Validated Numerics
a short introduction to rigorous computations

Warwick Tucker

The CAPA group
Department of Mathematics
Uppsala University, Sweden

UPPSALA
UNIVERSITET

Are floating point computations reliable?

Computing with the `C/C++` `single` format

Are floating point computations reliable?

Computing with the `C/C++` `single` format

Example 1: Repeated addition

$$\sum_{i=1}^{10^3} \langle 10^{-3} \rangle = 0.999990701675415,$$

$$\sum_{i=1}^{10^4} \langle 10^{-4} \rangle = 1.000053524971008.$$

UPPSALA
UNIVERSITET

Are floating point computations reliable?

Computing with the `C/C++` `single` format

Example 1: Repeated addition

$$\sum_{i=1}^{10^3} \langle 10^{-3} \rangle = 0.999990701675415,$$

$$\sum_{i=1}^{10^4} \langle 10^{-4} \rangle = 1.000053524971008.$$

Example 2: Order of summation

$$1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{10^6} = 14.357357,$$

$$\frac{1}{10^6} + \cdots + \frac{1}{3} + \frac{1}{2} + 1 = 14.392651.$$

Are floating point computations reliable?

Given the point $(x, y) = (77617, 33096)$, evaluate the function

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

Are floating point computations reliable?

Given the point $(x, y) = (77617, 33096)$, evaluate the function

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

IBM S/370 ($\beta = 16$) with FORTRAN:

type	p	$f(x, y)$
`REAL*4`	24	$1.172603\ldots$
`REAL*8`	53	$1.1726039400531\ldots$
`REAL*10`	64	$1.172603940053178\ldots$

Are floating point computations reliable?

Given the point $(x, y) = (77617, 33096)$, evaluate the function

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

IBM S/370 ($\beta = 16$) with FORTRAN:

type	p	$f(x, y)$
`REAL*4`	24	$1.172603\ldots$
`REAL*8`	53	$1.1726039400531\ldots$
`REAL*10`	64	$1.172603940053178\ldots$

Pentium III ($\beta = 2$) with C/C++ (gcc/g++):

type	p	$f(x, y)$
`float`	24	178702833214061281280
`double`	53	178702833214061281280
`long double`	64	178702833214061281280

Are floating point computations reliable?

Given the point $(x, y) = (77617, 33096)$, evaluate the function

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

IBM S/370 ($\beta = 16$) with FORTRAN:

type	p	$f(x,y)$
REAL*4	24	$1.172603\ldots$
REAL*8	53	$1.1726039400531\ldots$
REAL*10	64	$1.172603940053178\ldots$

Pentium III ($\beta = 2$) with C/C++ (gcc/g++):

type	p	$f(x,y)$
float	24	178702833214061281280
double	53	178702833214061281280
long double	64	178702833214061281280

Correct answer: $-0.8273960599\ldots$

UPPSALA UNIVERSITET

How do we control rounding errors?

Round each partial result both ways

If $x, y \in \mathbb{F}$ and $\star \in \{+, -, \times, \div\}$, we can enclose the exact result in an *interval*:

$$x \star y \in [\bigtriangledown(x \star y), \bigtriangleup(x \star y)].$$

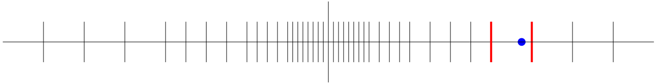
How do we control rounding errors?

Round each partial result both ways

If $x, y \in \mathbb{F}$ and $\star \in \{+, -, \times, \div\}$, we can enclose the exact result in an *interval*:

$$x \star y \in [\nabla(x \star y), \Delta(x \star y)].$$

Since all (modern) computers round with *maximal quality*, the interval is the smallest one that contains the exact result.



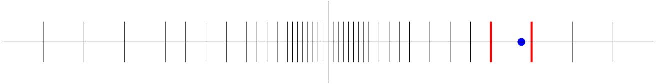
How do we control rounding errors?

Round each partial result both ways

If $x, y \in \mathbb{F}$ and $\star \in \{+, -, \times, \div\}$, we can enclose the exact result in an *interval*:

$$x \star y \in [\triangledown(x \star y), \triangle(x \star y)].$$

Since all (modern) computers round with *maximal quality*, the interval is the smallest one that contains the exact result.



Question

How do we compute with intervals? And why, really?

UPPSALA UNIVERSITET

Arithmetic over $\mathbb{IR}$

Definition

If $\star$ is one of the operators $+, -, \times, \div$, and if $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{IR}$, then

$$\boldsymbol{a} \star \boldsymbol{b} = \{a \star b \colon a \in \boldsymbol{a}, b \in \boldsymbol{b}\},$$

except that $\boldsymbol{a} \div \boldsymbol{b}$ is undefined if $0 \in \boldsymbol{b}$.

UPPSALA UNIVERSITET

Arithmetic over $\mathbb{IR}$

Definition

If $\star$ is one of the operators $+, -, \times, \div$, and if $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{IR}$, then

$$\boldsymbol{a} \star \boldsymbol{b} = \{a \star b \colon a \in \boldsymbol{a}, b \in \boldsymbol{b}\},$$

except that $\boldsymbol{a} \div \boldsymbol{b}$ is undefined if $0 \in \boldsymbol{b}$.

Simple arithmetic

$$\begin{aligned}
\boldsymbol{a} + \boldsymbol{b} &= [\underline{\boldsymbol{a}} + \underline{\boldsymbol{b}}, \overline{\boldsymbol{a}} + \overline{\boldsymbol{b}}] \\
\boldsymbol{a} - \boldsymbol{b} &= [\underline{\boldsymbol{a}} - \overline{\boldsymbol{b}}, \overline{\boldsymbol{a}} - \underline{\boldsymbol{b}}] \\
\boldsymbol{a} \times \boldsymbol{b} &= [\min\{\underline{\boldsymbol{a}}\underline{\boldsymbol{b}}, \underline{\boldsymbol{a}}\overline{\boldsymbol{b}}, \overline{\boldsymbol{a}}\underline{\boldsymbol{b}}, \overline{\boldsymbol{a}}\overline{\boldsymbol{b}}\}, \max\{\underline{\boldsymbol{a}}\underline{\boldsymbol{b}}, \underline{\boldsymbol{a}}\overline{\boldsymbol{b}}, \overline{\boldsymbol{a}}\underline{\boldsymbol{b}}, \overline{\boldsymbol{a}}\overline{\boldsymbol{b}}\}] \\
\boldsymbol{a} \div \boldsymbol{b} &= \boldsymbol{a} \times [1/\overline{\boldsymbol{b}}, 1/\underline{\boldsymbol{b}}], \quad \text{if } 0 \notin \boldsymbol{b}.
\end{aligned}$$

UPPSALA UNIVERSITET

Arithmetic over $\mathbb{IR}$

Definition

If $\star$ is one of the operators $+, -, \times, \div$, and if $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{IR}$, then

$$\boldsymbol{a} \star \boldsymbol{b} = \{a \star b \colon a \in \boldsymbol{a}, b \in \boldsymbol{b}\},$$

except that $\boldsymbol{a} \div \boldsymbol{b}$ is undefined if $0 \in \boldsymbol{b}$.

Simple arithmetic

$$\begin{aligned}
\boldsymbol{a} + \boldsymbol{b} &= [\underline{\boldsymbol{a}} + \underline{\boldsymbol{b}}, \overline{\boldsymbol{a}} + \overline{\boldsymbol{b}}] \\
\boldsymbol{a} - \boldsymbol{b} &= [\underline{\boldsymbol{a}} - \overline{\boldsymbol{b}}, \overline{\boldsymbol{a}} - \underline{\boldsymbol{b}}] \\
\boldsymbol{a} \times \boldsymbol{b} &= [\min\{\underline{\boldsymbol{a}}\underline{\boldsymbol{b}}, \underline{\boldsymbol{a}}\overline{\boldsymbol{b}}, \overline{\boldsymbol{a}}\underline{\boldsymbol{b}}, \overline{\boldsymbol{a}}\overline{\boldsymbol{b}}\}, \max\{\underline{\boldsymbol{a}}\underline{\boldsymbol{b}}, \underline{\boldsymbol{a}}\overline{\boldsymbol{b}}, \overline{\boldsymbol{a}}\underline{\boldsymbol{b}}, \overline{\boldsymbol{a}}\overline{\boldsymbol{b}}\}] \\
\boldsymbol{a} \div \boldsymbol{b} &= \boldsymbol{a} \times [1/\overline{\boldsymbol{b}}, 1/\underline{\boldsymbol{b}}], \quad \text{if } 0 \notin \boldsymbol{b}.
\end{aligned}$$

On a computer we use *directed rounding*, e.g.

$$\boldsymbol{a} + \boldsymbol{b} = [\bigtriangledown(\underline{\boldsymbol{a}} \oplus \underline{\boldsymbol{b}}), \bigtriangleup(\overline{\boldsymbol{a}} \oplus \overline{\boldsymbol{b}})].$$

Interval-valued functions

Range enclosure

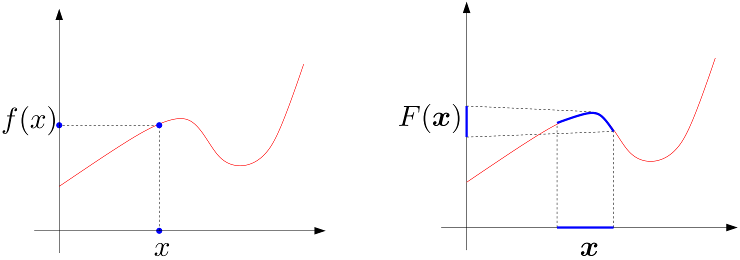
Extend a real-valued function f to an interval-valued F:

$$R(f;\boldsymbol{x}) = \{f(x) \colon x \in \boldsymbol{x}\} \subseteq F(\boldsymbol{x})$$

Range enclosure

Extend a real-valued function f to an interval-valued F:

$$R(f; \boldsymbol{x}) = \{f(x) \colon x \in \boldsymbol{x}\} \subseteq F(\boldsymbol{x})$$

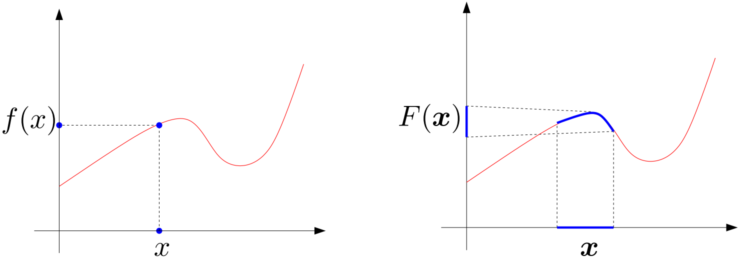


Interval-valued functions

Range enclosure

Extend a real-valued function f to an interval-valued F:

$$R(f;\boldsymbol{x}) = \{f(x) \colon x \in \boldsymbol{x}\} \subseteq F(\boldsymbol{x})$$



$y \notin F(\boldsymbol{x})$ implies that $f(x) \neq y$ for all $x \in \boldsymbol{x}$.

Some explicit formulas are given below:

Interval-valued functions

Some explicit formulas are given below:

$$
\begin{array}{lcll}
e^{\boldsymbol{x}} & = & [e^{\underline{\boldsymbol{x}}}, e^{\overline{\boldsymbol{x}}}] & \\
\sqrt{\boldsymbol{x}} & = & [\sqrt{\underline{\boldsymbol{x}}}, \sqrt{\overline{\boldsymbol{x}}}] & \text{if } 0 \leq \underline{\boldsymbol{x}} \\
\log \boldsymbol{x} & = & [\log \underline{\boldsymbol{x}}, \log \overline{\boldsymbol{x}}] & \text{if } 0 < \underline{\boldsymbol{x}} \\
\arctan \boldsymbol{x} & = & [\arctan \underline{\boldsymbol{x}}, \arctan \overline{\boldsymbol{x}}] & .
\end{array}
$$

Some explicit formulas are given below:

$$\begin{array}{lcll} e^{\boldsymbol{x}} & = & [e^{\underline{\boldsymbol{x}}}, e^{\overline{\boldsymbol{x}}}] & \\ \sqrt{\boldsymbol{x}} & = & [\sqrt{\underline{\boldsymbol{x}}}, \sqrt{\overline{\boldsymbol{x}}}] & \text{if } 0 \le \underline{\boldsymbol{x}} \\ \log \boldsymbol{x} & = & [\log \underline{\boldsymbol{x}}, \log \overline{\boldsymbol{x}}] & \text{if } 0 < \underline{\boldsymbol{x}} \\ \arctan \boldsymbol{x} & = & [\arctan \underline{\boldsymbol{x}}, \arctan \overline{\boldsymbol{x}}] & . \end{array}$$

Set $S^+ = \{2k\pi + \pi/2 \colon k \in \mathbb{Z}\}$ and $S^- = \{2k\pi - \pi/2 \colon k \in \mathbb{Z}\}$.
Then $\sin \boldsymbol{x}$ is given by

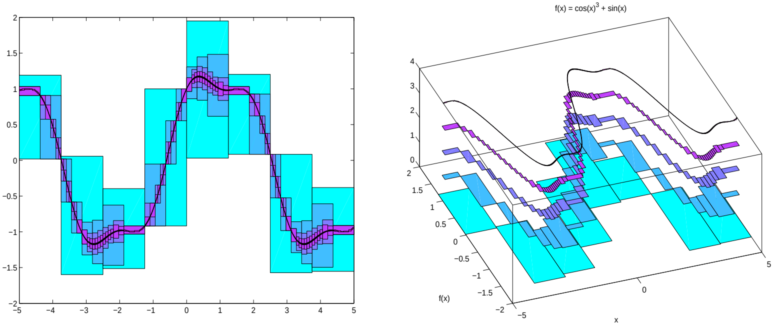
$$\begin{cases} [-1, 1] & : \text{ if } \boldsymbol{x} \cap S^- \neq \emptyset \text{ and } \boldsymbol{x} \cap S^+ \neq \emptyset, \\ [-1, \max\{\sin \underline{\boldsymbol{x}}, \sin \overline{\boldsymbol{x}}\}] & : \text{ if } \boldsymbol{x} \cap S^- \neq \emptyset \text{ and } \boldsymbol{x} \cap S^+ = \emptyset, \\ [\min\{\sin \underline{\boldsymbol{x}}, \sin \overline{\boldsymbol{x}}\}, 1] & : \text{ if } \boldsymbol{x} \cap S^- = \emptyset \text{ and } \boldsymbol{x} \cap S^+ \neq \emptyset, \\ [\min\{\sin \underline{\boldsymbol{x}}, \sin \overline{\boldsymbol{x}}\}, \max\{\sin \underline{\boldsymbol{x}}, \sin \overline{\boldsymbol{x}}\}] & : \text{ if } \boldsymbol{x} \cap S^- = \emptyset \text{ and } \boldsymbol{x} \cap S^+ = \emptyset. \end{cases}$$

A controlled discretization

We can now select and adapt the level of discretization

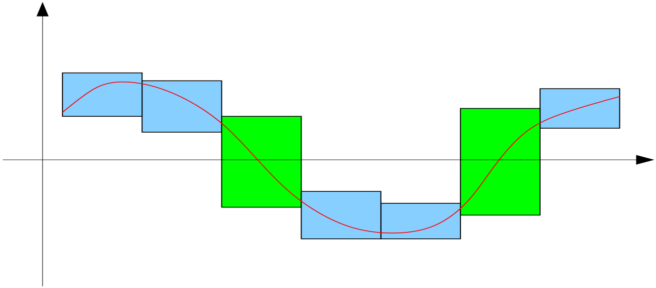
A controlled discretization

We can now select and adapt the level of discretization

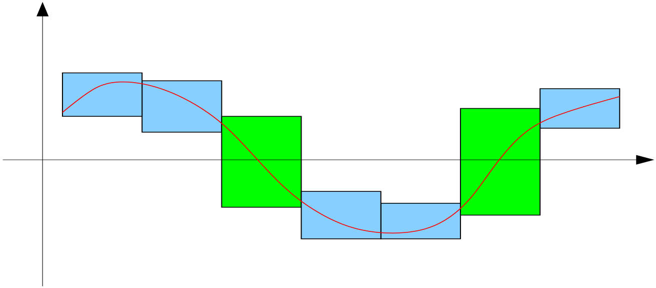


Various levels of discretization of $f(x) = \cos^3 x + \sin x$.

Solving non-linear equations

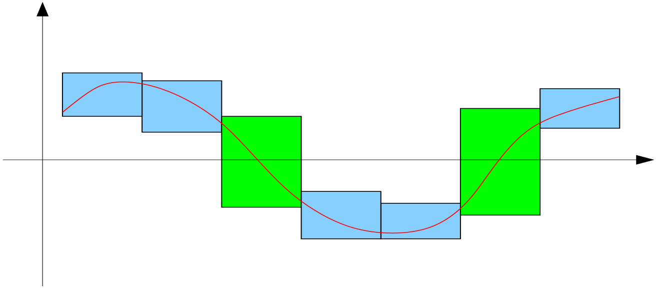


Solving non-linear equations



Consider everything. Keep what is good.
Avoid evil whenever you recognize it.
St. Paul, ca. 50 A.D. (The Bible, 1 Thess. 5:21-22)

Solving non-linear equations



Consider everything. Keep what is good.
Avoid evil whenever you recognize it.
St. Paul, ca. 50 A.D. (The Bible, 1 Thess. 5:21-22)

No solutions can be missed!

Solving non-linear equation

The code is transparent and natural

```
function bisect(fcnName, X, tol)
f = inline(fcnName);
if ( 0 <= f(X) )              % If f(X) contains zero...
    if Diam(X) < tol          % and the tolerance is met...
        X                     % print the interval X.
    else                      % Otherwise, divide and conquer.
        bisect(fcnName, interval(Inf(X), Mid(X)), tol);
        bisect(fcnName, interval(Mid(X), Sup(X)), tol);
    end
end
```

Solving non-linear equation

The code is transparent and natural

```
function bisect(fcnName, X, tol)
f = inline(fcnName);
if ( 0 <= f(X) )              % If f(X) contains zero...
    if Diam(X) < tol          % and the tolerance is met...
        X                     % print the interval X.
    else                      % Otherwise, divide and conquer.
        bisect(fcnName, interval(Inf(X), Mid(X)), tol);
        bisect(fcnName, interval(Mid(X), Sup(X)), tol);
    end
end
```

Nice property

If F is well-defined on the domain, the algorithm produces an enclosure of *all* zeros of f. [No existence is established, however.]

Existence and uniqueness

Existence and uniqueness require *fixed point* theorems.

Existence and uniqueness

Existence and uniqueness require *fixed point* theorems.

Brouwer's fixed point theorem

Let B be homeomorhpic to the closed unit ball in $\mathbb{R}^n$. Then given any continuous mapping $f: B \to B$ there exists $x \in B$ such that $f(x) = x$.

Existence and uniqueness

Existence and uniqueness require *fixed point* theorems.

Brouwer's fixed point theorem

Let B be homeomorhpic to the closed unit ball in $\mathbb{R}^n$. Then given any continuous mapping $f: B \to B$ there exists $x \in B$ such that $f(x) = x$.

Schauder's fixed point theorem

Let X be a normed vector space, and let $K \subset X$ be a non-empty, compact, and convex set. Then given any continuous mapping $f: K \to K$ there exists $x \in K$ such that $f(x) = x$.

UPPSALA UNIVERSITET

Existence and uniqueness

Existence and uniqueness require *fixed point* theorems.

Brouwer's fixed point theorem

Let B be homeomorhpic to the closed unit ball in $\mathbb{R}^n$. Then given any continuous mapping $f\colon B \to B$ there exists $x \in B$ such that $f(x) = x$.

Schauder's fixed point theorem

Let X be a normed vector space, and let $K \subset X$ be a non-empty, compact, and convex set. Then given any continuous mapping $f\colon K \to K$ there exists $x \in K$ such that $f(x) = x$.

Banach's fixed point theorem

If f is a contraction defined on a complete metric space X, then there exists a unique $x \in X$ such that $f(x) = x$.

Newton's method in $\mathbb{IR}$

Theorem

Let $f \in C^1(\mathbb{R}, \mathbb{R})$, *and set* $\check{x} = \mathrm{mid}(\boldsymbol{x})$. *We define*

$$N_f(\boldsymbol{x}) \stackrel{\text{def}}{=} N_f(\boldsymbol{x}, \check{x}) = \check{x} - [DF(\boldsymbol{x})]^{-1} f(\check{x}).$$

If $N_f(\boldsymbol{x})$ *is well-defined, then the following statements hold:*

UPPSALA UNIVERSITET

Newton's method in $\mathbb{R}$

Theorem

Let $f \in C^1(\mathbb{R}, \mathbb{R})$, *and set* $\check{x} = \mathrm{mid}(\boldsymbol{x})$. *We define*

$$N_f(\boldsymbol{x}) \overset{\text{def}}{=} N_f(\boldsymbol{x}, \check{x}) = \check{x} - [DF(\boldsymbol{x})]^{-1} f(\check{x}).$$

If $N_f(\boldsymbol{x})$ *is well-defined, then the following statements hold:*

(1) *if* $\boldsymbol{x}$ *contains a zero* x^* *of* f, *then so does* $N_f(\boldsymbol{x}) \cap \boldsymbol{x}$;

Newton's method in $\mathbb{IR}$

Theorem

Let $f \in C^1(\mathbb{R}, \mathbb{R})$, *and set* $\check{x} = \mathrm{mid}(\boldsymbol{x})$. *We define*

$$N_f(\boldsymbol{x}) \stackrel{\text{def}}{=} N_f(\boldsymbol{x}, \check{x}) = \check{x} - [DF(\boldsymbol{x})]^{-1} f(\check{x}).$$

If $N_f(\boldsymbol{x})$ *is well-defined, then the following statements hold:*

(1) *if* $\boldsymbol{x}$ *contains a zero* x^* *of* f, *then so does* $N_f(\boldsymbol{x}) \cap \boldsymbol{x}$;

(2) *if* $N_f(\boldsymbol{x}) \cap \boldsymbol{x} = \emptyset$, *then* $\boldsymbol{x}$ *contains no zeros of* f;

Newton's method in $\mathbb{IR}$

Theorem

Let $f \in C^1(\mathbb{R}, \mathbb{R})$, and set $\check{x} = \mathrm{mid}(\boldsymbol{x})$. We define

$$N_f(\boldsymbol{x}) \stackrel{\text{def}}{=} N_f(\boldsymbol{x}, \check{x}) = \check{x} - [DF(\boldsymbol{x})]^{-1} f(\check{x}).$$

If $N_f(\boldsymbol{x})$ is well-defined, then the following statements hold:

(1) *if $\boldsymbol{x}$ contains a zero x^* of f, then so does $N_f(\boldsymbol{x}) \cap \boldsymbol{x}$;*

(2) *if $N_f(\boldsymbol{x}) \cap \boldsymbol{x} = \emptyset$, then $\boldsymbol{x}$ contains no zeros of f;*

(3) *if $N_f(\boldsymbol{x}) \subseteq \boldsymbol{x}$, then $\boldsymbol{x}$ contains a unique zero of f.*

Newton's method in $\mathbb{IR}$

Theorem

Let $f \in C^1(\mathbb{R}, \mathbb{R})$, and set $\check{x} = \mathrm{mid}(\boldsymbol{x})$. We define

$$N_f(\boldsymbol{x}) \stackrel{\text{def}}{=} N_f(\boldsymbol{x}, \check{x}) = \check{x} - [DF(\boldsymbol{x})]^{-1} f(\check{x}).$$

If $N_f(\boldsymbol{x})$ is well-defined, then the following statements hold:

(1) *if $\boldsymbol{x}$ contains a zero x^* of f, then so does $N_f(\boldsymbol{x}) \cap \boldsymbol{x}$;*

(2) *if $N_f(\boldsymbol{x}) \cap \boldsymbol{x} = \emptyset$, then $\boldsymbol{x}$ contains no zeros of f;*

(3) *if $N_f(\boldsymbol{x}) \subseteq \boldsymbol{x}$, then $\boldsymbol{x}$ contains a unique zero of f.*

Similar statements hold for the *Krawczyk operator*

$$K_f(\boldsymbol{x}) \stackrel{\text{def}}{=} \check{x} - [Df(\check{x})]^{-1} f(\check{x}) - \left(1 - [Df(\check{x})]^{-1} F'(\boldsymbol{x})\right)[-r, r],$$

where we use the notation $r = \mathrm{rad}(\boldsymbol{x})$.

UPPSALA UNIVERSITET

Newton's method in $\mathbb{IR}$

Algorithm

Starting from an initial search region $\boldsymbol{x}_0$, we form the sequence

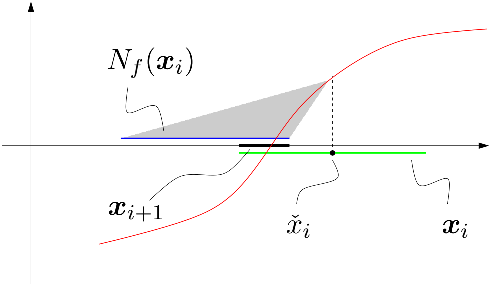
$$\boldsymbol{x}_{i+1} = N_f(\boldsymbol{x}_i) \cap \boldsymbol{x}_i \qquad i = 0, 1, \ldots.$$

Newton's method in $\mathbb{R}$

Algorithm

Starting from an initial search region $\boldsymbol{x}_0$, we form the sequence

$$\boldsymbol{x}_{i+1} = N_f(\boldsymbol{x}_i) \cap \boldsymbol{x}_i \qquad i = 0, 1, \ldots .$$

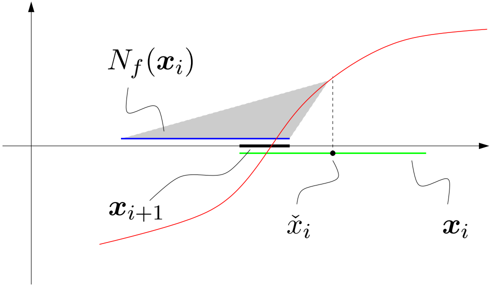


Newton's method in $\mathbb{R}$

Algorithm

Starting from an initial search region $\boldsymbol{x}_0$, we form the sequence

$$\boldsymbol{x}_{i+1} = N_f(\boldsymbol{x}_i) \cap \boldsymbol{x}_i \qquad i = 0, 1, \ldots .$$



Performance

If well-defined, this method is never worse than bisection, and it converges quadratically fast under mild conditions.

Example

Take $f(x) = -2.001 + 3x - x^3$ and start with $\boldsymbol{x}_0 = [-3, -3/2]$.

Newton's method in $\mathbb{IR}$

Example

Take $f(x) = -2.001 + 3x - x^3$ and start with $\boldsymbol{x}_0 = [-3, -3/2]$.

```
X(0) = [-3.000000000000000,-1.500000000000000]; rad = 7.50000e-01
X(1) = [-2.140015625000001,-1.546099999999996]; rad = 2.96958e-01
X(2) = [-2.140015625000001,-1.961277398284108]; rad = 8.93691e-02
X(3) = [-2.006849239640351,-1.995570580247208]; rad = 5.63933e-03
X(4) = [-2.000120104486270,-2.000103608530276]; rad = 8.24798e-06
X(5) = [-2.000111102890393,-2.000111102873815]; rad = 8.28893e-12
X(6) = [-2.000111102881727,-2.000111102881724]; rad = 1.55431e-15
X(7) = [-2.000111102881727,-2.000111102881724]; rad = 1.55431e-15
Finite convergence!
Unique root in -2.00011110288172 +- 1.555e-15
```

UPPSALA UNIVERSITET

Newton's method in $\mathbb{R}$

Example

Take $f(x) = -2.001 + 3x - x^3$ and start with $\boldsymbol{x}_0 = [-3, -3/2]$.

```
X(0) = [-3.000000000000000,-1.500000000000000]; rad = 7.50000e-01
X(1) = [-2.140015625000001,-1.546099999999996]; rad = 2.96958e-01
X(2) = [-2.140015625000001,-1.961277398284108]; rad = 8.93691e-02
X(3) = [-2.006849239640351,-1.995570580247208]; rad = 5.63933e-03
X(4) = [-2.000120104486270,-2.000103608530276]; rad = 8.24798e-06
X(5) = [-2.000111102890393,-2.000111102873815]; rad = 8.28893e-12
X(6) = [-2.000111102881727,-2.000111102881724]; rad = 1.55431e-15
X(7) = [-2.000111102881727,-2.000111102881724]; rad = 1.55431e-15
Finite convergence!
Unique root in -2.00011110288172 +- 1.555e-15
```

Stopping condition

Stop when no further improvement takes place.

The Krawczyk method with bisection

When we have several zeros, we must bisect to isolate the zeros.

Example

Take $f(x) = \sin(e^x + 1)$ and start with $\boldsymbol{x}_0 = [0, 3]$.

The Krawczyk method with bisection

When we have several zeros, we must bisect to isolate the zeros.

Example

Take $f(x) = \sin(e^x + 1)$ and start with $\boldsymbol{x}_0 = [0, 3]$.

```
Domain          : [0, 3]
Tolerance       : 1e-10
Function calls  : 71
 Unique zero in the interval 0.761549782880[8890,9006]
 Unique zero in the interval 1.664529193[6825445,7060436]
 Unique zero in the interval 2.131177121086[2673,3558]
 Unique zero in the interval 2.4481018026567[773,801]
 Unique zero in the interval 2.68838906601606[36,68]
 Unique zero in the interval 2.8819786295709[728,1555]
```

The Krawczyk method with bisection

When we have several zeros, we must bisect to isolate the zeros.

Example

Take $f(x) = \sin(e^x + 1)$ and start with $\boldsymbol{x}_0 = [0, 3]$.

```
Domain          : [0, 3]
Tolerance       : 1e-10
Function calls  : 71
 Unique zero in the interval 0.761549782880[8890,9006]
 Unique zero in the interval 1.664529193[6825445,7060436]
 Unique zero in the interval 2.131177121086[2673,3558]
 Unique zero in the interval 2.4481018026567[773,801]
 Unique zero in the interval 2.68838906601606[36,68]
 Unique zero in the interval 2.8819786295709[728,1555]
```

Applications

Counting short periodic orbits for ODEs [Z. Galias]
Measuring the stable regions of the quadratic map [D. Wilczak]

Implicit curves (level sets)

Example

Draw the level-set defined by

$$f(x,y) = \sin\left(\cos x^2 + 10 \sin y^2\right) - y \cos x = 0;$$

restricted to the domain $[-5,5] \times [-5,5]$.

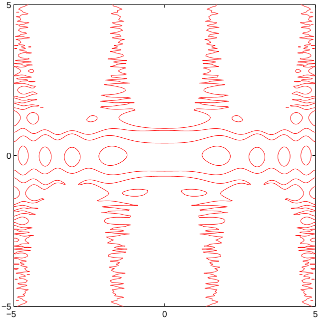
Example

Draw the level-set defined by

$$f(x,y) = \sin\left(\cos x^2 + 10\sin y^2\right) - y\cos x = 0;$$

restricted to the domain $[-5,5] \times [-5,5]$.

MATLAB produces the following picture:



Implicit curves (level sets)

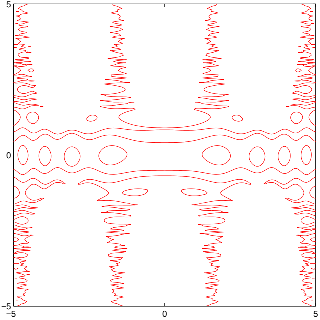
Example

Draw the level-set defined by

$$f(x,y) = \sin\left(\cos x^2 + 10 \sin y^2\right) - y \cos x = 0;$$

restricted to the domain $[-5,5] \times [-5,5]$.

MATLAB produces the following picture:



According to the same m-file, the level set defined by $|f(x,y)| = 0$, however, appears to be empty.

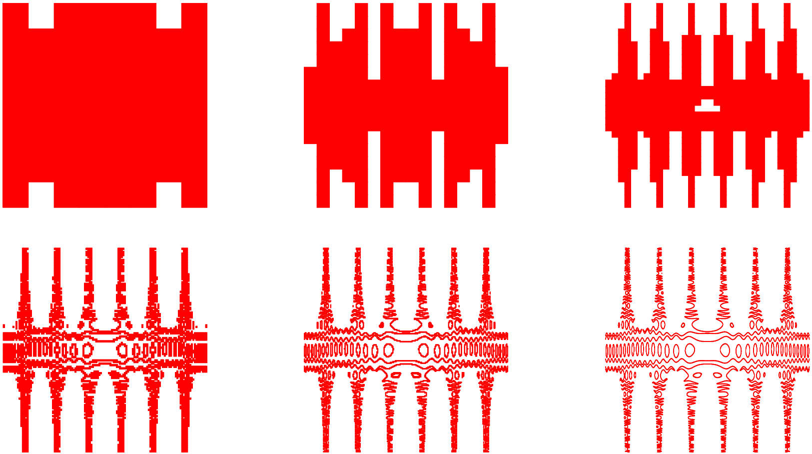
But this is the same set!!!

Implicit curves

The (increasingly tight) set-valued enclosures in both cases are

Implicit curves

The (increasingly tight) set-valued enclosures in both cases are



UPPSALA UNIVERSITET

Quadrature

Example (A bonus problem)

Compute the integral $\int_0^8 \sin\left(x + e^x\right) dx$.

Quadrature

Example (A bonus problem)

Compute the integral $\int_0^8 \sin\left(x + e^x\right) dx$.

A regular MATLAB session:

```
>> q = quad('sin(x + exp(x))', 0, 8)
q =
    0.251102722027180
```

Example (A bonus problem)

Compute the integral $\int_0^8 \sin\left(x + e^x\right) dx$.

A regular MATLAB session:

```
>> q = quad('sin(x + exp(x))', 0, 8)
q =
    0.251102722027180
```

Using an adaptive validated integrator:

```
$$ ./adQuad 0 8 4 1e-4
Partitions: 8542
CPU time   : 0.52 seconds
Integral   : 0.347[3863144222905,4140198005782]
```

Example (A bonus problem)

Compute the integral $\int_0^8 \sin\left(x + e^x\right) dx$.

A regular MATLAB session:

```
>> q = quad('sin(x + exp(x))', 0, 8)
q =
    0.251102722027180
```

Using an adaptive validated integrator:

```
$$ ./adQuad 0 8 4 1e-4
Partitions: 8542
CPU time   : 0.52 seconds
Integral   : 0.347[3863144222905,4140198005782]

$$ ./adQuad 0 8 20 1e-10
Partitions: 874
CPU time   : 0.45 seconds
Integral   : 0.3474001726[492276,652638]
```

Parameter estimation

Problem formulation

Given a finitely parametrized model function together with some (noisy) data, and a search region $\mathcal{P}$ in parameter space:

$$\underbrace{y = f(x; p)}_{\text{model}} \qquad \underbrace{\{(x_i, y_i)\}_{i=1}^{N}}_{\text{data}} \qquad \underbrace{p \in \mathcal{P}}_{\text{space}}$$

try to find parameters that give a *good agreement* between the data and the model. [A classic inverse problem]

Parameter estimation

Problem formulation

Given a finitely parametrized model function together with some (noisy) data, and a search region $\mathcal{P}$ in parameter space:

$$\underbrace{y = f(x;p)}_{\text{model}} \qquad \underbrace{\{(x_i, y_i)\}_{i=1}^N}_{\text{data}} \qquad \underbrace{p \in \mathcal{P}}_{\text{space}}$$

try to find parameters that give a *good agreement* between the data and the model. [A classic inverse problem]

- **Existence:** with noisy data, or with an incorrect model, there is usually *no* parameter that produces a perfect fit.

Parameter estimation

Problem formulation

Given a finitely parametrized model function together with some (noisy) data, and a search region $\mathcal{P}$ in parameter space:

$$\underbrace{y = f(x;p)}_{\text{model}} \qquad \underbrace{\{(x_i, y_i)\}_{i=1}^{N}}_{\text{data}} \qquad \underbrace{p \in \mathcal{P}}_{\text{space}}$$

try to find parameters that give a *good agreement* between the data and the model. [A classic inverse problem]

- **Existence:** with noisy data, or with an incorrect model, there is usually *no* parameter that produces a perfect fit.
- **Uniqueness:** even with *unlimited* amounts of *exact* data, there might not exist a unique solution $p^\sharp \in \mathcal{P}$ such that

$$f(x_i; p^\sharp) = y_i \qquad i = 1, \dots, N.$$

UPPSALA UNIVERSITET

Parameter estimation

Problem formulation

Given a finitely parametrized model function together with some (noisy) data, and a search region $\mathcal{P}$ in parameter space:

$$\underbrace{y = f(x;p)}_{\text{model}} \qquad \underbrace{\{(x_i, y_i)\}_{i=1}^{N}}_{\text{data}} \qquad \underbrace{p \in \mathcal{P}}_{\text{space}}$$

try to find parameters that give a *good agreement* between the data and the model. [A classic inverse problem]

- **Existence:** with noisy data, or with an incorrect model, there is usually *no* parameter that produces a perfect fit.
- **Uniqueness:** even with *unlimited* amounts of *exact* data, there might not exist a unique solution $p^\sharp \in \mathcal{P}$ such that

$$f(x_i; p^\sharp) = y_i \qquad i = 1, \ldots, N.$$

- **Instability:** inverse problems can be extremely ill-conditioned.

UPPSALA UNIVERSITET

A statistical approach

Use a (weighted) least-squares approach to find the best parameter:

$$\operatorname*{argmin}_{p\in\mathcal{P}} \sum_{i=1}^{N} w_i |f(x_i;p) - y_i|^2.$$

A statistical approach

Use a (weighted) least-squares approach to find the best parameter:

$$\operatorname*{argmin}_{p \in \mathcal{P}} \sum_{i=1}^{N} w_i |f(x_i; p) - y_i|^2.$$

- If the parameters enter f linearly, this is "straight-forward".

Introduction

A statistical approach

Use a (weighted) least-squares approach to find the best parameter:

$$\operatorname*{argmin}_{p \in \mathcal{P}} \sum_{i=1}^{N} w_i |f(x_i;p) - y_i|^2.$$

- If the parameters enter f linearly, this is “straight-forward”.
- Otherwise, we have moved the problem to *global optimization*.

UPPSALA UNIVERSITET

A statistical approach

Use a (weighted) least-squares approach to find the best parameter:

$$\operatorname*{argmin}_{p \in \mathcal{P}} \sum_{i=1}^{N} w_i |f(x_i; p) - y_i|^2.$$

- If the parameters enter f linearly, this is "straight-forward".
- Otherwise, we have moved the problem to *global optimization*.
- The selection of weights is almost always a delicate issue.

Introduction

A statistical approach

Use a (weighted) least-squares approach to find the best parameter:

$$\operatorname*{argmin}_{p\in\mathcal{P}} \sum_{i=1}^{N} w_i |f(x_i;p) - y_i|^2.$$

- If the parameters enter f linearly, this is "straight-forward".
- Otherwise, we have moved the problem to *global optimization*.
- The selection of weights is almost always a delicate issue.

A set-valued approach

Locate nearby models that are *consistent* with nearby data:

$$f(x;p) \longrightarrow f(x;\boldsymbol{p}) \qquad\qquad (x_i, y_i) \longrightarrow (x_i, \boldsymbol{y}_i).$$

Set-valued computations

Points versus sets in parameter space

We move from the *point-valued* model function $f(x; p)$ to the *set-valued* version $f(x; \boldsymbol{p})$.

Set-valued computations

Points versus sets in parameter space

We move from the *point-valued* model function $f(x;p)$ to the *set-valued* version $f(x;\boldsymbol{p})$.

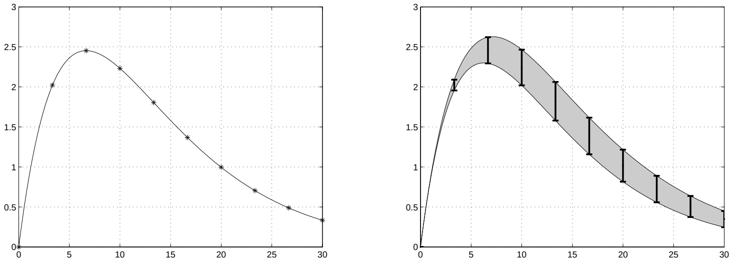


Figure: (a) $p = 0.15$, a point in $\mathcal{P}$. (b) $\boldsymbol{p} = [0.14, 0.16]$, a subset of $\mathcal{P}$. The model function is $f(x;p) = xe^{-px}$, and 10 samples are shown.

Parameter estimation

Strategy

Adaptively bisect the parameter space into sub-boxes: $\mathcal{P} = \cup_{j=1}^{K} \boldsymbol{p}_j$ and examine each $\boldsymbol{p}_j$ separately. [Good for parallelisation]

Parameter estimation

Strategy

Adaptively bisect the parameter space into sub-boxes: $\mathcal{P} = \cup_{j=1}^{K} \boldsymbol{p}_j$ and examine each $\boldsymbol{p}_j$ separately. [Good for parallelisation]

Each sub-box $\boldsymbol{p}$ of the parameter space falls into one of three categories:

Parameter estimation

Strategy

Adaptively bisect the parameter space into sub-boxes: $\mathcal{P} = \cup_{j=1}^{K} \boldsymbol{p}_j$ and examine each $\boldsymbol{p}_j$ separately. [Good for parallelisation]

Each sub-box $\boldsymbol{p}$ of the parameter space falls into one of three categories:

(1) consistent

if $f(x_i; \boldsymbol{p}) \subset \boldsymbol{y}_i$ for all $i = 0, \ldots, N$. SAVE

UPPSALA UNIVERSITET

Parameter estimation

Strategy

Adaptively bisect the parameter space into sub-boxes: $\mathcal{P} = \cup_{j=1}^{K} \boldsymbol{p}_j$ and examine each $\boldsymbol{p}_j$ separately. [Good for parallelisation]

Each sub-box $\boldsymbol{p}$ of the parameter space falls into one of three categories:

(1) consistent

if $f(x_i; \boldsymbol{p}) \subset \boldsymbol{y}_i$ for all $i = 0, \ldots, N$. SAVE

(2) inconsistent

if $f(x_i; \boldsymbol{p}) \cap \boldsymbol{y}_i = \emptyset$ for at least one i. DROP

Parameter estimation

Strategy

Adaptively bisect the parameter space into sub-boxes: $\mathcal{P} = \cup_{j=1}^{K} \boldsymbol{p}_j$ and examine each $\boldsymbol{p}_j$ separately. [Good for parallelisation]

Each sub-box $\boldsymbol{p}$ of the parameter space falls into one of three categories:

(1) consistent

if $f(x_i; \boldsymbol{p}) \subset \boldsymbol{y}_i$ for all $i = 0, \ldots, N$. SAVE

(2) inconsistent

if $f(x_i; \boldsymbol{p}) \cap \boldsymbol{y}_i = \emptyset$ for at least one i. DROP

(3) undetermined

not (1), but $f(x_i; \boldsymbol{p}) \cap \boldsymbol{y}_i \neq \emptyset$ for all $i = 0, \ldots, N$. SPLIT

Parameter estimation

Example

Consider the model function

$$f(x; p_1, p_2) = 5e^{-p_1 x} - 4 \times 10^{-6} e^{-p_2 x}$$

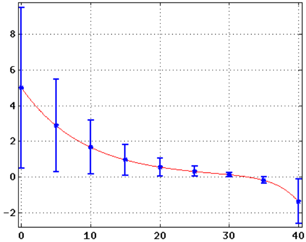
with samples taken at $x = 0, 5 \ldots, 40$ using $p^\sharp = (0.11, -0.32)$. Accepting a relative noise level of 90%, we get the following set of consistent parameters:

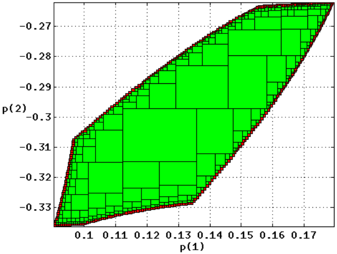
Example

Consider the model function

$$f(x; p_1, p_2) = 5e^{-p_1 x} - 4 \times 10^{-6} e^{-p_2 x}$$

with samples taken at $x = 0, 5 \ldots, 40$ using $p^\sharp = (0.11, -0.32)$. Accepting a relative noise level of 90%, we get the following set of consistent parameters:

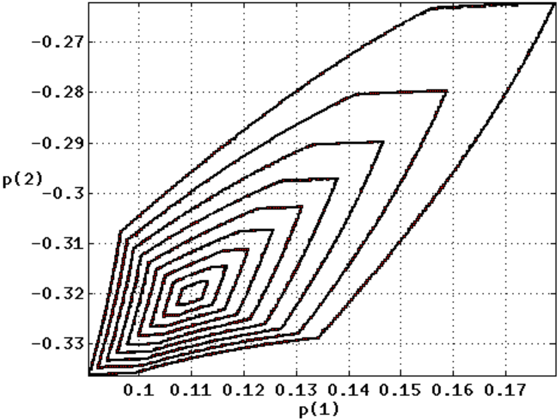




UPPSALA UNIVERSITET

Parameter estimation

Varying the relative noise levels between $10, 20 \ldots, 90\%$, we get the following indeterminate sets.



Constraint propagation

Constraining the parameter/data space

We use set-valued constraint propagation to quickly discard inconsistent regions in the data and the parameter space.

This is done without bisection!

Constraint propagation

Constraining the parameter/data space

We use set-valued constraint propagation to quickly discard inconsistent regions in the data and the parameter space.

This is done without bisection!

Example

Let $f(x;p) = xe^{-px}$, and consider the situation $\boldsymbol{p} = [0,1]$ and $(x, \boldsymbol{y}) = (2, [1,3])$.

Constraint propagation

Constraining the parameter/data space

We use set-valued constraint propagation to quickly discard inconsistent regions in the data and the parameter space.

This is done without bisection!

Example

Let $f(x; p) = xe^{-px}$, and consider the situation $\boldsymbol{p} = [0, 1]$ and $(x, \boldsymbol{y}) = (2, [1, 3])$. By a forward (interval) evaluation, we have

$$f(2; [0, 1]) = 2e^{-2[0,1]} = 2e^{[-2,0]} = 2[e^{-2}, 1] = [2e^{-2}, 2].$$

UPPSALA UNIVERSITET

Constraint propagation

Constraining the parameter/data space

We use set-valued constraint propagation to quickly discard inconsistent regions in the data and the parameter space.

This is done without bisection!

Example

Let $f(x;p) = xe^{-px}$, and consider the situation $\boldsymbol{p} = [0,1]$ and $(x, \boldsymbol{y}) = (2, [1,3])$. By a forward (interval) evaluation, we have

$$f(2; [0,1]) = 2e^{-2[0,1]} = 2e^{[-2,0]} = 2[e^{-2}, 1] = [2e^{-2}, 2].$$

This allows us to contract the data range according to

$$\boldsymbol{y} \mapsto \boldsymbol{y} \cap f(x; \boldsymbol{p}) = [1,3] \cap [2e^{-2}, 2] = [1,2].$$

Constraint propagation

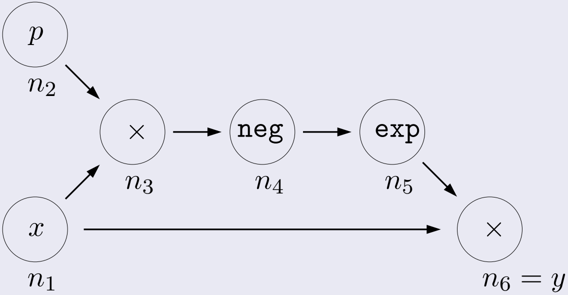
Directed Acyclic Graphs (DAGs)

We use a DAG representation of the model function to automate constraint propagations.

Constraint propagation

Directed Acyclic Graphs (DAGs)

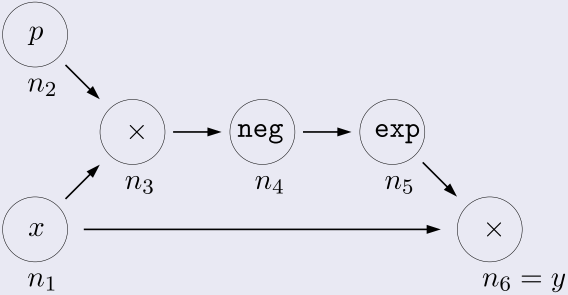
We use a DAG representation of the model function to automate constraint propagation.



Constraint propagation

Directed Acyclic Graphs (DAGs)

We use a DAG representation of the model function to automate constraint propagations.



$$
\begin{aligned}
n_1 &= x \\
n_2 &= p \\
n_3 &= n_1 \times n_2 \\
n_4 &= -n_3 \\
n_5 &= e^{n_4} \\
n_6 &= n_1 \times n_5.
\end{aligned}
$$

Figure: The DAG representation of a forward sweep of $y = xe^{-px}$, together with the corresponding code list.

UPPSALA UNIVERSITET

Constraint propagation

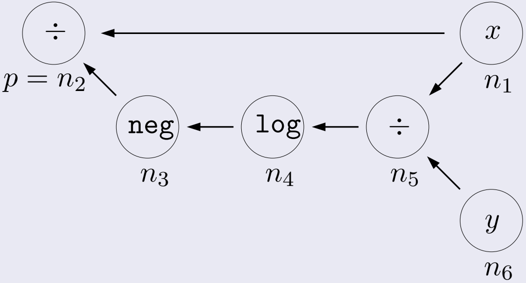
Directed Acyclic Graphs (DAGs)

We can propagate constraints from data to the parameter by moving *backwards* in the code list.

Constraint propagation

Directed Acyclic Graphs (DAGs)

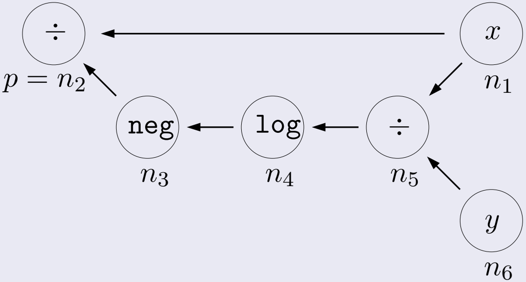
We can propagate constraints from data to the parameter by moving *backwards* in the code list.



Constraint propagation

Directed Acyclic Graphs (DAGs)

We can propagate constraints from data to the parameter by moving *backwards* in the code list.



$$
\begin{aligned}
n_5 &= n_6 \div n_1 \\
n_4 &= \log n_5 \\
n_3 &= -n_4 \\
n_2 &= n_3 \div n_1.
\end{aligned}
$$

Figure: The DAG representation of a backward sweep of $y = xe^{-px}$, together with the corresponding code list.

Example

Again, we work on the model function $y = f(x; p) = xe^{-px}$, but now with the data $(x, \boldsymbol{y}) = (2, [1, 3])$, together with the parameter domain $\boldsymbol{p} = [0, 1]$.

Constraint propagation

Example

Again, we work on the model function $y = f(x; p) = xe^{-px}$, but now with the data $(x, \boldsymbol{y}) = (2, [1, 3])$, together with the parameter domain $\boldsymbol{p} = [0, 1]$. The forward sweep, performed in Example 7, contracts the interval data to $\boldsymbol{y} = [1, 2]$.

Example

Again, we work on the model function $y = f(x;p) = xe^{-px}$, but now with the data $(x, \boldsymbol{y}) = (2, [1,3])$, together with the parameter domain $\boldsymbol{p} = [0,1]$. The forward sweep, performed in Example 7, contracts the interval data to $\boldsymbol{y} = [1,2]$. Performing a backward sweep contracts the interval parameter to $\boldsymbol{p} = [0, \frac{1}{2}\log 2]$:

$$
\begin{array}{rcccccc}
n_5 & = & n_6 \div n_1 & = & [1,2] \div 2 & = & [\frac{1}{2}, 1] \\
n_4 & = & \log n_5 & = & \log [\frac{1}{2}, 1] & = & [-\log 2, 0] \\
n_3 & = & -n_4 & = & [0, \log 2] & & \\
n_2 & = & n_3 \div n_1 & = & [0, \log 2] \div 2 & \approx & [0, 0.34657359].
\end{array}
$$

Constraint propagation

Example

Again, we work on the model function $y = f(x;p) = xe^{-px}$, but now with the data $(x, \boldsymbol{y}) = (2, [1,3])$, together with the parameter domain $\boldsymbol{p} = [0,1]$. The forward sweep, performed in Example 7, contracts the interval data to $\boldsymbol{y} = [1,2]$. Performing a backward sweep contracts the interval parameter to $\boldsymbol{p} = [0, \frac{1}{2}\log 2]$:

$$
\begin{array}{ccccccc}
n_5 & = & n_6 \div n_1 & = & [1,2] \div 2 & = & [\frac{1}{2}, 1] \\
n_4 & = & \log n_5 & = & \log\,[\frac{1}{2}, 1] & = & [-\log 2, 0] \\
n_3 & = & -n_4 & = & [0, \log 2] & & \\
n_2 & = & n_3 \div n_1 & = & [0, \log 2] \div 2 & \approx & [0, 0.34657359].
\end{array}
$$

Note that, in one forward/backward sweep, we managed to exclude over 65% of the parameter domain, at the same time reducing the data uncertainty by 50%.

Mixed-effects models

Mixed-effects models

We are given several data sets (trajectories) corresponding to k different "individuals":

$$
\begin{aligned}
\text{individual}_1 : &\quad (x_{11}, y_{11}), (x_{12}, y_{12}), \dots, (x_{1N}, y_{1N_1}) \\
\text{individual}_2 : &\quad (x_{21}, y_{21}), (x_{22}, y_{22}), \dots, (x_{2N}, y_{2N_2}) \\
&\qquad \vdots \qquad\qquad \vdots \qquad\qquad\qquad \vdots \\
\text{individual}_k : &\quad (x_{k1}, y_{k1}), (x_{k2}, y_{k2}), \dots, (x_{kN}, y_{kN_k}).
\end{aligned}
$$

Some model parameters are equal (shared) for all individuals, and some are distinct.

UPPSALA UNIVERSITET

Mixed-effects models

Mixed-effects models

We are given several data sets (trajectories) corresponding to k different "individuals":

$$\begin{aligned}
\text{individual}_1: &\quad (x_{11}, y_{11}), (x_{12}, y_{12}), \ldots, (x_{1N}, y_{1N_1}) \\
\text{individual}_2: &\quad (x_{21}, y_{21}), (x_{22}, y_{22}), \ldots, (x_{2N}, y_{2N_2}) \\
&\quad \vdots \\
\text{individual}_k: &\quad (x_{k1}, y_{k1}), (x_{k2}, y_{k2}), \ldots, (x_{kN}, y_{kN_k}).
\end{aligned}$$

Some model parameters are equal (shared) for all individuals, and some are distinct.

- We need to consider all individuals simultaneously. Otherwise the number of unknown parameters may be too large.

UPPSALA UNIVERSITET

A mixed-effects model for orange tree truncs

Example

We will apply our method to the following scenario:

A mixed-effects model for orange tree truncs

Example

We will apply our method to the following scenario:

$$\text{Model function:} \qquad f(x;p) = \frac{p_1}{1 + p_2 e^{p_3 x}}$$

A mixed-effects model for orange tree truncs

Example

We will apply our method to the following scenario:

Model function: $$f(x;p) = \frac{p_1}{1 + p_2 e^{p_3 x}}$$

Individual parameter: $$p_{i1} = p_1^\sharp + \eta_i, \quad \eta_i \sim N(0, \sigma^2)$$

A mixed-effects model for orange tree truncs

Example

We will apply our method to the following scenario:

$$\begin{aligned}
\text{Model function:} &\quad f(x;p) = \frac{p_1}{1 + p_2 e^{p_3 x}} \\
\text{Individual parameter:} &\quad p_{i1} = p_1^{\sharp} + \eta_i, \quad \eta_i \sim N(0, \sigma^2) \\
\text{Data perturbation:} &\quad y_{ij} = y_{ij}^{\sharp}(1 + \theta_{ij}), \quad \theta_{ij} \sim U(-\epsilon, +\epsilon)
\end{aligned}$$

A mixed-effects model for orange tree truncs

Example

We will apply our method to the following scenario:

$$\begin{aligned}
\text{Model function:} \quad & f(x;p) = \frac{p_1}{1 + p_2 e^{p_3 x}} \\
\text{Individual parameter:} \quad & p_{i1} = p_1^\sharp + \eta_i, \quad \eta_i \sim N(0, \sigma^2) \\
\text{Data perturbation:} \quad & y_{ij} = y_{ij}^\sharp(1 + \theta_{ij}), \quad \theta_{ij} \sim U(-\epsilon, +\epsilon)
\end{aligned}$$

For this specific example, we will use $N_p \in \{1, 2, 5, 50\}$ subjects, sampled at $N_d = 10$ data sites, evenly spaced within $[100, 1600]$.

UPPSALA UNIVERSITET

A mixed-effects model for orange tree truncs

Example

We will apply our method to the following scenario:

$$\text{Model function:} \qquad f(x;p) = \frac{p_1}{1+p_2 e^{p_3 x}}$$

$$\text{Individual parameter:} \qquad p_{i1} = p_1^\sharp + \eta_i, \quad \eta_i \sim N(0,\sigma^2)$$

$$\text{Data perturbation:} \qquad y_{ij} = y_{ij}^\sharp(1+\theta_{ij}), \quad \theta_{ij} \sim U(-\epsilon,+\epsilon)$$

For this specific example, we will use $N_p \in \{1,2,5,50\}$ subjects, sampled at $N_d = 10$ data sites, evenly spaced within $[100, 1600]$.

Target parameters:

$p^\sharp = (191.84, 8.153, -0.0029)$, $\sigma = 20$, $\epsilon \in \{0.01, 0.1, 0.2, 0.5\}$.

UPPSALA UNIVERSITET

A mixed-effects model for orange tree truncs

Example

We will apply our method to the following scenario:

$$
\begin{aligned}
\text{Model function:} &\qquad f(x;p) = \frac{p_1}{1+p_2 e^{p_3 x}} \\
\text{Individual parameter:} &\qquad p_{i1} = p_1^\sharp + \eta_i, \quad \eta_i \sim N(0,\sigma^2) \\
\text{Data perturbation:} &\qquad y_{ij} = y_{ij}^\sharp(1+\theta_{ij}), \quad \theta_{ij} \sim U(-\epsilon,+\epsilon)
\end{aligned}
$$

For this specific example, we will use $N_p \in \{1,2,5,50\}$ subjects, sampled at $N_d = 10$ data sites, evenly spaced within $[100,1600]$.

Target parameters:

$p^\sharp = (191.84, 8.153, -0.0029)$, $\sigma = 20$, $\epsilon \in \{0.01, 0.1, 0.2, 0.5\}$.

Search region:

$$\mathcal{P} = ([0,300],[0,9],[-1,0]).$$

UPPSALA UNIVERSITET

A mixed-effects model for orange tree truncs

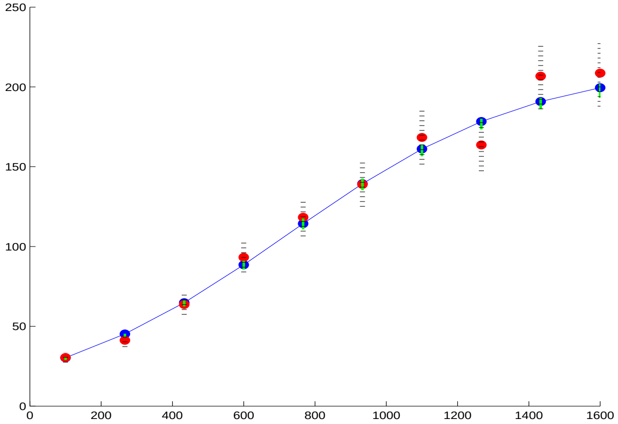


Figure: Data inflation and contraction for the example. The graph of the model function for one subject (blue line). The data points are marked with red dots. The inflated data sets are shown as striped bars, and the re-contracted data as green bars.

UPPSALA UNIVERSITET

A mixed-effects model for orange tree truncs

Numerical results

	$N_p = 1$	$N_p = 2$
$\epsilon = 0.01$	190.639 (– –) (0.010)	193.141 (19.6) (0.013)
$\epsilon = 0.1$	194.139 (– –) (0.092)	195.233 (21.1) (0.097)
$\epsilon = 0.2$	189.139 (– –) (0.190)	193.437 (20.3) (0.192)
$\epsilon = 0.5$	167.226 (– –) (0.604)	167.770 (26.6) (0.589)

	$N_p = 5$	$N_p = 50$
$\epsilon = 0.01$	191.675 (20.1) (0.014)	191.239 (20.1) (0.012)
$\epsilon = 0.1$	192.954 (21.4) (0.099)	198.428 (22.2) (0.110)
$\epsilon = 0.2$	191.773 (20.3) (0.203)	197.580 (23.6) (0.214)
$\epsilon = 0.5$	164.656 (23.9) (0.620)	174.318 (27.1) (0.618)

Table: The results of four experiments for the example, each using 100 trial runs with $p_1 = 191.184$, and $\sigma = 20.0$. For each pair (ϵ, N_p), we display the triple $\mu(p_1)$, $\mu(\sigma)$, and $\mu(\epsilon)$ – the average estimates of the distribution parameters for p_1, and the data error.

UPPSALA UNIVERSITET

A mixed-effects model for orange tree truncs

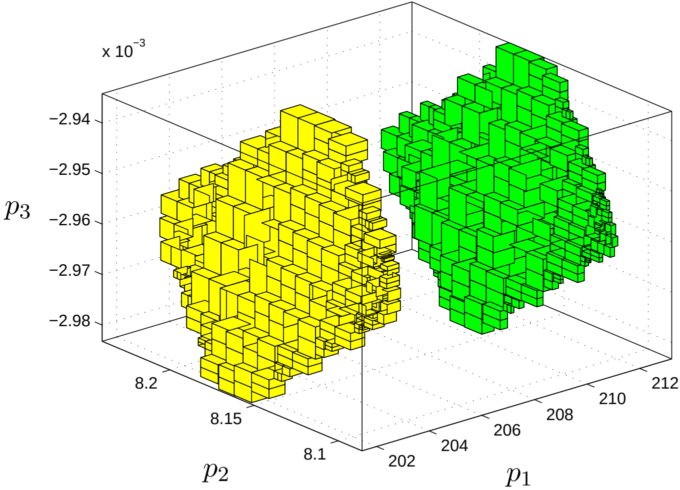


Figure: The set of consistent parameters for two subjects from the example.

Conclusions

UPPSALA
UNIVERSITET

Conclusions

- Standard numerics does not produce mathematics.

Conclusions

- Standard numerics does not produce mathematics.
- Set-valued mathematics enables validated numerics.

UPPSALA
UNIVERSITET

Conclusions

- Standard numerics does not produce mathematics.
- Set-valued mathematics enables validated numerics.
- Existence (uniqueness) comes from fixed point theorems.

Conclusions

- Standard numerics does not produce mathematics.
- Set-valued mathematics enables validated numerics.
- Existence (uniqueness) comes from fixed point theorems.
- Set-valued methods are suitable for inverse problems.

UPPSALA
UNIVERSITET

Conclusions

- Standard numerics does not produce mathematics.
- Set-valued mathematics enables validated numerics.
- Existence (uniqueness) comes from fixed point theorems.
- Set-valued methods are suitable for inverse problems.
- Parameter estimation is done via relaxation.

Conclusions

- Standard numerics does not produce mathematics.
- Set-valued mathematics enables validated numerics.
- Existence (uniqueness) comes from fixed point theorems.
- Set-valued methods are suitable for inverse problems.
- Parameter estimation is done via relaxation.
- The relaxed problem is solved via set inversion.

Further study...

Interval Computations Web Page

http://www.cs.utep.edu/interval-comp

Further study...

Interval Computations Web Page

http://www.cs.utep.edu/interval-comp

INTLAB – INTerval LABoratory

http://www.ti3.tu-harburg.de/~rump/intlab/

Further study...

Interval Computations Web Page

http://www.cs.utep.edu/interval-comp

INTLAB – INTerval LABoratory

http://www.ti3.tu-harburg.de/~rump/intlab/

CXSC – C eXtensions for Scientific Computation

http://www.xsc.de/

Further study...

Interval Computations Web Page

http://www.cs.utep.edu/interval-comp

INTLAB – INTerval LABoratory

http://www.ti3.tu-harburg.de/~rump/intlab/

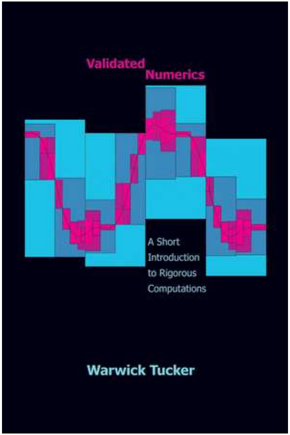
CXSC – C eXtensions for Scientific Computation

http://www.xsc.de/

CAPA – Computer–Aided Proofs in Analysis

http://www.math.uu.se/~warwick/CAPA/

A short message from your sponsors...



Validated Numerics:
A Short Introduction to Rigorous Computations

Warwick Tucker

Princeton University Press, 2011

ISBN: 9780691147819

152 pp.|6 x 9|41 illus.|12 tables.

USD 45.00/GBP 30.95

http://press.princeton.edu/titles/9488.html

UPPSALA UNIVERSITET