrix}, \quad p \in \mathbb{R} = [0, 1].
\]
**Theorem**

If \( x \in \Sigma_p \), then it solves

\[
|A(p^c)x - b(p^c)| \leq \sum_{k=1}^{K} p_k^A |A^k x - b^k|.
\]

**Proof.**

\[
|A(p^c)x - b(p^c)| = \left| \sum_{k=1}^{K} p_k^c (A^k x - b^k) \right| = \left| \sum_{k=1}^{K} p_k^c (A^k x - b^k) - \sum_{k=1}^{K} p_k (A^k x - b^k) \right|
\]

\[
= \left| \sum_{k=1}^{K} (p_k^c - p_k) (A^k x - b^k) \right| \leq \sum_{k=1}^{K} |p_k^c - p_k| |A^k x - b^k| \leq \sum_{k=1}^{K} p_k^A |A^k x - b^k|. \quad \Box
\]

- Popova (2009) showed that it is the complete characterization of \( \Sigma_p \) as long as no interval parameter appears in more than one equation.
- Checking \( x \in \Sigma_p \) for a given \( x \in \mathbb{R}^n \) is a polynomial problem via linear programming.
Relaxation and Preconditioning – First Idea

Evaluate $\mathbf{A} := A(p)$, $\mathbf{b} := b(p)$, choose $C \in \mathbb{R}^{n \times n}$ and solve

$$(C\mathbf{A})\mathbf{x} = C\mathbf{b}.$$ 

Relaxation and Preconditioning – Second Idea

Solve $\mathbf{A'} \mathbf{x} = \mathbf{b'}$, where

$$\mathbf{A'} := \sum_{k=1}^{K} (CA^k)p_k, \quad \mathbf{b'} := \sum_{k=1}^{K} (Cb^k)p_k.$$ 

Second Idea is Provably Better

Due to sub-distributivity law,

$$\mathbf{A'} := \sum_{k=1}^{K} (CA^k)p_k \subseteq C\left(\sum_{k=1}^{K} A^k p_k\right) = (C\mathbf{A}).$$
The Symmetric Solution Set of \(Ax = b\)

\[\{x \in \mathbb{R}^n : Ax = b \text{ for some symmetric } A \in A \text{ and } b \in b\}\].

Described by \(\frac{1}{2}(4^n - 3^n - 2 \cdot 2^n + 3) + n\) nonlinear inequalities (H., 2008).

**Example**

\[
A = \begin{pmatrix}
[1, 2] & [0, a] \\
[0, a] & -1
\end{pmatrix}, \quad b = \begin{pmatrix} 2 \end{pmatrix}.
\]

\[
A = \begin{pmatrix}
-1 & [-5, 5] \\
[-5, 5] & 1
\end{pmatrix}, \quad b = \begin{pmatrix} 1 \\
[1, 3]
\end{pmatrix}.
\]
Least Square Solution

Let \( A \in \mathbb{IR}^{m \times n} \), \( b \in \mathbb{IR}^{m} \) and \( m > n \). The least square solution of

\[
Ax = b,
\]

is defined as the optimal solution of

\[
\min_{x \in \mathbb{R}^{n}} \| Ax - b \|_{2},
\]

or, alternatively as the solution to

\[
A^{T}Ax = A^{T}b.
\]

Interval Least Square Solution Set

Let \( A \in \mathbb{IR}^{m \times n} \) and \( b \in \mathbb{IR}^{m} \) and \( m > n \). The LSQ solution set is defined

\[
\Sigma_{LSQ} := \{ x \in \mathbb{R}^{n} : \exists A \in A \exists b \in b : A^{T}Ax = A^{T}b \}.
\]

Proposition

\( \Sigma_{LSQ} \) is contained in the solution set to \( A^{T}Ax = A^{T}b \).
Proposition

Σ_{LSQ} is contained in the solution set to

\[
\begin{pmatrix}
0 & A^T \\
A & I_m
\end{pmatrix}
\begin{pmatrix}
x \\
y
\end{pmatrix}
=
\begin{pmatrix}
0 \\
b
\end{pmatrix}.
\] (1)

Proof.

Let \( A \in \mathbf{A}, \ b \in \mathbf{b}. \) If \( x, y \) solve

\[
A^T y = 0, \quad Ax + y = b,
\]

then

\[
0 = A^T (b - Ax) = A^T b - A^T Ax,
\]

and vice versa.

Proposition

Relaxing the dependencies, the solution set to \( A^T Ax = A^T b \) is contained in the solution set to (1).
1 Regularity of Interval Matrices
2 Parametric Interval Systems
3 AE Solution Set
4 Algorithmic Issues
Tolerable Solutions

Motivation
So far, existentially quantified interval systems
\[ \Sigma := \{ x \in \mathbb{R}^n : \exists A \in A \exists b \in b : Ax = b \}. \]
Now, incorporate universal quantification as well!

Definition (Tolerable Solutions)
A vector \( x \in \mathbb{R}^n \) is a tolerable solution to \( Ax = b \) if for each \( A \in A \) there is \( b \in b \) such that \( Ax = b \).

In other words,
\[ \forall A \in A \exists b \in b : Ax = b. \]

Equivalent Characterizations
- \( Ax \subseteq b \),
- \( |A^c x - b^c| \leq -A^\Delta |x| + b^\Delta. \)
A vector $x \in \mathbb{R}^n$ is a tolerable solution if and only if $x = x_1 - x_2$, where

$$\overline{A}x_1 - \underline{A}x_2 \leq \overline{b}, \quad A x_1 - \overline{A}x_2 \geq b,$$

$x_1, x_2 \geq 0$.

Proof.

“$\Leftarrow$” Let $A \in \mathbb{A}$. Then

$$Ax = A x_1 - A x_2 \leq \overline{A} x_1 - \underline{A} x_2 \leq \overline{b},$$

$$Ax = A x_1 - A x_2 \geq \underline{A} x_1 - \overline{A} x_2 \geq b$$

Thus, $Ax \in \mathbb{b}$ and $Ax = b$ for some $b \in \mathbb{b}$.

“$\Rightarrow$” Let $x \in \mathbb{R}^n$ be a tolerable solution. Define $x_1 := \max\{x, 0\}$ and $x_2 := \max\{-x, 0\}$ the positive and negative part of $x$, respectively. Then

$x = x_1 - x_2$, $|x| = x_1 + x_2$, and $|A^c x - b^c| \leq -A^\Delta |x| + b^\Delta$ draws

$$A^c (x_1 - x_2) - b^c \leq -A^\Delta (x_1 + x_2) + b^\Delta,$$

$$-A^c (x_1 - x_2) + b^c \leq -A^\Delta (x_1 + x_2) + b^\Delta.$$
Example (Leontief’s Input–Output Model of Economics)

- economy with $n$ sectors (e.g., agriculture, industry, transportation, etc.),
- sector $i$ produces a single commodity of amount $x_i$,
- production of each unit of the $j$th commodity will require $a_{ij}$ (amount) of the $i$th commodity
- $d_i$ the final demand in sector $i$.

Now the model draws

$$x_i = a_{i1}x_1 + \cdots + a_{in}x_n + d_i.$$ 

or, in a matrix form

$$x = Ax + d.$$ 

The solution $x = (I_n - A)^{-1}d = \sum_{k=0}^{\infty} A^k d$ is nonnegative if $\rho(A) < 1$.

**Question:** Exists $x$ such that for any $A \in A$ there is $d \in d$: $(I_n - A)x = d$?
Quantified system $Ax = b$

- each interval parameter $a_{ij}$ and $b_i$ is quantified by $\forall$ or $\exists$
- the universally quantified parameters are denoted by $A^\forall$, $b^\forall$,
- the existentially quantified parameters are denoted by $A^\exists$, $b^\exists$
- the system reads $(A^\forall + A^\exists)x = b^\forall + b^\exists$

Definition (AE Solution Set)

$$
\Sigma_{AE} := \{ x \in \mathbb{R}^n : \forall A^\forall \in A^\forall \forall b^\forall \in b^\forall \exists A^\exists \in A^\exists \exists b^\exists \in b^\exists : (A^\forall + A^\exists)x = b^\forall + b^\exists \}.
$$
Theorem (Shary, 1995)

$$\Sigma_{AE} = \{ x \in \mathbb{R}^n : A^\forall x - b^\forall \subseteq b^\exists - A^\exists x \}. \quad (2)$$

Proof.

$$\Sigma_{AE} = \{ x \in \mathbb{R}^n : \forall A^\forall \in A^\forall \forall b^\forall \in b^\forall \exists A^\exists \in A^\exists \exists b^\exists \in b^\exists : A^\forall x - b^\forall = b^\exists - A^\exists x \}$$

$$= \{ x \in \mathbb{R}^n : \forall A^\forall \in A^\forall \forall b^\forall \in b^\forall : A^\forall x - b^\forall \subseteq b^\exists - A^\exists x \}$$

$$= \{ x \in \mathbb{R}^n : A^\forall x - b^\forall \subseteq b^\exists - A^\exists x \}. \quad \square$$

Theorem (Rohn, 1996)

$$\Sigma_{AE} = \{ x \in \mathbb{R}^n : |A^c x - b^c| \leq ((A^\exists)^\Delta - (A^\forall)^\Delta)|x| + (b^\exists)^\Delta - (b^\forall)^\Delta \}. \quad (2)$$

Proof.

Using (2) and the fact $$p^c \subseteq q^c \iff |p^c - q^c| \leq p^\Delta - q^\Delta$$, we get

$$|(A^\forall x - b^\forall)^c - (b^\exists - A^\exists x)^c| \leq (A^\exists x - b^\exists)^\Delta - (b^\forall - A^\forall x)^\Delta$$

$$= (A^\exists)^\Delta|x| + b^\exists^\Delta - (A^\forall)^\Delta x| - b^\forall^\Delta. \quad \square$$
Exercises for YOU

Strong solution of $Ax = b$

Characterize when $x \in \mathbb{R}^n$ solves $Ax = b$ for every $A \in A$ and $b \in b$. 
### webComputing (by E. Popova)

- Parametric solution set
- AE solution set
- 3D standard solution set

### Parametric interval systems

- Mathematica package (Popova, 2004)
- C++ library C-XCS implementation (Popova and Krämer, 2007; Zimmer, Krämer and Popova, 2012)
*Linear optimization problems with inexact data.*  

M. Hladík.  
Enclosures for the solution set of parametric interval linear systems.  

E. D. Popova.  
Explicit description of AE solution sets for parametric linear systems.  

J. Rohn.  
Forty necessary and sufficient conditions for regularity of interval matrices: A survey.  

S. M. Rump.  
Verification methods: Rigorous results using floating-point arithmetic.  
1 Regularity of Interval Matrices
2 Parametric Interval Systems
3 AE Solution Set
4 Algorithmic Issues
Algorithmic Issues: Various solution concepts

Various solution concepts of $Ax = b$

- **Traditional solution concept:** $\exists x_0 \exists A \exists b$-concept
  - Solvability $\iff (\exists x_0 \in \mathbb{R}^n)(\exists A \in A)(\exists b \in b) Ax_0 = b$
  - We proved: checking solvability is \text{NP-complete}
  - But we know: checking \text{nonnegative} solvability — \text{polynomial time}

- **Strong solvability:** $\forall A \forall b \exists x_0$-concept
  - Strong solvability $\iff (\forall A \in A)(\forall b \in b)(\exists x_0 \in \mathbb{R}^n) Ax_0 = b$
  - Complexity: \text{coNP-complete}
  - Remains \text{coNP-complete} even if we restrict to $x_0 \geq 0$

- **Strong solution:** $\exists x_0 \forall A \forall b$-concept
  - $x_0 \in \mathbb{R}^n$ is a \textit{strong solution} if $(\forall A \in A)(\forall b \in b) Ax_0 = b$
  - Existence of a strong solution $\iff (\exists x_0 \in \mathbb{R}^n)(\forall A \in A)(\forall b \in b) Ax_0 = b$
  - Complexity of testing existence: \text{polynomial time}
  - \textbf{Remark.} Strong solutions exist very rarely; for example, a necessary condition for existence is $b^\Delta = 0$ (Exercise)
  - \textbf{Caution.} In case of linear \textit{inequalities}, the situation is different: a system $Ax \leq b$ is strongly solvable iff it has a strong solution. But nothing similar holds for equations...
Various solution concepts of $Ax = b$

- **Tolerable solution:** $\exists x_0 \forall A \exists b$-concept
  - Existence of a tolerable solution $\iff (\exists x_0 \in \mathbb{R}^n)(\forall A \in A)(\exists b \in b) \ Ax_0 = b$.
  - Complexity: polynomial time

- **Control solution:** $\exists x_0 \forall b \exists A$-concept
  - Existence of a control solution $\iff (\exists x_0 \in \mathbb{R}^n)(\forall b \in b)(\exists A \in A) \ Ax_0 = b$.
  - Complexity: NP-complete

- **AE-solution:** $\exists x_0 \forall A \forall b \exists A^\exists \exists b^\exists$-concept
  - Existence of AE-solution $\iff (\exists x_0 \in \mathbb{R}^n)(\forall A^\forall \in A^\forall)(\forall b^\forall \in b^\forall)(\exists A^\exists \in A^\exists)(\exists b^\exists \in b^\exists) \ (A^\forall + A^\exists)x_0 = b^\forall + b^\exists$.
  - Complexity: NP-complete
  - To recall: $2^n$-algorithm — orthant decomposition by Rohn’s Theorem
A natural generalization

One can imagine a natural generalization to any level of quantifier complexity, e.g.
- $\Sigma_k$-solution: $\exists \forall \exists \cdots$ with $k – 1$ quantifier alternations,
- $\Pi_k$-solution: $\forall \exists \forall \cdots$ with $k – 1$ quantifier alternations.

Study of formulae with $\Sigma_k$- and $\Pi_k$-prefixes is popular in logic (recall e.g. the Arithmetical Hierarchy) as well as in Complexity Theory (recall e.g. the Polynomial Time Hierarchy).

About $\Sigma_k$- and $\Pi_k$-solutions we can say only that checking existence is recursive: can be decided (in double-exponential time) via Tarski’s Quantifier Elimination Method.

But possibly more could be said and more efficient methods might exist...

If logic and complexity theory “like” building hierarchies based on quantifier complexity, why couldn’t we try something similar in interval analysis?
Algorithm: Regularity

Let $E$ denote the all-one matrix.

**Proposition**

The following statements are equivalent:

(a) Rohn’s system $|Ax| \leq e, \|x\|_1 \geq 1$ is solvable.

(b) Interval system $[A - E, A + E]x = 0, [-e^T, e^T]x = 1$ has a solution.

(c) $[A - E, A + E]$ is singular (= contains a singular matrix).

**Corollary**

(a) Checking regularity of an interval matrix is a coNP-complete problem.

(b) Checking existence of a solution of an interval system $Ax = b$ is an NP-complete problem. (This is another proof of a previously proved statement.)

(c) Checking existence of a control solution of an interval system $Ax = b$ is an NP-complete problem.
Proof of Proposition

Step 1. Singularity of \([A - E, A + E] \iff \text{solvability of } [A - E, A + E]x = 0, [-e^T, e^T]x = 1.\)

- \(A' \in [A - E, A + E] \) is singular \(\iff A'x = 0, \|x\|_1 = 1 \) is solvable \(\iff A'x = 0, \text{sgn}(x)^T x = 1 \) is solvable. Now \(A' \in [A - E, A + E], \text{sgn}(x)^T \in [-e^T, e^T].\)
- \(A'x = 0, c^T x = 1 \) is solvable for \(A' \in [A - E, A + E], c^T \in [-e^T, e^T]\)
  \(\Rightarrow \) there is a solution \(x \neq 0 \Rightarrow A' \) is singular.

Step 2. Solvability of \(|Ax| \leq e, \|x\|_1 \geq 1 \iff \text{solvability of } [A - E, A + E]x = 0, [-e^T, e^T]x = 1.\)

- \(x \) solves \(|Ax| \leq e, \|x\|_1 \geq 1 \iff x' := \frac{x}{\|x\|_1} \) solves

\[
\left| \begin{pmatrix} A \\ 0^T \end{pmatrix} x - \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right| \leq \begin{pmatrix} E \\ e^T \end{pmatrix} |x|.
\]

The last inequality is Oettli-Prager expression for the solution set of \([A - E, A + E]x = 0, [-e^T, e^T]x = 1.\)
Exercise

Prove in detail that checking regularity of a given interval matrix is indeed in coNP.
Interval linear inequalities

Interval Programming 4

Milan Hladík ¹  Michal Černý ²

¹ Faculty of Mathematics and Physics, Charles University in Prague, Czech Republic
   http://kam.mff.cuni.cz/~hladik/

² Faculty of Computer Science and Statistics, University of Economics, Prague, Czech Republic
   http://nb.vse.cz/~cernym/

Workshop on Interval Programming
7th International Conference of Iranian Operation Research Society
Semnan, Iran, May 12–13, 2014
Outline

1. Software Presentation
2. Interval Linear Inequalities – Solution Set
3. Algorithmic Issues
1 Software Presentation

2 Interval Linear Inequalities – Solution Set

3 Algorithmic Issues
- **webComputing** (E. Popova)
  visualization of solution sets

- **Intlab** (S. M. Rump)
  interval library for Matlab
Interval Linear Inequalities

Let \( A \in \mathbb{IR}^{m \times n} \) and \( b \in \mathbb{IR}^m \). The family of systems

\[
Ax \leq b, \quad A \in A, \ b \in b.
\]

is called interval linear inequalities and abbreviated as \( Ax \leq b \).

Solution set

The solution set is defined

\[
\Sigma := \{ x \in \mathbb{R}^n : \exists A \in A \exists b \in b : Ax \leq b \}.
\]

Theorem (Gerlach, 1981)

A vector \( x \in \mathbb{R}^n \) is a solution of \( Ax \leq b \) if and only if

\[
A^c x \leq A^\Delta |x| + \overline{b}.
\]

Corollary

An \( x \in \mathbb{R}^n \) is a solution of \( Ax \leq b, \ x \geq 0 \) if and only if \( \underline{A}x \leq \underline{b}, \ x \geq 0 \).
Proof of Gerlach’s Theorem

Theorem (Gerlach, 1981)

A vector $x \in \mathbb{R}^n$ is a solution of $Ax \leq b$ if and only if

$$A^c x \leq A^\Delta |x| + \overline{b}. \tag{1}$$

Proof.

If $x$ is a solution of $Ax \leq b$, then $Ax \leq b$ for some $A \in A$ and $b \in b$, and one has

$$A^c x \leq A^c x + b - Ax = (A^c - A)x + b \leq |(A^c - A)| |x| + b \leq A^\Delta |x| + \overline{b}.$$  

Conversely, let $x$ satisfy (1). Set $z := \text{sgn}(x)$, so $|x| = \text{diag}(z)x$. Thus (1) takes the form of

$$A^c x \leq A^\Delta \text{diag}(z)x + \overline{b},$$

or

$$(A^c - A^\Delta \text{diag}(z))x \leq \overline{b}.$$ 

Hence $x$ fulfills $Ax \leq b$ for $b := \overline{b}$ and $A := A^c - A^\Delta \text{diag}(z)$. \qed
Example of the Solution Set

Example (An interval polyhedron)

\[
\begin{bmatrix}
-2, 5 & 7, 11 \\
1, 13 & 4, 6 \\
5, 8 & -2, 1 \\
-1, 4 & 5, 9 \\
-5, 6 & 0, 4
\end{bmatrix} \quad x \leq \begin{bmatrix}
61, 63 \\
19, 20 \\
15, 22 \\
24, 25 \\
26, 37
\end{bmatrix}
\]

- union of all feasible sets in light gray,
- intersection of all feasible sets in dark gray,
A vector $x \in \mathbb{R}^n$ is a strong solution to $Ax \leq b$ if $Ax \leq b$ for every $A \in \mathcal{A}$ and $b \in \mathcal{b}$.

**Theorem (Rohn & Kreslová, 1994)**

A vector $x \in \mathbb{R}^n$ is a strong solution iff there are $x^1, x^2 \in \mathbb{R}^n$ such that

$$x = x^1 - x^2, \quad \overline{A}x^1 - Ax^2 \leq b, \quad x^1 \geq 0, \quad x^2 \geq 0.$$  \hspace{1cm} (2)

**Theorem (Machost, 1970)**

A vector $x \in \mathbb{R}^n$ is a strong solution $Ax \leq b, \ x \geq 0$ iff it solves

$$\overline{A}x \leq b, \ x \geq 0.$$

**Proof.**

One direction is trivial. Conversely, if $x^*$ solves $\overline{A}x \leq b, \ x \geq 0$, then for each $A \in \mathcal{A}$ and $b \in \mathcal{b}$,

$$Ax^* \leq \overline{A}x^* \leq b \leq b.$$
Theorem (Rohn & Kreslová, 1994)

An interval system $Ax \leq b \ (x \geq 0)$ has a strong solution iff $Ax \leq b$ is feasible for each $A \in A$ and $b \in b$.

Proof.

One direction obvious, the latter not obvious.

Remark

The statement is surprising. Analogy for interval equations does not hold, for example

$$x + y = [1, 2], \quad x - y = [2, 3]$$

is feasible for each realization, but there is no common solution.
What are topological properties of the solution set to $Ax \leq b$?

1. Can $\Sigma$ be disconnected?
2. Can $\Sigma$ have both bounded and unbounded connectivity components?
3. Can $\Sigma$ have several bounded connectivity components?
<table>
<thead>
<tr>
<th>solution type</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>solution of $Ax = b$</td>
<td>$</td>
</tr>
<tr>
<td>strong solution of $Ax = b$</td>
<td>$A^c x - b^c = A^\Delta</td>
</tr>
<tr>
<td>tolerance solution of $Ax = b$</td>
<td>$</td>
</tr>
<tr>
<td>solution of $Ax \leq b$</td>
<td>$A^c x - b^c \leq A^\Delta</td>
</tr>
<tr>
<td>strong solution of $Ax \leq b$</td>
<td>$A^c x - b^c \leq -A^\Delta</td>
</tr>
</tbody>
</table>
*Linear optimization problems with inexact data.*  

W. Gerlach.  
Zur Lösung linearer Ungleichungssysteme bei Störung der rechten Seite und der Koeffizientenmatrix.  

M. Hladík.  
Weak and strong solvability of interval linear systems of equations and inequalities.  

B. Machost.  
Numerische Behandlung des Simplexverfahrens mit intervallanalytischen Methoden.  

J. Rohn and J. Kreslová.  
Linear interval inequalities.  
1 Software Presentation

2 Interval Linear Inequalities – Solution Set

3 Algorithmic Issues
Polynomial-time cases

- **Nonnegative solvability.** By Gerlach: the system $Ax \leq b, x \geq 0$ is solvable $\iff$ the system $Ax \leq \bar{b}, x \geq 0$ is solvable (LP).

- **Strong nonnegative solvability (and existence of a strong nonnegative solution).** Check $\bar{A}x \leq b, x \geq 0$ (LP).

- **Strong solvability (and existence of a strong solution).** The system $Ax \leq b$ is strongly solvable $\iff$ it has a strong solution $x^0 \iff (\exists x^1, x^2 \geq 0)$ s.t. $x^0 = x^1 - x^2$ and $\bar{A}x^1 - Ax^2 \leq b$ (LP).

**Theorem**

*Checking solvability of $Ax \leq b$ is NP-complete.*
Proof.

Rohn’s system $|Ax| \leq e$, $\|x\|_1 \geq 1$ can be rewritten as

$$
\begin{pmatrix}
A \\
-A \\
0^T
\end{pmatrix} x - \begin{pmatrix} 0 \\ 0 \\ e^T \end{pmatrix} |x| \leq \begin{pmatrix} e \\ e \\ -1 \end{pmatrix}
$$

and this is Gerlach’s inequality for

$$
\begin{pmatrix}
[A, A] \\
[-A, -A] \\
[-e^T, e^T]
\end{pmatrix} x \leq \begin{pmatrix} e \\ e \\ -1 \end{pmatrix}.
$$

Remark. Observe that there is no “dependency problem” even if $A$ occurs in both the first and the second inequality.

Remark. Observe that the problem is NP-complete even if $b$ is crisp and $A$ has intervals in one row only.
### AlgoIss: Comparison

<table>
<thead>
<tr>
<th>Condition</th>
<th>$A\mathbf{x} = \mathbf{b}$</th>
<th>$A\mathbf{x} \leq \mathbf{b}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Solvability $\mathbf{x} \in \mathbb{R}^n$</td>
<td>NP-complete</td>
<td>NP-complete</td>
</tr>
<tr>
<td>Solvability $\mathbf{x} \geq 0$</td>
<td>poly-time</td>
<td>poly-time</td>
</tr>
<tr>
<td>Strong Solvability $\mathbf{x} \in \mathbb{R}^n$</td>
<td>coNP-complete</td>
<td>coNP-complete</td>
</tr>
<tr>
<td>Strong Solvability $\mathbf{x} \geq 0$</td>
<td>coNP-complete</td>
<td>coNP-complete</td>
</tr>
</tbody>
</table>

To recall: **strong solvability** means

$$(\forall A \in A)(\forall b \in b)(\exists x \in \mathbb{R}^n)\; A\mathbf{x} = \mathbf{b} \; (A\mathbf{x} \leq \mathbf{b}).$$
Interval linear programming

Interval Programming 5

Milan Hladík ¹  Michal Černý ²

¹ Faculty of Mathematics and Physics,
Charles University in Prague, Czech Republic
http://kam.mff.cuni.cz/~hladik/

² Faculty of Computer Science and Statistics,
University of Economics, Prague, Czech Republic
http://nb.vse.cz/~cernym/

Workshop on Interval Programming
7th International Conference of Iranian Operation Research Society
Semnan, Iran, May 12–13, 2014
Outline

1. Introduction to Interval linear programming
2. Optimal Value Range
3. Optimal Solution Set
4. Basis Stability
5. Applications
6. Algorithmic Issues
1 Introduction to Interval linear programming
2 Optimal Value Range
3 Optimal Solution Set
4 Basis Stability
5 Applications
6 Algorithmic Issues
Introduction

Linear programming – three basic forms

\[ f(A, b, c) \equiv \min c^T x \quad \text{subject to} \quad Ax = b, \; x \geq 0, \]

\[ f(A, b, c) \equiv \min c^T x \quad \text{subject to} \quad Ax \leq b, \]

\[ f(A, b, c) \equiv \min c^T x \quad \text{subject to} \quad Ax \leq b, \; x \geq 0. \]

Interval linear programming

Family of linear programs with \( A \in A, \; b \in b, \; c \in c, \) in short

\[ f(A, b, c) \equiv \min c^T x \quad \text{subject to} \quad Ax \overset{(\leq)}{=} b, \; (x \geq 0). \]

The three forms are not transformable between each other!

Goals

- determine the optimal value range;
- determine a tight enclosure to the optimal solution set.
## Complexity of Basic Problems

<table>
<thead>
<tr>
<th></th>
<th>$Ax = b, \ x \geq 0$</th>
<th>$Ax \leq b$</th>
<th>$Ax \leq b, \ x \geq 0$</th>
</tr>
</thead>
<tbody>
<tr>
<td>strong feasibility</td>
<td>co-NP-hard</td>
<td>polynomial</td>
<td>polynomial</td>
</tr>
<tr>
<td>weak feasibility</td>
<td>polynomial</td>
<td>NP-hard</td>
<td>polynomial</td>
</tr>
<tr>
<td>strong unboundedness</td>
<td>co-NP-hard</td>
<td>polynomial</td>
<td>polynomial</td>
</tr>
<tr>
<td>weak unboundedness</td>
<td>suff. / necessary conditions only</td>
<td>suff. / necessary conditions only</td>
<td>polynomial</td>
</tr>
<tr>
<td>strong optimality</td>
<td>co-NP-hard</td>
<td>co-NP-hard</td>
<td>polynomial</td>
</tr>
<tr>
<td>weak optimality</td>
<td>suff. / necessary conditions only</td>
<td>suff. / necessary conditions only</td>
<td>polynomial</td>
</tr>
<tr>
<td>optimal value range</td>
<td>$f$ polynomial, $\overline{f}$ NP-hard</td>
<td>$f$ NP-hard, $\overline{f}$ polynomial</td>
<td>polynomial</td>
</tr>
</tbody>
</table>
1. Introduction to Interval linear programming
2. Optimal Value Range
3. Optimal Solution Set
4. Basis Stability
5. Applications
6. Algorithmic Issues
### Optimal Value Range

#### Definition

\[ \underline{f} := \min f(A, b, c) \text{ subject to } A \in A, \ b \in b, \ c \in c, \]
\[ \overline{f} := \max f(A, b, c) \text{ subject to } A \in A, \ b \in b, \ c \in c. \]

#### Theorem (Vajda, 1961)

We have for type \((Ax \leq b, \ x \geq 0)\)

\[ \underline{f} = \min c^T x \text{ subject to } Ax \leq b, \ x \geq 0, \]
\[ \overline{f} = \min \overline{c}^T x \text{ subject to } \overline{A}x \leq b, \ x \geq 0. \]

#### Theorem (Rohn, 2006)

We have for type \((Ax = b, \ x \geq 0)\)

\[ \underline{f} = \min c^T x \text{ subject to } Ax \leq b, \ \overline{A}x \geq b, \ x \geq 0, \]
\[ \overline{f} = \max_{p \in \{\pm 1\}^m} f(A^c - \text{diag}(p)A^\Delta, b^c + \text{diag}(p)b^\Delta, \overline{c}). \]
Algorithm (Optimal value range \([\underline{f}, \overline{f}]\))

1. Compute
   \[
   \underline{f} := \inf (c^c)^T x - (c^\Delta)^T |x| \quad \text{subject to} \quad x \in \mathcal{M},
   \]
   where \(\mathcal{M}\) is the primal solution set.

2. If \(\underline{f} = \infty\), then set \(\overline{f} := \infty\) and stop.

3. Compute
   \[
   \overline{\varphi} := \sup (b^c)^T y + (b^\Delta)^T |y| \quad \text{subject to} \quad y \in \mathcal{N},
   \]
   where \(\mathcal{N}\) is the dual solution set.

4. If \(\overline{\varphi} = \infty\), then set \(\overline{f} := \infty\) and stop.

5. If the primal problem is strongly feasible, then set \(\overline{f} := \overline{\varphi}\); otherwise set \(\overline{f} := \infty\).
1. Introduction to Interval linear programming
2. Optimal Value Range
3. Optimal Solution Set
4. Basis Stability
5. Applications
6. Algorithmic Issues
The optimal solution set

Denote by $S(A, b, c)$ the set of optimal solutions to

$$\min c^T x \text{ subject to } Ax = b, \; x \geq 0,$$

Then the optimal solution set is defined

$$S := \bigcup_{A \in \mathbf{A}, \; b \in \mathbf{b}, \; c \in \mathbf{c}} S(A, b, c).$$

Goal

Find a tight enclosure to $S$.

Characterization

By duality theory, we have that $x \in S$ if and only if there is some $y \in \mathbb{R}^m$, $A \in \mathbf{A}$, $b \in \mathbf{b}$, and $c \in \mathbf{c}$ such that

$$Ax = b, \; x \geq 0, \; A^T y \leq c, \; c^T x = b^T y,$$

where $A \in \mathbf{A}$, $b \in \mathbf{b}$, $c \in \mathbf{c}$.
1. Introduction to Interval linear programming
2. Optimal Value Range
3. Optimal Solution Set
4. Basis Stability
5. Applications
6. Algorithmic Issues
Basis Stability

**Definition**

The interval linear programming problem

\[
\min c^T x \quad \text{subject to} \quad Ax = b, \ x \geq 0,
\]

is $B$-stable if $B$ is an optimal basis for each realization.

**Theorem**

*B-stability implies that the optimal value bounds are*

\[
f = \min c_B^T x \quad \text{subject to} \quad A_B x_B \leq \bar{b}, \ -A_B x_B \leq -b, \ x_B \geq 0,
\]

\[
\bar{f} = \max \bar{c}_B^T x \quad \text{subject to} \quad A_B x_B \leq \bar{b}, \ -A_B x_B \leq -b, \ x_B \geq 0.
\]

*Under the unique $B$-stability, the set of all optimal solutions reads*

\[
A_B x_B \leq \bar{b}, \ -A_B x_B \leq -b, \ x_B \geq 0, \ x_N = 0.
\]

*(Otherwise each realization has at least one optimal solution in this set.)*
Basis Stability

Non-interval case

Basis $B$ is optimal iff

C1. $A_B$ is non-singular;

C2. $A_B^{-1}b \geq 0$;

C3. $c_N^T - c_B^T A_B^{-1} A_N \geq 0^T$.

Interval case

The problem is $B$-stable iff C1–C3 holds for each $A \in \mathbf{A}, \, b \in \mathbf{b}, \, c \in \mathbf{c}$.

Condition C1

- C1 says that $A_B$ is regular;
- co-NP-hard problem;
- Beeck’s sufficient condition: $\rho \left( \left| (A^c)_B \right|^{-1} (A^\Delta)_B \right) < 1$. 
Basis Stability

Non-interval case

Basis $B$ is optimal iff

C1. $A_B$ is non-singular;
C2. $A_B^{-1}b \geq 0$;
C3. $c_N^T - c_B^T A_B^{-1} A_N \geq 0^T$.

Interval case

The problem is $B$-stable iff C1–C3 holds for each $A \in A$, $b \in b$, $c \in c$.

Condition C2

- C2 says that the solution set to $A_B x_B = b$ lies in $\mathbb{R}_+^n$;
- sufficient condition: check of some enclosure to $A_B x_B = b$. 
Basis Stability

Non-interval case

Basis $B$ is optimal iff

C1. $A_B$ is non-singular;
C2. $A_B^{-1}b \geq 0$;
C3. $c_N^T - c_B^T A_B^{-1} A_N \geq 0^T$.

Interval case

The problem is B-stable iff C1–C3 holds for each $A \in \mathbf{A}$, $b \in \mathbf{b}$, $c \in \mathbf{c}$.

Condition C3

- C2 says that $A_N^T y \leq c_N$, $A_B^T y = c_B$ is strongly feasible;
- co-NP-hard problem;
- sufficient condition: $(A_N^T)y \leq c_N$, where $y$ is an enclosure to $A_B^T y = c_B$. 
**Theorem**

Condition C3 holds true if and only if for each \( q \in \{\pm 1\}^m \) the polyhedral set described by

\[
\begin{align*}
((A^c)\!^T_B - (A^\Delta)\!^T_B \text{ diag}(q))y &\leq \bar{c}_B, \\
-( (A^c)\!^T_B + (A^\Delta)\!^T_B \text{ diag}(q))y &\leq -c_B, \\
\text{diag}(q)y &\geq 0
\end{align*}
\]

lies inside the polyhedral set

\[
((A^c)\!^T_N + (A^\Delta)\!^T_N \text{ diag}(q))y \leq c_N, \quad \text{diag}(q)y \geq 0.
\]
Example

Consider an interval linear program

\[
\begin{align*}
\text{max } & \quad ([5, 6], [1, 2])^T x \\
\text{s.t. } & \quad \begin{pmatrix} -[2, 3] & [7, 8] \\ [6, 7] & -[4, 5] \end{pmatrix} x \leq \begin{pmatrix} [15, 16] \\ [18, 19] \\ [6, 7] \end{pmatrix}, \ x \geq 0.
\end{align*}
\]

- union of all feasible sets in light gray,
- intersection of all feasible sets in dark gray,
- set of optimal solutions in dotted area
### Interval case

Basis $B$ is optimal iff

- **C1.** $A_B$ is non-singular;
- **C2.** $A_B^{-1} b \geq 0$ for each $b \in \mathbf{b}$.
- **C3.** $c_N^T - c_B^T A_B^{-1} A_N \geq 0^T$.

### Condition C1

- C1 and C3 are trivial
- C2 is simplified to

\[
A_B^{-1} \mathbf{b} \geq 0,
\]

which is easily verified by interval arithmetic

- overall complexity: polynomial
Basis Stability – Interval Objective Function

**Interval case**

Basis $B$ is optimal iff

- **C1.** $A_B$ is non-singular;
- **C2.** $A_B^{-1}b \geq 0$;
- **C3.** $c_N^T - c_B^T A_B^{-1} A_N \geq 0^T$ for each $c \in \mathbf{c}$

**Condition C1**

- C1 and C2 are trivial
- C3 is simplified to

\[ A_N^T y \leq c_N, \quad A_B^T y = c_B \]

or,

\[ (A_N^T A_B^{-T}) c_B \leq c_N. \]

- overall complexity: polynomial
Introduction to Interval linear programming

Optimal Value Range

Optimal Solution Set

Basis Stability

Applications

Algorithmic Issues
Applications

Real-Life Applications
- Transportation problems with uncertain demands, suppliers, and/or costs.
- Networks flows with uncertain capacities.
- Diet problems with uncertain amounts of nutrients in foods.
- Portfolio selection with uncertain rewards.
- Matrix games with uncertain payoffs.

Technical Applications
- Tool for global optimization.
- Measure of sensitivity of linear programs.
Example (Stigler’s Nutrition Model)


- \( n = 20 \) different types of food,
- \( m = 9 \) nutritional demands,
- \( a_{ij} \) is the amount of nutrient \( j \) contained in one unit of food \( i \),
- \( b_i \) is the required minimal amount of nutrient \( j \),
- \( c_j \) is the price per unit of food \( j \),
- minimize the overall cost

The model reads

\[
\min c^T x \quad \text{subject to} \quad Ax \geq b, \quad x \geq 0.
\]

The entries \( a_{ij} \) are not stable!
## Example (Stigler’s Nutrition Model (cont.))

### Nutritive value of foods (per dollar spent)

<table>
<thead>
<tr>
<th>Food</th>
<th>Calorie (1000)</th>
<th>Protein (g)</th>
<th>Calcium (g)</th>
<th>Iron (mg)</th>
<th>Vitamin-a (1000iu)</th>
<th>Vitamin-b1 (mg)</th>
<th>Vitamin-b2 (mg)</th>
<th>Niacin (mg)</th>
<th>Vitamin-c (mg)</th>
</tr>
</thead>
<tbody>
<tr>
<td>wheat</td>
<td>44.7</td>
<td>1411</td>
<td>2.0</td>
<td>365</td>
<td>55.4</td>
<td>33.3</td>
<td>441</td>
<td></td>
<td></td>
</tr>
<tr>
<td>cornmeal</td>
<td>36</td>
<td>897</td>
<td>1.7</td>
<td>99</td>
<td>30.9</td>
<td>17.4</td>
<td>7.9</td>
<td>106</td>
<td></td>
</tr>
<tr>
<td>cannedmilk</td>
<td>8.4</td>
<td>422</td>
<td>15.1</td>
<td>9</td>
<td>26</td>
<td>3</td>
<td>23.5</td>
<td>11</td>
<td>60</td>
</tr>
<tr>
<td>margarine</td>
<td>20.6</td>
<td>17</td>
<td>.6</td>
<td>6</td>
<td>55.8</td>
<td>.2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cheese</td>
<td>7.4</td>
<td>448</td>
<td>16.4</td>
<td>19</td>
<td>28.1</td>
<td>.8</td>
<td>10.3</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>peanut-b</td>
<td>15.7</td>
<td>661</td>
<td>1</td>
<td>48</td>
<td>9.6</td>
<td>8.1</td>
<td>471</td>
<td></td>
<td></td>
</tr>
<tr>
<td>lard</td>
<td>41.7</td>
<td></td>
<td></td>
<td>.2</td>
<td></td>
<td></td>
<td></td>
<td>.5</td>
<td>5</td>
</tr>
<tr>
<td>liver</td>
<td>2.2</td>
<td>333</td>
<td>.2</td>
<td>139</td>
<td>169.2</td>
<td>6.4</td>
<td>50.8</td>
<td>316</td>
<td>525</td>
</tr>
<tr>
<td>porkroast</td>
<td>4.4</td>
<td>249</td>
<td>.3</td>
<td>37</td>
<td>18.2</td>
<td>3.6</td>
<td>79</td>
<td></td>
<td></td>
</tr>
<tr>
<td>salmon</td>
<td>5.8</td>
<td>705</td>
<td>6.8</td>
<td>45</td>
<td>3.5</td>
<td>1</td>
<td>4.9</td>
<td>209</td>
<td></td>
</tr>
<tr>
<td>greenbeans</td>
<td>2.4</td>
<td>138</td>
<td>3.7</td>
<td>80</td>
<td>4.3</td>
<td>5.8</td>
<td>37</td>
<td>862</td>
<td></td>
</tr>
<tr>
<td>cabbage</td>
<td>2.6</td>
<td>125</td>
<td>4</td>
<td>36</td>
<td>7.2</td>
<td>9</td>
<td>4.5</td>
<td>26</td>
<td>5369</td>
</tr>
<tr>
<td>onions</td>
<td>5.8</td>
<td>166</td>
<td>3.8</td>
<td>59</td>
<td>16.6</td>
<td>4.7</td>
<td>5.9</td>
<td>21</td>
<td>1184</td>
</tr>
<tr>
<td>potatoes</td>
<td>14.3</td>
<td>336</td>
<td>1.8</td>
<td>118</td>
<td>6.7</td>
<td>29.4</td>
<td>7.1</td>
<td>198</td>
<td>2522</td>
</tr>
<tr>
<td>spinach</td>
<td>1.1</td>
<td>106</td>
<td></td>
<td>138</td>
<td>918.4</td>
<td>5.7</td>
<td>13.8</td>
<td>33</td>
<td>2755</td>
</tr>
<tr>
<td>sweet-pot</td>
<td>9.6</td>
<td>138</td>
<td>2.7</td>
<td>54</td>
<td>290.7</td>
<td>8.4</td>
<td>5.4</td>
<td>83</td>
<td>1912</td>
</tr>
<tr>
<td>peaches</td>
<td>8.5</td>
<td>87</td>
<td>1.7</td>
<td>173</td>
<td>86.8</td>
<td>1.2</td>
<td>4.3</td>
<td>55</td>
<td>57</td>
</tr>
<tr>
<td>prunes</td>
<td>12.8</td>
<td>99</td>
<td>2.5</td>
<td>154</td>
<td>85.7</td>
<td>3.9</td>
<td>4.3</td>
<td>65</td>
<td>257</td>
</tr>
<tr>
<td>limabeans</td>
<td>17.4</td>
<td>1055</td>
<td>3.7</td>
<td>459</td>
<td>5.1</td>
<td>26.9</td>
<td>38.2</td>
<td>93</td>
<td></td>
</tr>
<tr>
<td>navybeans</td>
<td>26.9</td>
<td>1691</td>
<td>11.4</td>
<td>792</td>
<td>38.4</td>
<td>24.6</td>
<td>217</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
If the entries $a_{ij}$ are known with 10% accuracy, then

- the problem is not basis stable
- the minimal cost ranges in $[0.09878, 0.12074]$
- the interval enclosure of the solution set is

$$[0, 0.0734], [0, 0.0438], [0, 0.0576], [0, 0.0283], [0, 0.0535], [0, 0.0315], [0, 0.0339],$$
$$[0, 0.0300], [0, 0.0246], [0, 0.0337], [0, 0.0358], [0, 0.0387], [0, 0.0396], [0, 0.0429],$$
$$[0, 0.0370], [0, 0.0443], [0, 0.0290], [0, 0.0330], [0, 0.0472], [0, 0.1057].$$

If the entries $a_{ij}$ are known with 1% accuracy, then

- the problem is basis stable
- the minimal cost ranges in $[0.10758, 0.10976]$
- the interval hull of the solution set is

$$x_1 = [0.0282, 0.0309], x_8 = [0.0007, 0.0031], x_{12} = [0.0110, 0.0114],$$
$$x_{15} = [0.0047, 0.0053], x_{20} = [0.0600, 0.0621].$$
Research Directions

- Special cases of linear programs.
- Generalizations to nonlinear, multiobjective and other programs.
- Considering simple dependencies (H., Č., 2014).
- Other concepts of optimality; similarly to AE-solutions. (W. Li, J. Luo et al., 2013, 2014)
Open Problems

- A sufficient and necessary condition for weak unboundedness, strong boundedness and weak optimality.
- A method for determining the image of the optimal value function.
- A sufficient and necessary condition for duality gap to be zero for each realization.
- A method to test if a basis $B$ is optimal for some realization.
- Tight enclosure to the optimal solution set.


Introduction to Interval linear programming

Optimal Value Range

Optimal Solution Set

Basis Stability

Applications

Algorithmic Issues
AlgoIss: Optimal Value Range for the form $A x = b$, $x \geq 0$

**To recall:** By correctness of the Optimal Value Range algorithm, we have the following form of IntLP-duality:

**Lemma**

If $f = \sup_{(A,b,c) \in (A,b,c)} \inf \{ c^T x : A^T x = b, x \geq 0 \}$ is finite, then

$$
\overline{f} = \varphi := \sup \{(b^c)^T y + (b^\Delta)^T |y| : y \in \mathcal{N}\},
$$

where $\mathcal{N} = \bigcup_{(A,b,c) \in (A,b,c)} \{y : Ay \leq c\} = \{y : A^c y - A^\Delta |y| \leq \overline{c}\}$ is the dual solution set.

An interesting special case with crisp $A^T = (A^T, -A^T)$, $c = (e^T, e^T)$ and interval $b = [-e, e]$:

**Corollary**

If $f = \sup_{b \in [-e, e]} \inf \{ e^T x^1 + e^T x^2 : A^T (x^1 - x^2) = b, x^1 \geq 0, x^2 \geq 0 \}$ is finite, then

$$
\overline{f} = \max \{e^T |y| : -e \leq Ay \leq e\} \quad (= \max \{\|y\|_1 : -e \leq Ay \leq e\}).
$$
We have almost proved:

**Theorem**

Computation of $\overline{f}$ is NP-hard for the form $Ax = b, x \geq 0$.

**Proof.**

- The following form of Rohn’s generic problem is NP-complete: *given a regular matrix $A$, decide whether the system $-e \leq Ax \leq e, \|x\|_1 \geq 1$ is solvable.* [Singular matrices can be easily excluded: if $A$ is singular, then $Ax = 0, \|x\|_1 \geq 1$ has a solution, and so does $-e \leq Ax \leq e, \|x\|_1 \geq 1$.]

- Let a regular matrix $A$ be given and consider the problem of computing $\overline{f}$ for

$$
\min e^T x^1 + e^T x^2 : \quad A^T(x^1 - x^2) = [-e, e], \quad x^1 \geq 0, \quad x^2 \geq 0.
$$

The dual feasible set $\mathcal{N} = \{y : -e \leq Ay \leq e\}$ is nonempty ($0 \in \mathcal{N}$) and bounded (since $A$ is regular); moreover, $|b| \leq e$ is also bounded. Thus the dual problem is feasible and bounded for every $b \in [-e, e]$, and so is the primal problem by LP-duality. Thus $\overline{f}$ is finite and

$$
\overline{f} = \max \{\|y\|_1 : -e \leq Ay \leq e\}.
$$

Now $\overline{f} \geq 1$ iff $-e \leq Ax \leq e, \|x\|_1 \geq 1$ is solvable.
To recall: \( f \) is computable in polynomial time by the LP
\[
\min c^T x \text{ s.t. } Ax \leq b, \overline{Ax} \geq b, x \geq 0.
\]
The NP-hardness result shows that the \( 2^n \)-algorithm based on orthant decomposition
\[
\overline{f} = \max_{s \in \{\pm 1\}} \min \{ \overline{c}^T x : (A^c - T_s A^\Delta)x = b^c + T_s b^\Delta, x \geq 0 \}
\]
with \( T_s = \text{diag}(s) \) is the “best possible”.

Exercise

Prove analogous results for the forms
- \( Ax \leq b \): \( \overline{f} \) poly-time, \( f \) NP-hard;
- \( Ax \leq b, x \geq 0 \): both \( \overline{f}, \overline{f} \) poly-time.

Note that duality plays role here: the forms \( Ax \leq b \) and \( Ax = b, x \geq 0 \) are dual to each other and complexity results are “complementary”. The form \( Ax \leq b, x \geq 0 \) is “self-dual”.
Consider the linear regression model
\[ y = X\beta + \varepsilon, \]
where columns of \( X \) are *regressors* and \( y \) is a *dependent variable*. Often we use minimum norm estimators
- \( \hat{\beta} = \arg\min_{\beta} \| y - X\beta \|_2 = (X^T X)^{-1}X^T y \) (least squares),
- \( \hat{\beta} = \arg\min_{\beta} \| y - X\beta \|_1 \) (least absolute deviations),
- \( \hat{\beta} = \arg\min_{\beta} \| y - X\beta \|_{\infty} \) (Chebyshev approximation).

The \( \| \cdot \|_1 \) and \( \| \cdot \|_{\infty} \) problems can be written as linear programs:
\[
\begin{align*}
\min_{r, \beta} e^T r \quad & \text{s.t.} \quad X\beta - y \leq r, \quad -X\beta + y \leq r, \quad r \geq 0. \\
\min_{t, \beta} t \quad & \text{s.t.} \quad X\beta - y \leq te, \quad -X\beta + y \leq te, \quad t \geq 0.
\end{align*}
\]

We will consider the latter problem with interval data \((X, y)\):
\[
\begin{align*}
\min_{t, \beta} t \quad & \text{s.t.} \quad X\beta - y \leq te, \quad -X\beta + y \leq te, \quad t \geq 0.
\end{align*}
\]
We are given interval data \((X, y)\) and we are to solve

\[
\min_{t, \beta} t \text{ s.t. } X\beta - y \leq te, \quad -X\beta + y \leq te, \quad t \geq 0.
\]  

(1)

**Illustration**

Basis stability = robustness of classification:

- Let Class 1 be defined by \(C_1 = \{i : y_i \geq X_i; \hat{\beta}\}\).
- Let Class 2 be defined by \(C_2 = \{i : y_i \leq X_i; \hat{\beta}\}\).
- Basis stability: the same classification (i.e. \(C_1 = C_2\)) for every \((X, y) \in (X, y)\).

**Theorem**

*Testing basis stability of the interval LP (1) is a coNP-complete problem.*

**Remark.** The IntLP (1) is a fake IntLP since it suffers from dependencies...
We will show that testing regularity of a given interval matrix $A$ is reducible to testing basis stability of

$$\min_{t, \beta} t \text{ s.t. } X\beta - y \leq te, \quad -X\beta + y \leq te, \quad t \geq 0.$$  \hspace{4cm} (2)$$

Let $A$ be given and consider (2) with $(X, y) = (A, [-e, e])$.

**Step 1.** Regularity $\Rightarrow$ Basis stability. Let $X = A$ be regular. For every $X \in X$, $\beta = X^{-1}y$, $t = 0$ is the optimal solution. Thus, all $2n + 1$ inequalities of the system

$$X\beta - y \leq te, \quad -X\beta + y \leq te, \quad t \geq 0$$

hold as equations. Thus the basis $\{1, \ldots, n, 2n + 1\}$ is optimal.

**Step 2.** Singularity $\Rightarrow$ Basis instability. Let $X_0 \in X = A$ be singular. We will show two different choices of $y \in [-e, e]$ leading to two different optimal bases.
Proof (contd.)

To recall: we work with $\min t, \beta \ t \ s.t. \ X\beta - y \leq te, \ -X\beta + y \leq te, \ t \geq 0$.

We want to prove Step 2: Singularity $\Rightarrow$ Basis instability. Let $X_0 \in X = A$ be singular. We will show two different choices of $y \in [-e, e]$ leading to two different optimal bases.

- **Choice 1:** Let $y_0 \in [-e, e]$ s.t. $y_0$ is linearly independent of columns of $X_0$. (By singularity of $X_0$, such a choice is possible.) Any optimal solution of
  \[
  X_0\beta - y_0 \leq te, \ -X_0\beta + y_0 \leq te, \ t \geq 0
  \]
  must have $t > 0$ (since $t = 0$ implies $X_0\beta = y_0$ and $y_0$ is dependent on columns of $X_0$). Thus an optimal basis does not contain the inequality $t \geq 0$ (= index $2n + 1$) since always $t > 0$.

- **Choice 2:** Let $y_0 = 0$. Then $\beta = 0$, $t = 0$ is an optimal solution. Thus every optimum solution has $t = 0$ and we must choose an optimal basis containing $t = 0$ (= index $2n + 1$).
Eigenvalues and positive definiteness of interval matrices
Interval Programming 6

Milan Hladík ¹ Michal Černý ²

¹ Faculty of Mathematics and Physics, Charles University in Prague, Czech Republic
http://kam.mff.cuni.cz/~hladik/

² Faculty of Computer Science and Statistics, University of Economics, Prague, Czech Republic
http://nb.vse.cz/~cernym/

Workshop on Interval Programming
7th International Conference of Iranian Operations Research Society
Semnan, Iran, May 12–13, 2014
Outline

1. Eigenvalues of Symmetric Interval Matrices
2. Positive (Semi-)Definiteness
3. Application: Convexity Testing
1 Eigenvalues of Symmetric Interval Matrices

2 Positive (Semi-)Definiteness

3 Application: Convexity Testing
A Symmetric Interval Matrix

\[ \mathbf{A}^S := \{ A \in \mathbf{A} : A = A^T \}. \]

Without loss of generality assume that \( A = A^T, \bar{A} = \bar{A}^T \), and \( \mathbf{A}^S \neq \emptyset \).

Eigenvalues of a Symmetric Interval Matrix

Eigenvalues of a symmetric \( A \in \mathbb{R}^{n \times n} \): \( \lambda_1(A) \geq \cdots \geq \lambda_n(A) \).

Eigenvalue sets of \( \mathbf{A}^S \):

\[ \lambda_i(\mathbf{A}^S) := \left\{ \lambda_i(A) : A \in \mathbf{A}^S \right\}, \quad i = 1, \ldots, n. \]

Theorem

Checking whether \( 0 \in \lambda_i(\mathbf{A}^S) \) for some \( i = 1, \ldots, n \) is NP-hard.

Proof.

\( A \) is singular iff \( \mathbf{M}^S := \begin{pmatrix} 0 & A \\ A^T & 0 \end{pmatrix}^S \) is singular (has a zero eigenvalue). \( \square \)
Example

Let

\[ A \in \mathbb{A} = \begin{pmatrix} [1, 2] & 0 & 0 \\ 0 & [7, 8] & 0 \\ 0 & 0 & [4, 10] \end{pmatrix} \]

What are the eigenvalue sets?
We have \( \lambda_1(A^S) = [7, 10] \), \( \lambda_2(A^S) = [4, 8] \) and \( \lambda_3(A^S) = [1, 2] \).

Eigenvalue sets are compact intervals. They may intersect or equal.
Theorem (Hertz, 1992)

We have

\[ \lambda_1(A^S) = \max_{z \in \{\pm 1\}^n} \lambda_1(A^c + \text{diag}(z)A^\Delta \text{diag}(z)), \]

\[ \lambda_n(A^S) = \min_{z \in \{\pm 1\}^n} \lambda_n(A^c - \text{diag}(z)A^\Delta \text{diag}(z)). \]

Proof.

"Upper bound." By contradiction suppose that there is \( A \in A^S \) such that

\[ \lambda_1(A) > \max_{z \in \{\pm 1\}^n} \lambda_1(A_z), \]

where \( A_z \equiv A^c + \text{diag}(z)A^\Delta \text{diag}(z) \).

Thus \( Ax = \lambda_1(A)x \) for some \( x \) with \( \|x\|_2 = 1 \).

Put \( z^* := \text{sgn}(x) \), and by the Rayleigh–Ritz Theorem we have

\[ \lambda_1(A) = x^TAx \leq x^TA_{z^*}x \]

\[ \leq \max_{y: \|y\|_2 = 1} y^TA_{z^*}y = \lambda_1(A_{z^*}). \]
Eigenvalues – Some Other Exact Bounds

**Theorem**

Δ₁(AS) and \( \bar{\lambda}_n(AS) \) are polynomially computable by semidefinite programming.

**Proof.**

We have

\[
\bar{\lambda}_n(AS) = \max \alpha \text{ subject to } A - \alpha I_n \text{ is positive semidefinite, } A \in AS.
\]

Consider a block diagonal matrix \( M(A, \alpha) \) with blocks

\[
A - \alpha I_n, \ a_{ij} - a_{ij}, \ \bar{a}_{ij} - a_{ij}, \ i \leq j.
\]

Then the optimization problem reads

\[
\bar{\lambda}_n(AS) = \max \alpha \text{ subject to } M(A, \alpha) \text{ is positive semidefinite.}
\]
Theorem

We have

\[ \lambda_i(A^S) \subseteq [\lambda_i(A^c) - \rho(A^\Delta), \lambda_i(A^c) + \rho(A^\Delta)], \quad i = 1, \ldots, n. \]

Proof.

Recall for any \( A, B \in \mathbb{R}^{n \times n} \),

\[ |A| \leq B \implies \rho(A) \leq \rho(|A|) \leq \rho(B), \]

and for \( A, B \) symmetric (Weyl’s Theorem)

\[ \lambda_i(A) + \lambda_n(B) \leq \lambda_i(A + B) \leq \lambda_i(A) + \lambda_1(B), \quad i = 1, \ldots, n. \]

Let \( A \in A^S \), so \( |A - A^c| \leq A^\Delta \). Then

\[
\lambda_i(A) = \lambda_i(A^c + (A - A^c)) \leq \lambda_i(A^c) + \lambda_1(A - A^c) \\
\leq \lambda_i(A^c) + \rho(|A - A^c|) \leq \lambda_i(A^c) + \rho(A^\Delta).
\]

Similarly for the lower bound.
Eigenvalues – Easy Cases

Theorem

1. If $A^c$ is essentially non-negative, i.e., $A^c_{ij} \geq 0 \ \forall i \neq j$, then
   $$\lambda_1(A^S) = \lambda_1(A).$$

2. If $A^\Delta$ is diagonal, then
   $$\lambda_1(A^S) = \lambda_1(A), \quad \lambda_n(A^S) = \lambda_n(A).$$

Proof.

1. For the sake of simplicity suppose $A^c \geq 0$. Then $\forall A \in A^S$ we have $|A| \leq A$, whence
   $$\lambda_1(A) = \rho(A) \leq \rho(A) = \lambda_1(A).$$

2. By Hertz’s theorem,
   $$\lambda_1(A^S) = \max_{z \in \{\pm 1\}^n} \lambda_1(A^c + \text{diag}(z)A^\Delta \text{diag}(z)),$$
   $$= \lambda_1(A^c + A^\Delta) = \lambda_1(A).$$
1. Eigenvalues of Symmetric Interval Matrices

2. Positive (Semi-)Definiteness

3. Application: Convexity Testing
**Positive Semidefiniteness**

\( A^S \) is *positive semidefinite* if every \( A \in A^S \) is positive semidefinite.

**Theorem**

The following are equivalent

1. \( A^S \) is positive semidefinite,
2. \( A_z = A^c - \text{diag}(z)A^\Delta \text{diag}(z) \) is positive semidefinite \( \forall z \in \{\pm 1\}^n \),
3. \( x^T A^c x - |x|^T A^\Delta |x| \geq 0 \) for each \( x \in \mathbb{R}^n \).

**Proof.**

“(1) \( \Rightarrow \) (2)” Obvious from \( A_z \in A^S \).

“(2) \( \Rightarrow \) (3)” Let \( x \in \mathbb{R}^n \) and put \( z := \text{sgn}(x) \). Now,

\[
x^T A^c x - |x|^T A^\Delta |x| = x^T A^c x - x^T \text{diag}(z)A^\Delta \text{diag}(z)x = x^T A_z x \geq 0.
\]

“(3) \( \Rightarrow \) (1)” Let \( A \in A^S \) and \( x \in \mathbb{R}^n \). Now,

\[
x^T Ax = x^T A^c x + x^T (A - A^c)x \geq x^T A^c x - |x|^T (A - A^c)x|
\]

\[
\geq x^T A^c x - |x|^T A^\Delta |x| \geq 0.
\]
**Positive Definiteness**

\( A^S \) is *positive definite* if every \( A \in A^S \) is positive definite.

**Theorem**

The following are equivalent

1. \( A^S \) is positive definite,
2. \( A_z \equiv A^c - \text{diag}(z)A^\Delta \text{diag}(z) \) is positive definite for each \( z \in \{\pm 1\}^n \),
3. \( x^T A^c x - |x|^T A^\Delta |x| > 0 \) for each \( 0 \neq x \in \mathbb{R}^n \),
4. \( A^c \) is positive definite and \( A \) is regular.

**Proof.**

“(1) \iff (2) \iff (3)” analogously.

“(1) \Rightarrow (4)” If there are \( A \in A \) and \( x \neq 0 \) such that \( Ax = 0 \), then

\[
0 = x^T Ax = x^T \frac{1}{2}(A + A^T)x,
\]

and so \( \frac{1}{2}(A + A^T) \in A^S \) is not positive definite.

“(4) \Rightarrow (1)” Positive definiteness of \( A^c \) implies \( \lambda_i(A^c) > 0 \ \forall i \), and regularity of \( A \) implies \( \lambda_i(A^S) > 0 \ \forall i \).
Complexity

Theorem (Nemirovskii, 1993)
Checking positive semidefiniteness of $A^S$ is co-NP-hard.

Theorem (Rohn, 1994)
Checking positive definiteness of $A^S$ is co-NP-hard.

Theorem (Jaulin and Henrion, 2005)
Checking whether there is a positive semidefinite matrix in $A^S$ is a polynomial time problem.

Proof.
There is a positive semidefinite matrix in $A^S$ iff $\lambda_n(A^S) \geq 0$. So we can check it by semidefinite programming.
Theorem

1. $A^S$ is positive semidefinite if $\lambda_n(A^c) \geq \rho(A^\Delta)$.
2. $A^S$ is positive definite if $\lambda_n(A^c) > \rho(A^\Delta)$.
3. $A^S$ is positive definite if $A^c$ is positive definite and $\rho(||(A^c)^{-1}|A^\Delta||) < 1$.

Proof.

1. $A^S$ is positive semidefinite iff $\lambda_n(A^S) \geq 0$.
   Now, employ the smallest eigenvalue set enclosure
   \[ \lambda_n(A^S) \subseteq [\lambda_n(A^c) - \rho(A^\Delta), \lambda_n(A^c) + \rho(A^\Delta)]. \]

2. Analogous.

3. Use Beeck’s sufficient condition for regularity of $A$. 
1 Eigenvalues of Symmetric Interval Matrices

2 Positive (Semi-)Definiteness

3 Application: Convexity Testing
Application: Convexity Testing

**Theorem**

A function $f : \mathbb{R}^n \mapsto \mathbb{R}$ is convex on $x \in \mathbb{IR}^n$ iff its Hessian $\nabla^2 f(x)$ is positive semidefinite $\forall x \in \text{int } x$.

**Corollary**

A function $f : \mathbb{R}^n \mapsto \mathbb{R}$ is convex on $x \in \mathbb{IR}^n$ if $\nabla^2 f(x)$ is positive semidefinite.
Application: Convexity Testing

Example

Let

\[ f(x, y, z) = x^3 + 2x^2y - xyz + 3yz^2 + 8y^2, \]

where \( x \in x = [2, 3], \ y \in y = [1, 2] \) and \( z \in z = [0, 1] \). The Hessian of \( f \) reads

\[
\nabla^2 f(x, y, z) = \begin{pmatrix}
6x + 4y & 4x - z & -y \\
4x - z & 16 & -x + 6z \\
-y & -x + 6z & 6y
\end{pmatrix}
\]

Evaluation the Hessian matrix by interval arithmetic results in

\[
\nabla^2 f(x, y, z) \subseteq \begin{pmatrix}
[16, 26] & [7, 12] & [-1, 2] \\
[7, 12] & 16 & [-3, 4] \\
-1 & [-3, 4] & [6, 12]
\end{pmatrix}
\]

Now, both sufficient conditions for positive definiteness succeed. Thus, we can conclude that \( f \) is convex on the interval domain.
References

M. Hladík, D. Daney, and E. Tsigaridas.
Bounds on real eigenvalues and singular values of interval matrices.

M. Hladík, D. Daney, and E. P. Tsigaridas.
Characterizing and approximating eigenvalue sets of symmetric interval matrices.

L. Jaulin and D. Henrion.
Contracting optimally an interval matrix without loosing any positive semi-definite matrix is a tractable problem.

J. Rohn.
Positive definiteness and stability of interval matrices.

J. Rohn.
A handbook of results on interval linear problems.
Handling constraints rigorously

Interval Programming 7

Milan Hladík ¹  Michal Černý ²

¹ Faculty of Mathematics and Physics, Charles University in Prague, Czech Republic
http://kam.mff.cuni.cz/~hladik/

² Faculty of Computer Science and Statistics, University of Economics, Prague, Czech Republic
http://nb.vse.cz/~cernym/

Workshop on Interval Programming
7th International Conference of Iranian Operation Research Society
Semnan, Iran, May 12–13, 2014
Outline

1. Nonlinear Equations
2. Interval Newton method (square system)
3. Krawczyk method (square case)
4. More general constraints
1 Nonlinear Equations

2 Interval Newton method (square system)

3 Krawczyk method (square case)

4 More general constraints
Nonlinear Equations

Problem Statement

Find all solutions to

\[ f_j(x_1, \ldots, x_n) = 0, \quad j = 1, \ldots, j^* \]

inside the box \( x^0 \in \mathbb{R}^n \).

Theorem (Zhu, 2005)

For a polynomial \( p(x_1, \ldots, x_n) \), there is no algorithm solving

\[ p(x_1, \ldots, x_n)^2 + \sum_{i=1}^{n} \sin^2(\pi x_i) = 0. \]

Proof.

From Matiyasevich’s theorem solving the 10th Hilbert problem.

Remark

Using the arithmetical operations only, the problem is decidable by Tarski’s theorem (1951).
1. Nonlinear Equations

2. Interval Newton method (square system)

3. Krawczyk method (square case)

4. More general constraints
Interval Newton method

Classical Newton method

... is an iterative method

\[ x^{k+1} := x^k - \nabla f(x^k)^{-1}f(x^k), \quad k = 0, \ldots \]

Cons

- Can miss some solutions
- Not verified (Are we really close to the true solution?)

Interval Newton method – Stupid Intervalization

\[ x^{k+1} := x^k - \nabla f(x^k)^{-1}f(x^k), \quad k = 0, \ldots \]

Interval Newton method – Good Intervalization

\[ N(x^k, x^k) := x^k - \nabla f(x^k)^{-1}f(x^k), \]

\[ x^{k+1} := x^k \cap N(x^k), \quad k = 0, \ldots \]
Theorem (Moore, 1966)

If \( x, x^0 \in \mathbf{x} \) and \( f(x) = 0 \), then \( x \in N(x^0, x) \).

Proof.

By the Mean value theorem,

\[
    f_i(x) - f_i(x^0) = \nabla f_i(c_i)^T (x - x^0), \quad \forall i = 1, \ldots, n.
\]

If \( x \) is a root, we have

\[
    -f_i(x^0) = \nabla f_i(c_i)^T (x - x^0).
\]

Define \( A \in \mathbb{R}^{n \times n} \) such that its \( i \)th row is equal to \( \nabla f_i(c_i)^T \). Hence

\[
    -f(x^0) = A(x - x^0),
\]

from which

\[
    x = x^0 - A^{-1} f(x^0) \in x^0 - \nabla f(x)^{-1} f(x^0).
\]

Notice, that this does not mean that there is \( c \in \mathbf{x} \) such that

\[
    -f(x^0) = \nabla f(c)(x - x^0).
\]
Theorem (Nickel, 1971)

If $\emptyset \neq N(x^0, x) \subseteq x$, then there is a unique root in $x$ and $\nabla f(x)$ is regular.

Proof.

“Regularity.” Easy.

“Existence.” By Brouwer’s fixed-point theorem.

[Any continuous mapping of a compact convex set into itself has a fixed point.]

“Uniqueness.” If there are two roots $y_1 \neq y_2$ in $x$, then by the Mean value theorem,

$$f(y_1) - f(y_2) = A(y_1 - y_2)$$

for some $A \in \nabla f(x)$; since $f(y_1) = f(y_2) = 0$, we get

$$A(y_1 - y_2) = 0$$

and by the nonsingularity of $A$, the roots are identical.
Interval Newton method

Practical Implementation

Instead of
\[ N(x^k, x^k) := x^k - \nabla f(x^k)^{-1}f(x^k) \]
let \( N(x^k, x^k) \) be an enclosure of the solution set (with respect to \( x \)) of
\[ \nabla f(x)(x - x^0) = -f(x^0). \]

Extended Interval Arithmetic

So far
\[ \frac{[12, 15]}{[-2, 3]} = (-\infty, \infty). \]

Now,
\[ a/b := \{a/b : a \in a, 0 \neq b \in b\}. \]
So,
\[ \frac{[12, 15]}{[-2, 3]} = (-\infty, -6] \cup [4, \infty). \]
Interval Newton method

Example

\[ f(x) = x^3 - x + 0.2 \]

In six iterations precision \(10^{-11}\) (quadratic convergence).
Interval Newton method

Example (Moore, 1993)

\[ f(x) = x^2 + \sin(x^{-3}) \]

All 318 roots of in the interval \([0.1, 1]\) found with accuracy \(10^{-10}\). The left most root is contained in \([0.10003280626, 0.10003280628]\).

Summary

- \(N(x^0, x)\) contains all solutions in \(x\)
- If \(x \cap N(x^0, x) = \emptyset\), then there is no root in \(x\)
- If \(\emptyset \neq N(x^0, x) \subseteq x\), then there is a unique root in \(x\)
1 Nonlinear Equations

2 Interval Newton method (square system)

3 Krawczyk method (square case)

4 More general constraints
Krawczyk method

Krawczyk operator

Let \( x^0 \in \mathbf{x} \) and \( C \in \mathbb{R}^{n \times n} \), usually \( C \approx \nabla f(x^0)^{-1} \). Then

\[
K(x) := x^0 - Cf(x^0) + (I_n - C\nabla f(x))(x - x^0).
\]

Theorem

Any root of \( f(x) \) in \( \mathbf{x} \) is included in \( K(x) \).

Proof.

If \( x^1 \) is a root of \( f(x) \), then it is a fixed point of

\[
g(x) := x - Cf(x).
\]

By the mean value theorem,

\[
g(x^1) \in g(x^0) + \nabla g(x)(x^1 - x^0),
\]

whence

\[
x^1 \in g(x) \subseteq g(x^0) + \nabla g(x)(x - x^0) = x^0 - Cf(x^0) + (I_n - C\nabla f(x))(x - x^0).
\]

Theorem

If \( K(x) \subseteq x \), then there is a root in \( x \).

Proof.

Recall

\[
g(x) := x - Cf(x).
\]

By the proof of the previous Theorem, \( K(x) \subseteq x \) implies

\[
g(x) \subseteq x.
\]

Thus, there is a fixed point \( x^0 \in x \) of \( g(x) \),

\[
g(x^0) = x^0 - Cf(x^0) = x^0,
\]

so \( x^0 \) is a root of \( f(x) \).
Theorem (Kahan, 1968)

If $K(x) \subseteq \text{int} \ x$, then there is a unique root in $x$ and $\nabla f(x)$ is regular.

Recall Theorem from Lecture 2

Let $x \in \mathbb{R}^n$ and $C \in \mathbb{R}^{n \times n}$. If

$$K(x) = Cb + (I_n - CA)x \subseteq \text{int} \ x,$$

then $C$ is nonsingular, $A$ is regular, and $\Sigma \subseteq x$.

Proof.

The inclusion $K(x) \subseteq \text{int} \ x$ reads

$$-Cf(x^0) + (I_n - C\nabla f(x))(x - x^0) \subseteq \text{int}(x - x^0)$$

Apply the above Theorem for

$$b := -f(x^0), \quad A := \nabla f(x), \quad x := x - x^0$$

We have that $\nabla f(x)$ is regular, which implies uniqueness.
Exercise
Let \( f(x, c) : \mathbb{R}^n \times \mathbb{R} \mapsto \mathbb{R}^n \) be a function depending on parameter \( c \). Let \( c \in \mathbb{R} \) and \( x \in \mathbb{R}^n \). Give a condition under which there is a simple zero in \( f(x, c) \) in \( x \) for each \( c \in c \).
**Verification**

**Problem formulation**

Given an approximate solution \( x^* \), find \( y \in \mathbb{R}^n \) such that there is a solution in \( x^* + y \).

**\( \varepsilon \)-inflation method (Rump, 1983)**

Put \( y := -Cf(x^0) \).

Repeat inflating \( z := [0.9, 1.1]y + 10^{-20}[-1, 1] \) and updating

\[
    y := -Cf(x^0) + (I_n - C\nabla f(x))z
\]

until \( y \subseteq \text{int } z \).

Then, there is a unique solution in \( x^* + y \).
\[ \pi^2(y - \pi/2) + 4x^2 \sin(x) = 0, \quad x - \pi - \cos(y) = 0. \]

Approximate solution \( x^* = (3.1415, 1.5708)^T \).

Enclosing with accuracy \( 10^{-5} \) fails, but accuracy \( 10^{-4} \) succeeds.
1. Nonlinear Equations

2. Interval Newton method (square system)

3. Krawczyk method (square case)

4. More general constraints
More general constraints

Constraints

- equations $h_i(x) = 0$, $i = 1, \ldots, I$
- inequalities $g_j(x) \leq 0$, $j = 1, \ldots, J$
- may be others, but not considered here
  ($\neq$, quantifications, logical operators, lexicographic orderings, \ldots)

Problem

Denote by $\Sigma$ the set of solutions in an initial box $x^0 \in \mathbb{IR}^n$?
Problem: How to describe $\Sigma$?

Subpavings

Split $x$ into a union of three sets of boxes such that

- the first set has boxes provably containing no solution
- the second set has boxes that provably consist of only solutions
- the third set has boxes which may or may not contain a solution
Example

\[ x^2 + y^2 \leq 16, \]
\[ x^2 + y^2 \geq 9 \]

**Figure:** Exact solution set

**Figure:** Subpaving approximation
Subpaving Algorithm

Branch & Bound Scheme

1: \( \mathcal{L} := \{x^0\} \), [set of boxes to process]
2: \( \mathcal{S} := \emptyset \), [set of boxes with solutions only]
3: \( \mathcal{N} := \emptyset \), [set of boxes with no solutions]
4: \( \mathcal{B} := \emptyset \), [set of the undecidable boxes]
5: while \( \mathcal{L} \neq \emptyset \) do
6: choose \( x \in \mathcal{L} \) and remove \( x \) from \( \mathcal{L} \)
7: if \( x \subseteq \Sigma \) then
8: \( \mathcal{S} := \mathcal{S} \cup x \)
9: else if \( x \cap \Sigma = \emptyset \) then
10: \( \mathcal{N} := \mathcal{N} \cup x \)
11: else if \( x_i^A < \varepsilon \ \forall i \) then
12: \( \mathcal{B} := \mathcal{B} \cup x \)
13: else
14: split \( x \) into sub-boxes and put them into \( \mathcal{L} \)
15: end if
16: end while
Example (thanks to Elif Garajová)

1. $\varepsilon = 1.0$
   - Time: 0.952 s

2. $\varepsilon = 0.5$
   - Time: 2.224 s

3. $\varepsilon = 0.125$
   - Time: 9.966 s
Algorithm More in Detail

Test $x \subseteq \Sigma$
- no equations and $\overline{g}_j(x) \leq 0 \ \forall j$

Test $x \cap \Sigma = \emptyset$
- $0 \notin h_i(x)$ for some $i$
- $\overline{g}_j(x) > 0$ for some $j$

Also very important
- Which box to choose (data structure for $\mathcal{L}$)?
- How to divide the box? (which coordinate, which place, how many sub-boxes)

Improvement
Contraction of $x$ such that no solution is missed (and do not use $\mathcal{B}$).
Contractors

Definition

Contractor A function $C : \mathbb{R}^n \rightarrow \mathbb{R}^n$ is called a contractor if $\forall x \in \mathbb{R}^n$ we have

- $C(x) \subseteq x$
- $C(x) \cap \Sigma = x \cap \Sigma$

Example

\[ C \rightarrow \frac{25}{30} \]
Example

Consider the constraint

\[ x + yz = 7, \quad x \in [0, 3], \quad y \in [3, 5], \quad z \in [2, 4]. \]

- **Express \( x \)**

  \[ x = 7 - yz \in 7 - [3, 5][2, 4] = [-13, 1]. \]

  Thus, the domain for \( x \) is \([0, 3] \cap [-13, 1] = [0, 1].\)

- **Express \( y \)**

  \[ y = (7 - x)/z \in (7 - [0, 1])/[2, 4] = [1.5, 3.5]. \]

  Thus, the domain for \( y \) is \([3, 5] \cap [1.5, 3.5] = [3, 3.5].\)

- **Express \( z \)**

  \[ z = (7 - x)/y \in (7 - [0, 1])/[3, 3.5] = [\frac{12}{7}, \frac{7}{3}]. \]

  Thus, the domain for \( z \) is \([2, 4] \cap [\frac{12}{7}, \frac{7}{3}] = [2, \frac{7}{3}].\)

No further propagation needed as each variable appears just once.
Example

Consider the constraint

\[ e^x - xyz = 10, \quad x \in x = [4, 5], \quad y \in y = [3, 4], \quad z \in z = [2, 3]. \]

Contractions of domains:

<table>
<thead>
<tr>
<th>Iteration</th>
<th>x</th>
<th>y</th>
<th>z</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>[4, 4.2485]</td>
<td>[3.4991, 4]</td>
<td>[2.6243, 3]</td>
</tr>
<tr>
<td>2</td>
<td>[4, 4.1106]</td>
<td>[3.6165, 4]</td>
<td>[2.7124, 3]</td>
</tr>
<tr>
<td>3</td>
<td>[4, 4.0831]</td>
<td>[3.6409, 4]</td>
<td>[2.7306, 3]</td>
</tr>
<tr>
<td>4</td>
<td>[4, 4.0775]</td>
<td>[3.6458, 4]</td>
<td>[2.7344, 3]</td>
</tr>
<tr>
<td>5</td>
<td>[4, 4.0764]</td>
<td>[3.6469, 4]</td>
<td>[2.7351, 3]</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Multiple appearance of \( x \) causes infinite convergence.
**Definition (2B-consistency)**

A set of constraints $c_k(x), \ k = 1, \ldots, K,$ on a box $x^0 \in \mathbb{IR}^n$ is 2B-consistent if for each $k \in \{1, \ldots, K\}$ and each $i \in \{1, \ldots, n\}$ there are some $x, x' \in x^0$ such that $x_i = x^0_i, \ x'_i = \overline{x^0}_i,$ and conditions $c_k(x)$ and $c_k(x')$ are valid.

**Remark**

- Constraint propagation tries to approach 2B-consistency.
- Drawback: 2B-consistency looks at constraints separately.
Free Constraint Solving Software

- **Alias** (by Jean-Pierre Merlet, COPRIN team),
  A C++ library for system solving, with Maple interface,

- **Quimper** (by Gill Chabert and Luc Jaulin),
  written in an interval C++ library IBEX,
  a language for interval modelling and handling constraints,
  http://www.emn.fr/z-info/ibex

- **RealPaver** (by L. Granvilliers and F. Benhamou),
  a C++ package for modeling and solving nonlinear and nonconvex constraint satisfaction problems,
  http://pagesperso.lina.univ-nantes.fr/info/perso/permanents/granvil/realpaver

- **RSolver** (by Stefan Ratschan),
  solver for quantified constraints over the real numbers,
  implemented in the programming language OCaml,
  http://rsolver.sourceforge.net/
G. Alefeld and J. Herzberger.  
*Introduction to Interval Computations.*  

F. Benhamou and L. Granvilliers.  
Continuous and interval constraints.  

G. Chabert and L. Jaulin.  
Contractor programming.  

F. Goualard and C. Jermann.  
A reinforcement learning approach to interval constraint propagation.  

L. Jaulin, M. Kieffer, O. Didrit, and É. Walter.  
*Applied Interval Analysis.*  

S. M. Rump.  
Verification methods: Rigorous results using floating-point arithmetic.  
Global Optimization by Interval Techniques

Interval Programming 8

Milan Hladík 1 Michal Černý 2

1 Faculty of Mathematics and Physics,
Charles University in Prague, Czech Republic
http://kam.mff.cuni.cz/~hladik/

2 Faculty of Computer Science and Statistics,
University of Economics, Prague, Czech Republic
http://nb.vse.cz/~cernym/

Workshop on Interval Programming
7th International Conference of Iranian Operation Research Society
Semnan, Iran, May 12–13, 2014
Global optimization problem

Compute global (not just local!) optima to

$$\min f(x) \text{ subject to } g(x) \leq 0, \ h(x) = 0, \ x \in x^0,$$

where $x^0 \in \mathbb{R}^n$ is an initial box.

Theorem (Zhu, 2005)

There is no algorithm solving global optimization problems using operations $+, \times, \sin$.

Proof.

From Matiyasevich’s theorem solving the 10th Hilbert problem.

Remark

Using the arithmetical operations only, the problem is decidable by Tarski’s theorem (1951).
Interval Approach to Global Optimization

**Branch & Bound Scheme**

1: \[ \mathcal{L} := \{x^0\}, \] [set of boxes]
2: \[ c^* := \infty, \] [upper bound on the minimal value]
3: while \( \mathcal{L} \neq \emptyset \) do
4: choose \( x \in \mathcal{L} \) and remove \( x \) from \( \mathcal{L} \),
5: contract \( x \),
6: find a feasible point \( x \in x \) and update \( c^* \),
7: if \( \max_i x_i^\Delta > \varepsilon \) then
8: split \( x \) into sub-boxes and put them into \( \mathcal{L} \),
9: else
10: give \( x \) to the output boxes,
11: end if
12: end while

It is a rigorous method to enclose all global minima in a set of boxes.
Which box to choose?

- the oldest one
- the one with the largest edge, i.e., for which \( \max_i x_i^A \) is maximal
- the one with minimal \( f(x) \).
Division Directions

How to divide the box?

1. Take the widest edge of \( \mathbf{x} \), that is

\[
k := \arg \max_{i=1,...,n} x_i^\Delta.
\]

2. (Walster, 1992) Choose a coordinate in which \( f \) varies possibly mostly

\[
k := \arg \max_{i=1,...,n} f'_{x_i}(\mathbf{x})^\Delta x_i^\Delta.
\]

3. (Ratz, 1992) It is similar to the previous one, but uses

\[
k := \arg \max_{i=1,...,n} (f'_{x_i}(\mathbf{x})x_i)^\Delta.
\]

Remarks

- by Ratschek & Rokne (2009) there is no best strategy for splitting
- combine several of them
- the splitting strategy influences the overall performance
Contracting and Pruning

Aim

Shrink $\mathbf{x}$ to a smaller box (or completely remove) such that no global minimum is removed.

Simple Techniques

- if $0 \not\in h_i(x)$ for some $i$, then remove $\mathbf{x}$
- if $0 < g_j(x)$ for some $j$, then remove $\mathbf{x}$
- if $0 < f'_x(x)$ for some $i$, then fix $x_i := \underline{x}_i$
- if $0 > f'_x(x)$ for some $i$, then fix $x_i := \overline{x}_i$

Optimality Conditions

- employ the Fritz–John (or the Karush–Kuhn–Tucker) conditions

$$u_0 \nabla f(x) + u^T \nabla h(x) + v^T \nabla g(x) = 0, \quad v \geq 0,$$

$$h(x) = 0, \quad g(x) \leq 0, \quad v_\ell g_\ell(x) = 0 \quad \forall \ell, \quad \|(u_0, u, v)\| = 1.$$
Inside the Feasible Region

Suppose there are no equality constraints and $g_j(x) < 0 \forall j$.

- (monotonicity test) if $0 \not\in f'_x(x)$ for some $i$, then remove $x$

- apply the Interval Newton method to the additional constraint $\nabla f(x) = 0$

- (nonconvexity test) if the interval Hessian $\nabla^2 f(x)$ contains no positive semidefinite matrix, then remove $x$
### Feasibility Test

<table>
<thead>
<tr>
<th><strong>Aim</strong></th>
<th>Find a feasible point $x^<em>$, and update $c^</em> := \min(c^<em>, f(x^</em>))$.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Why?</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Remove boxes with $f(x) &gt; c^*$.</td>
</tr>
<tr>
<td></td>
<td>• We can include $f(x) \leq c^*$ to the constraints.</td>
</tr>
<tr>
<td><strong>No equations</strong></td>
<td>If no equality constraints, take, e.g., $x^* := x^c$ provided $g(x^c) \leq 0$.</td>
</tr>
</tbody>
</table>
For equations

- if $k$ equality constraints, fix $n - k$ variables $x_i := x_i^c$ and solve system of equations by the interval Newton method
- if $k = 1$, fix the variables corresponding to the smallest absolute values in $\nabla h(x^c)$

- If $k > 1$, transform the matrix $\nabla h(x^c)$ to REF by using a complete pivoting, and fix components corresponding to the right most columns
Lower Bounds

Aim

Given a box \( x \in \mathbb{R}^n \), determine a lower bound to \( f(x) \).

Why?

- if \( f(x) > c^* \), we can remove \( x \)
- minimum over all boxes gives a lower bound on the optimal value

Methods

- interval arithmetic
- mean value form
- slope form
- Lipschitz constant approach
- \( \alpha \)BB algorithm
- ...
Special cases: Bilinear terms

For every $y \in \mathbb{R}$ and $z \in \mathbb{R}$ we have

$$yz \geq \max\{yz + zy - yz, \ yz + zy - yz\}.$$ 

General case: Convex underestimators for $f(x)$

Construct a function $g : \mathbb{R}^n \mapsto \mathbb{R}$ satisfying:

- $f(x) \geq g(x)$ for every $x \in x$,
- $g(x)$ is convex on $x \in x$.

αBB algorithm (Androulakis, Maranas & Floudas, 1995)

Consider an underestimator $g(x) \leq f(x)$ in the form

$$g(x) := f(x) - \sum_{i=1}^{n} \alpha_i (\bar{x}_i - x_i)(x_i - \underline{x}_i), \quad \text{where } \alpha_i \geq 0 \ \forall i.$$
Function $f(x)$ and its convex underestimator $g(x)$. 
Computation of $\alpha$

**Idea**

The Hessian of $g(x)$ reads

$$\nabla^2 g(x) = \nabla^2 f(x) + 2 \text{diag}(\alpha).$$

Choose $\alpha$ large enough to ensure positive semidefiniteness of the Hessian of

$$g(x) := f(x) - \sum_{i=1}^{n} \alpha_i (\overline{x}_i - x_i)(x_i - \underline{x}_i).$$

**Interval Hessian matrix**

Let $H$ be an interval matrix enclosing the image of $\nabla^2 f(x)$ over $x \in x$:

$$\frac{\partial^2}{\partial x_i \partial x_j} f(x) \in h_{ij} = \left[ h_{ij}, \overline{h}_{ij} \right], \quad \forall x \in x.$$

**Remarks**

- Checking positive semidefiniteness of each $H \in H$ is co-NP-hard.
- Various enclosures for eigenvalues of $H \in H$.
- Scaled Gerschgorin method enables to express $\alpha_i$-s.
Computation of $\alpha$

**Scaled Gerschgorin method for $\alpha$**

$$\alpha_i := \max \left\{ 0, -\frac{1}{2} \left( h_{ii} - \sum_{j \neq i} |h_{ij}| d_j / d_i \right) \right\}, \quad i = 1, \ldots, n,$$

where $|h_{ij}| = \max \{ |h_{ij}|, |\bar{h}_{ij}| \}$.

To reflect the range of the variable domains, use $d := \overline{x} - \underline{x}$.

**Theorem (H., 2014)**

The choice $d := \overline{x} - \underline{x}$ is optimal (i.e., it minimizes the maximum separation distance between $f(x)$ and $g(x)$) if

$$h_{ii} d_i - \sum_{j \neq i} |h_{ij}| d_j \leq 0, \quad \forall i = 1, \ldots, n.$$
Linearization

Interval linear programming approach

- linearize constraints,
- compute new bounds and iterate.

Example

\[ x' \subseteq x \]

\[ S \]
Mean value form

**Theorem**

Let \( f : \mathbb{R}^n \rightarrow \mathbb{R} \), \( x \in \mathbb{R}^n \) and \( a \in x \). Then

\[
f(x) \subseteq f(a) + \nabla f(x)^T (x - a),
\]

**Proof.**

By the mean value theorem, for any \( x \in x \) there is \( c \in x \) such that

\[
f(x) = f(a) + \nabla f(c)^T (x - a) \in f(a) + \nabla f(x)^T (x - a).
\]

**Improvements**

- **successive mean value form**

\[
f(x) \subseteq f(a) + f'_{x_1}(x_1, a_2, \ldots, a_n)(x_1 - a_1)
+ f'_{x_2}(x_1, x_2, a_3 \ldots, a_n)(x_2 - a_2) + \ldots
+ f'_{x_n}(x_1, \ldots, x_{n-1}, x_n)(x_n - a_n).
\]

- **replace derivatives by slopes**
Slopes

Slope form enclosure

\[ f(x) \subseteq f(a) + S(x, a)(x - a), \]

where \( a \in x \) and

\[ S(x, a) := \begin{cases} \frac{f(x) - f(a)}{x - a} & \text{if } x \neq a, \\ f'(x) & \text{otherwise}. \end{cases} \]

Remarks

- Slopes can be replaced by derivatives, but slopes are tighter.
- Slopes can be computed in a similar way as derivatives.

<table>
<thead>
<tr>
<th>function</th>
<th>its slope ( S(x, a) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( x )</td>
<td>1</td>
</tr>
<tr>
<td>( f(x) \pm g(x) )</td>
<td>( S_f(x, a) \pm S_g(x, a) )</td>
</tr>
<tr>
<td>( f(x) \cdot g(x) )</td>
<td>( S_f(x, a)g(a) + f(x)S_g(x, a) )</td>
</tr>
<tr>
<td>( e^{f(x)} )</td>
<td>( e^{f(x)}S_f(x, a) )</td>
</tr>
</tbody>
</table>
Example

\[ f(x) = \frac{1}{4}x^2 - x + \frac{1}{2}, \quad x = [1, 7]. \]

\[ f'(x) = [-\frac{1}{2}, \frac{5}{2}], \quad S_f(x, x^c) = [\frac{1}{4}, \frac{7}{4}]. \]
Interval linearization

Let $x^0 \in \mathbf{x}$. Suppose that for some interval matrices $A$ and $B$ we have

\[
    h(x) \subseteq A(x - x^0) + h(x^0), \quad \forall x \in \mathbf{x}
\]
\[
    g(x) \subseteq B(x - x^0) + g(x^0), \quad \forall x \in \mathbf{x},
\]

e.g., by the mean value form, slopes, …

Interval linear programming formulation

Now, the set $S$ is enclosed by

\[
    A(x - x^0) + h(x^0) = 0,
\]
\[
    B(x - x^0) + g(x^0) \leq 0.
\]

What remains to do

- Solve the interval linear program
- Choose $x^0 \in \mathbf{x}$
Let $x^0 := x$. Since $x - x^0$ is non-negative, the solution set to
\[
\begin{align*}
A(x - x^0) + h(x^0) &= 0, \\
B(x - x^0) + g(x^0) &\leq 0,
\end{align*}
\]
is described by
\[
\begin{align*}
Ax &\leq Ax - h(x), \\
\overline{Ax} &\geq \overline{Ax} - h(x), \\
Bx &\leq Bx - g(x).
\end{align*}
\]

Similarly if $x^0$ is any other vertex of $x$.
General case

Let \( x^0 \in x \). The solution set to

\[
A(x - x^0) + h(x^0) = 0, \\
B(x - x^0) + g(x^0) \leq 0,
\]

is described by

\[
|A^c(x - x^0) + h(x^0)| \leq A^\Delta |x - x^0|, \\
B^c(x - x^0) + g(x^0) \leq B^\Delta |x - x^0|.
\]

- Non-linear description due to the absolute values.
- How to get rid of them?
- Estimate from above by a linear function: \( |x - x^0| \leq \alpha(x - x^0) + \beta \).
  (Easy to find the best upper linear estimation.)
Example

Typical situation when choosing \( x^0 \) to be vertex:
Typical situation when choosing $x^0$ to be the opposite vertex:
Linearization

Example

Typical situation when choosing $x^0 = x^c$:
Example

Typical situation when choosing $x^0 = x^c$ (after linearization):
Example

Typical situation when choosing all of them:
Examples

Example (The COPRIN examples, 2007, precision $\sim 10^{-6}$)

- **tf12** (origin: COCONUT, solutions: 1, computation time: 60 s)
  
  \[
  \begin{align*}
  \min & \quad x_1 + \frac{1}{2}x_2 + \frac{1}{3}x_3 \\
  \text{s.t.} & \quad -x_1 - \frac{i}{m}x_2 - \left( \frac{i}{m} \right)^2 x_3 + \tan\left( \frac{i}{m} \right) \leq 0, \quad i = 1, \ldots, m \quad (m = 101).
  \end{align*}
  \]

- **o32** (origin: COCONUT, solutions: 1, computation time: 2.04 s)
  
  \[
  \begin{align*}
  \min & \quad 37.293239x_1 + 0.8356891x_5x_1 + 5.3578547x_3^2 - 40792.141 \\
  \text{s.t.} & \quad -0.0022053x_3x_5 + 0.0056858x_2x_5 + 0.0006262x_1x_4 - 6.665593 \leq 0, \\
  & \quad -0.0022053x_3x_5 - 0.0056858x_2x_5 - 0.0006262x_1x_4 - 85.334407 \leq 0, \\
  & \quad 0.0071317x_2x_5 + 0.0021813x_3^2 + 0.0029955x_1x_2 - 29.48751 \leq 0, \\
  & \quad -0.0071317x_2x_5 - 0.0021813x_3^2 - 0.0029955x_1x_2 + 9.48751 \leq 0, \\
  & \quad 0.0047026x_3x_5 + 0.0019085x_3x_4 + 0.0012547x_1x_3 - 15.699039 \leq 0, \\
  & \quad -0.0047026x_3x_5 - 0.0019085x_3x_4 - 0.0012547x_1x_3 + 10.699039 \leq 0.
  \end{align*}
  \]

- **Rastrigin** (origin: Myatt (2004), solutions: 1 (approx.), time: 2.07 s)
  
  \[
  \begin{align*}
  \min & \quad 10n + \sum_{j=1}^{n}(x_j - 1)^2 - 10 \cos(2\pi(x_j - 1)),
  \end{align*}
  \]
  
  where $n = 10$, $x_j \in [-5.12, 5.12]$. 

Examples

One of the Rastrigin functions.

rastriginsfcn([x/10,y/10])

One of the Rastrigin functions.
Rigorous global optimization software

- **GlobSol** (by R. Baker Kearfott), written in Fortran 95, open-source exist conversions from AMPL and GAMS representations, http://interval.louisiana.edu/
- **COCONUT Environment**, open-source C++ classes http://www.mat.univie.ac.at/~coconut/coconut-environment/
- **GLOBAL** (by Tibor Csendes), for Matlab / Intlab, free for academic purposes, http://www.inf.u-szeged.hu/~csendes/linkek_en.html
- **PROFIL / BIAS** (by O. Knüppel et al.), free C++ class http://www.ti3.tu-harburg.de/Software/PROFILEnglisch.html

See also

- **C.A. Floudas** (http://titan.princeton.edu/tools/)
- **A. Neumaier** (http://www.mat.univie.ac.at/~neum/glopt.html)
References

C. A. Floudas.
*Deterministic Global Optimization. Theory, Methods and Applications.*

C. A. Floudas and P. M. Pardalos, editors.
*Encyclopedia of Optimization. 2nd ed.*

E. R. Hansen and G. W. Walster.
*Global Optimization Using Interval Analysis.*

R. B. Kearfott.
*Rigorous Global Search: Continuous Problems.*

A. Neumaier.
Complete Search in Continuous Global Optimization and Constraint Satisfaction.
Minimization of $f(x_1, \ldots, x_n)$ — usual testing functions:

- Rastrigin’s function $20 + x_1^2 + x_2^2 - 10(\cos 2\pi x_1 + \cos 2\pi x_2)$
- Banana (Rosenbrock) function $(1 - x_1)^2 + 100(x_2 - x_1^2)^2$
- hidden minimum function
How to represent the function $f$?

- Special optimization problems: analytical expression (e.g. $\frac{1}{2}x^T Hx + c^T x + \alpha$ for quadratic programming)
- A general function: oracle model — there is a blackbox $f$ s.t. on a query $x$ it returns $f(x)$
- In the oracle model: complexity often measured by the number of oracle queries
- Problem: How to measure the size of input?
**Theorem**

There is no finite upper bound on the number of steps for an optimization algorithm to locate the global minimum.

**Proof.**

For a finite number of testing points, $\varepsilon$ can be chosen so small that the algorithm cannot distinguish between the zero function and the hidden-minimum function.
Approximate solution

- The minimizer may be **irrational** — a problem with representation by a Turing machine.

- **Example:** $\min\{x^T A x : x^T x \leq 1\} = \lambda_{\text{min}}(A)$ for $A$ negative definite. The number $\lambda_{\text{min}}(A)$ is often irrational even for a rational matrix $A$; the minimizer is its corresponding eigenvector.

- Possible solution: **real-number computation model.** [Drawback: we lose finite-time convergence of many weakly polynomial methods, such as the Ellipsoid Method or IPMs.]

- Another solution: find approximate optima.
Definition

A point $x$ is an $\varepsilon$-approximate minimum is $f(x) - f(x^*) \leq \varepsilon$, where $x^*$ is the true minimizer.

Remarks.

- Weak definition — though $f(x)$ is close to $f(x^*)$, the distance $\|x - x^*\|_{\infty}$ can still be extremely large. So sometimes one also adds the requirement: “... and $\|x - x^*\|_{\infty} \leq \varepsilon$”.
- This problem vanishes for Lipschitz functions.

Theorem

If $f$ is $L$-Lipschitz, then the $\varepsilon$-approximate global minimum of $f$ over a unit cube can be found in $\approx \left(\frac{1}{2} \frac{L}{\varepsilon}\right)^n$ steps and not faster.
If $f$ is $L$-Lipschitz, then the $\varepsilon$-approximate global minimum of $f$ over the unit cube $[0, e]$ can be found in $\approx \left(\frac{1}{2} \frac{L}{\varepsilon}\right)^n$ steps and not faster.

Proof idea.

- **Upper bound.** Cover the cube $[0, e]$ by a regular $n$-dimensional grid with distance $\frac{2\varepsilon}{L}$ between neighbor points and evaluate $f$ in each grid point; then take the minimum. Then $\|x - x^*\|_\infty \leq \frac{\varepsilon}{L}$ for some grid point $x$ and $f(x) - f(x^*) \leq L\|x - x^*\| \leq \varepsilon$.

- **Lower bound.** Let $\varepsilon' > \varepsilon$. Define a hidden-minimum function $f_v$

  $$f_v(x) = \begin{cases} 
  0 & \text{if} \|v - x\|_\infty \geq \frac{\varepsilon'}{L}, \\
  L\|v - x\|_\infty - \varepsilon' & \text{if} \|v - x\|_\infty < \frac{\varepsilon'}{L}.
  \end{cases}$$

Then, $v$ is the minimizer. Idea: any algorithm that uses less than $\left(\frac{1}{2} \frac{L}{\varepsilon}\right)^n$ oracle queries to $f$ leaves some region of $[0, e]$ “uninspected”; so we can place $v$ into that region. Thus the algorithm cannot find it and it cannot distinguish between $f_v$ and the zero function.
In general, global minimization is **nonrecursive** (to recall: an algorithm for the question “$\min f(x_1, \ldots, x_n) \leq 0$” would solve Hilbert’s Tenth Problem. This holds true even for the case $n = 1$.

So we must inspect the general problem by subcases.

### Polynomials

- Recursive by Tarski’s quantifier elimination, but extremely slow.
- Idea: the question “does a given polynomial $p(x_1, \ldots, x_n)$ attain a value $f_0$?” can be written as an arithmetical formula

\[
(\exists x_1) \cdots (\exists x_n) p(x_1, \ldots, x_n) = f_0 \\
\quad \& \ x_1 \leq x_1 \leq x_1 \ & \cdots \ & x_n \leq x_n \leq x_n. \tag{1}
\]

- Tarski proved that Theory of Real Closed Fields is decidable. So in principle, we can enumerate all proofs until we find a proof of (1) or its negation. This proves recursivity.
Convex optimization

- “Nice” case: local minimum = global minimum
- However, in general nothing can be proved without additional assumptions
- $\varepsilon$-approximate minimization of a differentiable convex $L$-Lipschitz function can be done in time $O(n^2(\log n + \log \frac{L}{\varepsilon}))$ in the oracle model

Further problems we must face: Example

- Optimization under quadratic constraints $x^T H x + c^T x \leq \gamma$ — is the feasibility problem in NP?
- Problem: a feasible point cannot be used as an NP-witness, since it can happen that bit-size of (a unique) feasible point is exponential in bitsize of $H, c, \gamma$. 
Quadratic programming $\min x^T H x + c^T x \ \text{s.t.} \ Ax \leq b$

- The convex case ($H$ psd): polynomial time
- A single eigenvalue of $H$ is negative: NP-hard
- $H$ general: optimization over an ellipsoid: polynomial time
- $H$ general: optimization over a simplex: NP-hard
- $H$ general: “is given $x$ a local minimum of $x^T H x + c^T x$?”: NP-hard
- Further results: the form $\min x^T H x + c^T x \ \text{s.t.} \ x \in \mathbf{x}$ with $H$ nsd of a fixed rand: polynomial time