ROBUST METHOD FOR FINDING OF DISPERSION CURVES IN A THICK PLATE PROBLEM

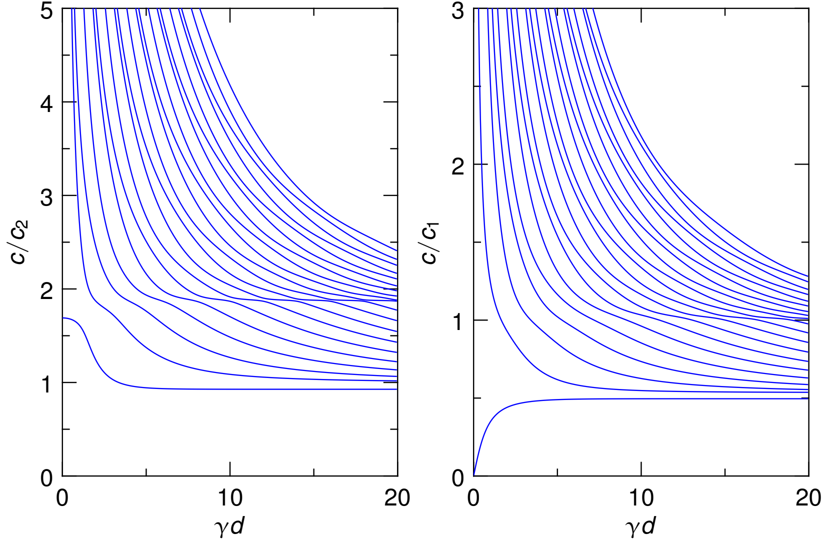
David Pelikán, Petr Hora

Institute of Thermomechanics AS CR, v. v. i.
Prague, Czech Republic

COMPUTATIONAL MECHANICS 2014

November 3 – 5, 2014
Špičák, Czech Republic

Motivation



Interval Analysis

Computing with intervals began:
Bounding rounding and truncation errors in finite precision arithmetic
Ramon Moore, 1966, Interval Analysis

One of Methods of Representing Uncertainty.

Uncertain parameters are described by an upper and lower bound,
then rigorous bounds on the response are computed using
interval arithmetic
and
interval functions.

Moore, R. E.: Interval analysis.
Prentice-Hall series in automatic computation.
Englewood Cliffs, N.J.: Prentice-Hall, 1966.

Interval Arithmetic

Upper and Lower bound

$$[a, b] = x | a \leqslant x \leqslant b$$

Midpoint and Radius

$$[x_0, r] = x | x_0 - r \leqslant x \leqslant x_0 + r$$

Degenerate Interval - Scalar

$$x = [a, a]$$

Width

$$w([a, b]) = b - a$$

Magnitude

$$|[a, b]| = \max(|a|, |b|)$$

Midpoint

$$\mathrm{mid}([a, b]) = (a + b)/2$$

Equality

$$[a, b] = [c, d] \Longleftrightarrow a = b, c = d$$

Ordering

$$[a, b] < [c, d] \Longleftrightarrow b < c$$

Interval Arithmetic

Addition

$$[a, b] + [c, d] = [a + c, b + d]$$

Subtraction

$$[a, b] - [c, d] = [a - d, b - c]$$

Multiplication

$$[a, b] \times [c, d] = [\min(ac, ad, bc, bd), \max(ac, ad, bc, bd)]$$

Division

$$[a, b] \div [c, d] = [a, b] \times \left[\frac{1}{d}, \frac{1}{c}\right]$$

provided $[c, d]$ does not contain 0

Theorem (Ratz, 1996)

Let $\langle a, b\rangle$ and $\langle c, d\rangle$ be two non-empty bounded real intervals.
Then

$$\langle a, b\rangle \div \langle c, d\rangle = \begin{cases} \langle a, b\rangle \times \langle 1/d, 1/c\rangle & \text{if } 0 \notin \langle c, d\rangle \\ \langle -\infty, \infty\rangle & \text{if } 0 \in \langle a, b\rangle \wedge 0 \in \langle c, d\rangle \\ \langle b/c, \infty\rangle & \text{if } b < 0 \wedge c < d = 0 \\ \langle -\infty, b/d\rangle \cup \langle b/c, \infty\rangle & \text{if } b < 0 \wedge c < 0 < d \\ \langle -\infty, b/d\rangle & \text{if } b < 0 \wedge 0 = c < d \\ \langle -\infty, a/c\rangle & \text{if } 0 < a \wedge c < d = 0 \\ \langle -\infty, a/c\rangle \cup \langle a/d, \infty\rangle & \text{if } 0 < a \wedge c < 0 < d \\ \langle a/d, \infty\rangle & \text{if } 0 < a \wedge 0 = c < d \\ \emptyset & \text{if } 0 \notin \langle a, b\rangle \wedge c = d = 0 \end{cases}$$

Ratz, D.: On extended interval arithmetic and inclusion isotonicity. Institut für Angewandte Mathematik, Universität Karlsruhe, 1996.

Anomalies in Interval Arithmetic

Subdistributivity

$$[a, b] \times ([c, d] \pm [e, f]) \subseteq [a, b] \times [c, d] \pm [a, b] \times [e, f]$$

Subcancelation

$$[a, b] - [c, d] \subseteq ([a, b] + [e, f]) - ([c, d] + [e, f])$$

$$[a, b] \div [c, d] \subseteq ([a, b] \times [e, f]) \div ([c, d] \times [e, f])$$

$$0 \in [a, b] - [a, b]$$

$$1 \in [a, b] \div [a, b]$$

Images of Functions

Monotone functions: If $f(x) : x \to \mathbb{R}$ is non-decreasing, then

$$f([a,b]) = [f(a), f(b)].$$

Examples:

$$exp([a,b]) = [exp(a), exp(b)]$$

$$\log([a,b]) = [log(a), log(b)]$$

Some basic functions: Images $x^2, sin(x), \ldots$, are easily calculated, too.

Example:

$$[a,b]^2 = \begin{cases} \left[\min(a^2,b^2), \max(a^2,b^2)\right], & \text{if } 0 \notin [a,b], \\ \left[0, \max(a^2,b^2)\right], & \text{otherwise.} \end{cases}$$

More complex functions: Bessel, ...

Goal of Interval Computation

To compute the SHARPEST possible interval solution set
which COMPLETELY CONTAINS the true solution set.

Rigor is easy to accomplish - $[-\infty, \infty]$ always true but useless

Tight solutions are difficult - primarily due to dependence

The Dependence Problem

Interval arithmetic implicitly assumes
each occurrence of a variable is independent.

Example: Compute X^2 without the power rules where $X = [-1, 2]$.

$$X^2 = X \times X = [\min(-2, 1, 4), \max(-2, 1, 4)] = [-2, 4]$$

The answer should be $X^2 = [0, 4]$

$\Downarrow$

The excess width is know as the dependence problem.

If an interval variable appears only once in an expression
no widening of the interval occurs.

Solving the Dependence Problem

1. Analytically rewrite expressions to minimize the number of times a variable occurs.
2. Delay interval computations as late as possible
3. Reduce dependencies computationally (e.g. with automatic differentiation)

Automatic Differentiation (AD)

- is a set of techniques based on the mechanical application of the chain rule to obtain derivatives of a function given as a computer program.

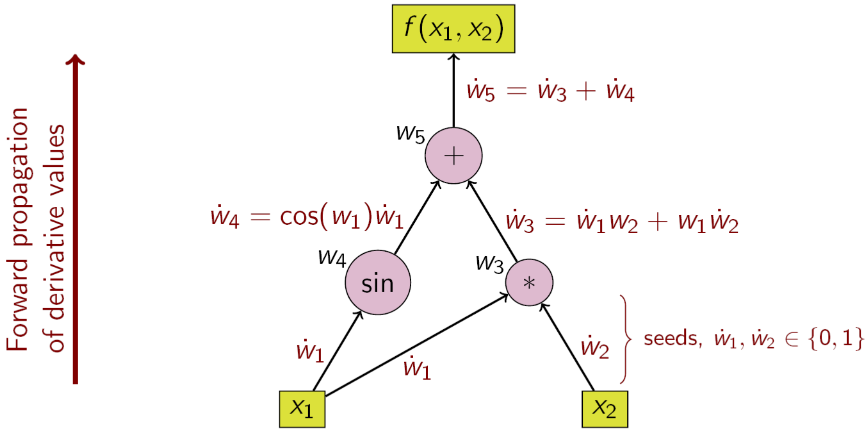
AD exploits the fact that every computer program, no matter how complicated, executes a sequence of elementary arithmetic operations such as additions or elementary functions such as exp(). By applying the chain rule of derivative calculus repeatedly to these operations, derivatives of arbitrary order can be computed automatically, and accurate to working precision.

Conceptually, AD is different from symbolic differentiation and approximations by divided differences.

AD is used in the following areas:

- Numerical Methods
- Sensitivity Analysis
- Design Optimization
- Data Assimilation & Inverse Problems

$$f(x_1, x_2) = x_1 x_2 + \sin(x_1)$$



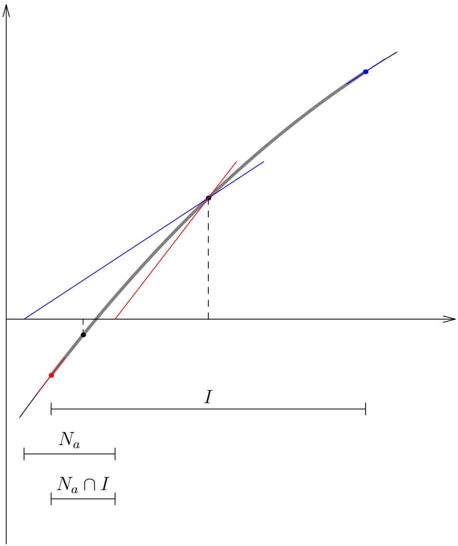
By Berland at en.wikipedia [Public domain], from Wikimedia Commons

```
function [hodnota] = DispRov3(GammaD, Zeta, Poisson)

hodnota=NaN;

while 1

    k=(1-2*Poisson)/(2*(1-Poisson));

    GammaD=scal2matr(GammaD, Zeta);
    Zeta=scal2matr(Zeta,GammaD);

    if (size(GammaD)==size(Zeta))
        hodnota=hodnota.*ones(size(GammaD));
    else
        disp('GammaD a Zeta musí mít stejnou velikost')
        break
    end

    if (all(all(GammaD>=0)) && all(all(Zeta>=0)))
        Zeta(~(Zeta>=(1/sqrt(k))))=NaN; %Hodnoty přesahující vymezený obor hodnot jsou ignorovány.

        hodnota=(Zeta.^2-2).^2.*sin(GammaD.*sqrt(Zeta.^2-1)).*cos(GammaD.*sqrt(k.*Zeta.^2-1))
        +4.*sqrt(Zeta.^2-1).*sqrt(k.*Zeta.^2-1).*cos(GammaD.*sqrt(Zeta.^2-1)).*sin(GammaD.*sqrt(k.*Zeta.^2-1));
    else
        disp('GammaD a Zeta musí být matice kladných reálných čísel (intervalů)')
        break
    end

    break
end
end
```

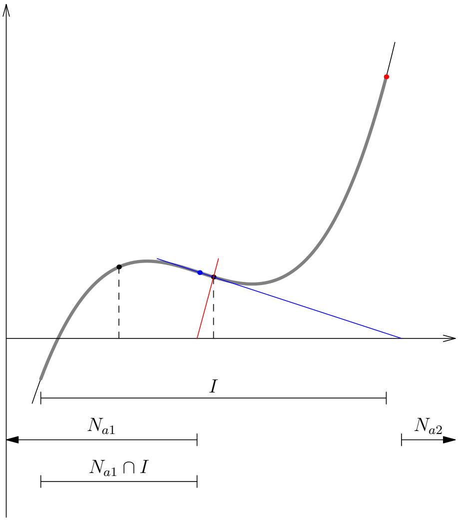
Programming with Intervals

- Matlab
  - INTLAB - pure Matlab
    - interval arithmetic with real and complex data including scalars, vectors, matrices and sparse matrices
    - automatic differentiation
    - rigorous interval standard functions
    - rigorous input and output
    - multiple precision interval arithmetic
  - b4m - uses C interval library, BIAS
- C/C++ and Fortran 77/90 - BIAS, PROFIL, C-XSC, INTLIB, GlobSol
- Python - interval, mpmath(iv)
- Maple - intpak, intpakX
- Mathematica

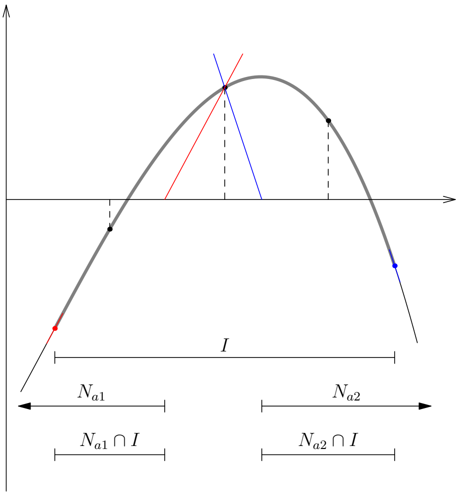
Interval Newton Method



Interval Newton Method



Interval Newton Method



Arbitrary precision computations

Evaluate $f(x, y) = (333.75 - x^2)y^6 + x^2(11x^2y^2 - 121y^4 - 2) + 5.5y^8 + x/(2y)$ at $(77617, 33096)$.

single precision	$f \approx 1.172603\ldots$
double precision	$f \approx 1.1726039400531\ldots$
extended precision	$f \approx 1.172603940053178\ldots$
the true value	$f = -0.827386\ldots$

Taschini, S.: Interval Arithmetic: Python Implementation and Applications. In the Proceedings of the 7th Python in Science Conference (SciPy 2008).

Rump, S.M.: Verification methods: Rigorous results using floating-point arithmetic. Acta Numerica 19:287-449, 2010.

Multiprecision interval arithmetic

```
from sympy.mpmath import iv
print 'using 35 decimal places ...'
iv.dps = 35
iv_f = lambda x,y: (iv.mpf('333.75') \
- x**2)*y**6 \
+ x**2*(iv.mpf('11')*x**2*y**2 \
- iv.mpf('121')*y**4 - iv.mpf('2')) \
+ iv.mpf('5.5')*y**8 + x/(iv.mpf('2')*y);
iv_a = iv.mpf(str(a))
iv_b = iv.mpf(str(b))
iv_z = iv_f(iv_a,iv_b)
print iv_z
```

shows

```
using 35 decimal places ...
[-6.827396059946821368141165095479816292382, \
1.172603940053178631858834904520183709123]
```

```
print 'using 36 decimal places ...'
iv.dps = 36
...
```

shows

```
using 36 decimal places ...
[-0.827396059946821368141165095479816292200549, \
-0.827396059946821368141165095479816291817411]
```

$$= -(0.827396059946821368141165095479816292^{200549}_{181741})$$

width of interval = upper bound on error

Conclusion

Interval arithmetic provides a robust method
for finding the roots of the dispersion equation.

For relatively low speed, this method is not suitable
to complete the calculation, it is however possible
to use the first approach with low accuracy,
or when using the Gaussian integration method.

Thank you for your attention!

Any questions?

The work was supported by the institutional support RVO: 61388998.

Contents