Current developments in the nonlinear solver SONIC

Solver and **O**ptimizer for **N**onlinear Problems
based on **I**nterval **C**omputation

Elke Just
University of Wuppertal

June 5, 2013

Contents

Basic facts

- Main contributors:
  Thomas Beelitz, Bruno Lang, Elke Just, Paul Willems
  (Applied Computer Science, University of Wuppertal)
  Peer Ueberholz (Hochschule Niederrhein, University of Applied Sciences)

- initial goal: *efficient* and *robust* design of dynamic systems

Basic facts

- rigorous general purpose solver for nonlinear systems
- also contains optimizer
- written in `C++`
- “generic” interval code provides performance and portability
  - supported interval libraries: `C-XSC`, `SUN C++`, `filib++`
- parallelization available (*OpenMP*, *MPI*)
- based on a branch-and-bound algorithm

Subdivision strategies

- **subdivision of unbounded boxes**
- subdivision using “gaps” (caused by division)
- several basic strategies to choose subdivision direction
- **hybrid subdivision strategy**
  (apply subdivision scheme that seems most promising for current box)
- **shift of subdivision point**
- iterated subdivision

Contraction methods

- **Constraint propagation** (CP)
  on finite unions of real intervals
- **Taylor refinement**
  - first and second order
  - separately or integrated into CP
- **Interval Newton method**
  on a hierarchy of extended systems
- **Several preconditioners**
  - inverse midpoint preconditioner
  - optimal linear programming preconditioners

Verification

- for square systems
  - Newton
  - Miranda
  - Borsuk
  - Tests based on topological degree
- for non-square systems
  (with adapted information)
- **computation and verification can be done separately**
  (different methods can be tested without calculating anew)

Subdivide an unbounded box

Strategy:

- subdivide in all components with $\mathbf{x}_i = [-\infty, \infty]$

subsequently

- subdivide all half-bounded components
  - determine subdivision direction k by

$$v(\mathbf{x}_k) = \max_{i \in \{1,\dots,n\}} v(\mathbf{x}_j) \text{ with } v(\mathbf{y}) := \begin{cases} \overline{y} & \text{for } \underline{y} = -\infty \\ -\underline{y} & \text{for } \overline{y} = \infty \end{cases}$$

  - subdivision point has to be chosen with care too

Algorithm 1 SUBDIVISION OF AN UNBOUNDED BOX $(\mathbf{x}, \delta_1, \delta_2)$

if $\exists\, i \in \{1, \ldots, n\}$ with $\mathbf{x}_i = [-\infty, \infty]$ **then**
  subdivision direction $k := \min\{i\}$ with $\mathbf{x}_i = [-\infty, \infty]$
  subdivision point $p := \delta_1$
else {only half-bounded components}
  determine subdivision direction k maximizing $v(\mathbf{x}_k)$
  if $\inf(\mathbf{x}_k) = -\infty$ **then**
    subdivision point $p := \min\{-\delta_2, 2 \cdot \sup(\mathbf{x}_k)\}$
  else
    subdivision point $p := \max\{\delta_2, 2 \cdot \inf(\mathbf{x}_k)\}$
  end if
end if

Subdivide an unbounded box - Parameters

Additionally, a new **stopping criterion** is introduced:
$\exists i$ with $\text{mig}(\mathbf{x}) = \langle \mathbf{x}_i \rangle \geq \Psi_i$ for threshold vector Ψ

Many problems have a root in zero

$\Rightarrow$ we chose $\delta_1 = 0.1$

The minimum width for the subboxes resulting from the subdivision of unbounded boxes can be shown to be $\delta_2/2$

$\Rightarrow$ we chose $\delta_2 = 1$

Basic idea

- reducing the cluster effect
- without increasing the overall box number
- ansatz (for bounded boxes only):
  shift midpoint p according to parameter $\eta \in [0, 1]$

$$p = \underline{x}_k + \eta \cdot \mathrm{width}\,(\mathbf{x}_k) \tag{1}$$

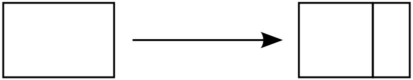


Figure: Bisection with shifted point

Numerical results

Test set with 14 problems

η	0.9	0.8	0.7	0.6	0.55	0.51	0.5
$\text{boxes}_{\text{ratio}}$	6.62	2.25	1.44	2.57	1.08	**0.87**	1.00
$\text{time}_{\text{ratio}}$	5.20	1.94	1.24	2.24	1.08	**0.87**	1.00

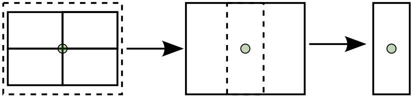
η	0.1	0.2	0.3	0.4	0.45	0.49
$\text{boxes}_{\text{ratio}}$	8.29	2.24	1.31	5.64	1.17	96.32
$\text{time}_{\text{ratio}}$	8.05	2.27	1.29	5.59	1.15	93.52

1 *outlier*, all other problems behave as assumed

Reducing the cluster effect - Other approaches

- Subdivision into 3 subboxes (at least once per direction)
  - Disadvantage: higher box number

    (worst case: 3^r instead of 2^r boxes for r subdivisions of each box)
- A further (not implemented) approach:



Verfification

We only display the handling for **square** systems.

The box informations **ContainsSolutionForSubsystem** and **UniqueSolutionForSubsystem** for underdetermined systems are handled correspondingly.

Collecting boxes

Main goals:

- lose no information

under this constraint

- minimize number of boxes
- save smaller boxes

Algorithm 2 COLLECT BOXES (list L_s of solution boxes)

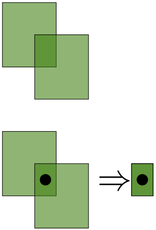
for all pairs of different solution boxes $\mathbf{x}_{\mathrm{sup}}$ and $\mathbf{x}_{\mathrm{sub}}$ with $\mathbf{x}_{\mathrm{sup}} \supseteq \mathbf{x}_{\mathrm{sub}}$) **do**
    if info($\mathbf{x}_{\mathrm{sup}}$) $=$ *NoInformation* **then**
        if info($\mathbf{x}_{\mathrm{sub}}$) $=$ *NoInformation* **then**
            delete $\mathbf{x}_{\mathrm{sub}}$
        else
            set info($\mathbf{x}_{\mathrm{sup}}$) $:=$ *ContainsSolution*
        end if
    end if
    if info($\mathbf{x}_{\mathrm{sup}}$) $=$ *ContainsSolution* **then**
        if info($\mathbf{x}_{\mathrm{sub}}$) $=$ *NoInformation* **then**
            delete $\mathbf{x}_{\mathrm{sub}}$
        end if
    end if

Algorithm 2 COLLECT BOXES - CONTINUED

    if info($\mathbf{x}_{\mathrm{sup}}$) $=$ *UniqueSolution* **then**
        if info($\mathbf{x}_{\mathrm{sub}}$) $=$ *NoInformation* **then**
            delete $\mathbf{x}_{\mathrm{sub}}$
        else
            if info($\mathbf{x}_{\mathrm{sub}}$) $=$ *ContainsSolution* **then**
                set info($\mathbf{x}_{\mathrm{sub}}$) $=$ *UniqueSolution*
                delete $\mathbf{x}_{\mathrm{sup}}$
            else
                if info($\mathbf{x}_{\mathrm{sub}}$) $=$ *UniqueSolution* **then**
                    delete $\mathbf{x}_{\mathrm{sup}}$
                end if
            end if
        end if
    end if
end for

Collecting unique boxes

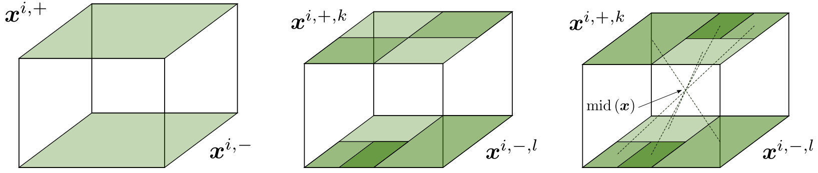
- option for **reducing the number of solution boxes**
  - boxes containing unique tested for intersections (may occur due to epsilon-inflation)
  - intersections tested for uniqueness
  - but: test for intersections is expensive
    $\Rightarrow$ not used in default settings



Symmetrical subfacets - Background

For the verification tests we consider *facets* and subfacets.

For the Borsuk test we need *symmetrical subfacets*.



(a) Facets in component i

(b) Asymmetric subfacets

(c) Symmetric subfacets

Figure: 2-dimensional facets and subfacets for a 3-dimensional box **x**

General problem

In exact arithmetic, subdivide pairs of facets/subfacet in

- the same direction
- in their midpoint

BUT: not sufficient in numerical calculations!

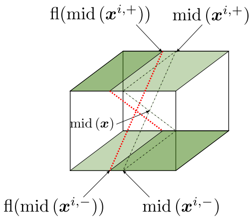
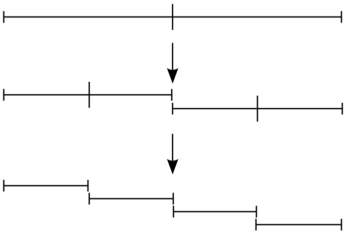


Figure: Asymmetric subdivision of a facet

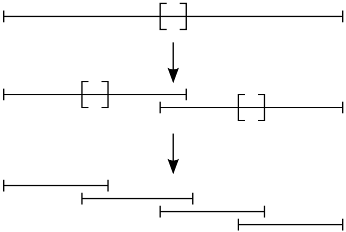
Option 1: enclosure of the midpoints

- enclose the midpoints in which we subdivide

Disadvantage: *overestimation* of the subfacets grows when we subdivide more than once per direction.



(a) optimal

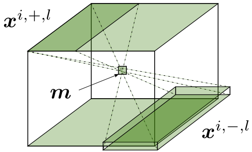


(b) with enclosure of midpoint

Option 2: computing the second subfacet

- subdivide one facet/subfacet in the computed midpoint (even if inexact)
- compute an enclosure **m** of the midpoint of the box
- calculate (an enclosure of) the symmetric counterpart out of these two values by

$$\mathbf{x}^{i,-,l} = 2\mathbf{m} - \mathbf{x}^{i,+,l}$$



Advantage: we need to **store only one** subfacet out of each pair.

Work in progress

- web interface
  (reduced in functionality)
- translators for problems in AMPL, GAMS formulation
- graphical output of solution boxes

Plans and ideas

- improve first test implementation of Taylor models
  - by LDB
  - inverse models
- general acceleration
  - focus: optimal preconditioner
- further research, which structures may "predict" contraction success in interval Newton method

Contact

For further questions or suggestions write to

just@math.uni-wuppertal.de