Introduction to Validated Computing

Verified Computations with Taylor Model Using Symbolic Algebra

Tomasz Strek
Institute of Applied Mechanics
Poznan University of Technology
Piotrowo 3, 60-965 Poznan, Poland
tstrek@sol.put.poznan.pl

1. Introduction

The idea of verified computation is based on the rigorous estimation of the influences of uncertainties on the calculation. Such uncertainties arise mainly from two reasons. First is associated with computational inaccuracies based on the finite accuracy of the computational environments. Second reason is the fact that there are uncertainties in the variables of the model to be analyzed.

The primary motivation for using interval arithmetic for verified computations is that the interval extension of a function provides bounds for the variation of the function. This comes from the fundamental property of interval arithmetic mentioned above. Different from floating-point computations, interval methods offer a simple mechanism to evaluate an enclosure of a function. The limitation of these methods is the overestimation mostly caused by the lack of information on functional dependency. It is well-known that standard interval arithmetic operations when used for the evaluation of functional ranges using a formal replacement of numeric variables by interval ones, gives a guaranteed enclosure for the solution, which is usually too large.

The Taylor model method models a function by a higher order polynomial, which keeps the majority of the functional dependency, and an interval, which contains the small remaining error. Although the approximation formula in the Taylor model has order $n+1$, it is difficult to realize in practice. The reason for this is that it requires the computation of the range of an n-th degree polynomial, a difficult problem in its own right. For this reason, for computing the range of a multivariate function over intervals, it is proposed to combine the Taylor model method with the Bernstein form of polynomial.

The above-mentioned method will be realized with Mathematica, one of the comprehensive computer systems for advanced mathematics. It is complete mathematical problem-solving environment that supports a wide variety of mathematical operations such as numerical analysis, symbolic algebra, and graphics.

The Taylor model polynomial and remaining error polynomial are easy to obtain using symbolic algebra. We do not have to use polynomial algebra. The Bernstein form of any polynomial we can get after some mathematical manipulations using symbolic algebra. Details of algorithms will be presented during presentation. The combined method is a very effective approximation method.

This new proposed method can be applied to compute validated solutions of initial value problems for ordinary differential equations. The numerical results of approximation of range of multivariate function by the Taylor model with the Bernstein polynomials can be compared with results of approximation with standard interval arithmetic.

Real interval arithmetic (RIA)

The most famous and very old example of interval enclosures is given by the method due to Archimedes. He considered inscribed polygons and circumscribed polygons of a circle with radius one. He obtained an increasing sequence of lower bounds and a decreasing sequence of upper bounds for the area of the corresponding circle. Thus stopping this process with a circumscribing and an inscribed polygon, each of n sides, he obtained an interval containing the number π. By choosing n large enough, an interval of arbitrary small width can be found in this way containing π.

Interval arithmetic is the arithmetic defined on sets of intervals, rather than sets of real numbers. A form of interval arithmetic perhaps first appeared in 1924 and 1931 in (Burkill, 1924; Young, 1931), than later in (Sunaga, 1958). Modern development of interval arithmetic began with Moore's dissertation (Moore, 1962, 1966). Since than, thousands of research articles (e.g. Alefeld, Claudio, 1998; Alefeld, Mayer, 2000) and books (e.g. Alefeld, Herzberger, 1974,1983) have been published on the subject. There is an interesting amount of software support for interval computations (e.g. Kaucher, Kulisch, Ullrich 1987).

Let's consider closed and bounded intervals

$$[a] := [\underline{a}, \overline{a}] = \{x \in \mathbf{R} \mid \underline{a} \le x \le \overline{a}\},$$

where $\mathbf{R}$ is the set of real numbers. The set of all such intervals is denoted by $\mathbf{IR}$. Real numbers a can be considered as special elements of $\mathbf{IR}$ with $[a] = [a, a]$.

If $\circ$ denotes one of the four operations $\circ \in \{+, -, \cdot, /\}$ for real numbers then the corresponding operations for two elements $[a]$ and $[b]$ from $\mathbf{IR}$ are defined by

$$[a] \circ [b] = \{a \circ b \mid (a \in [a]) \wedge (b \in [b])\}.$$

In the case of division $0 \notin [b]$ is assumed. Basic mathematical operations in the set of intervals can be defined by

$$[x] + [y] = [\underline{x} + \underline{y}; \overline{x} + \overline{y}],$$

$$[x] - [y] = [\underline{x} - \overline{y}; \overline{x} - \underline{y}],$$

$$[x] \cdot [y] = [\min(\underline{x}\underline{y}, \underline{x}\overline{y}, \overline{x}\underline{y}, \overline{x}\overline{y}), \max(\underline{x}\underline{y}, \underline{x}\overline{y}, \overline{x}\underline{y}, \overline{x}\overline{y})]$$

$$\frac{[x]}{[y]} = [x] \cdot \frac{1}{[y]}, \text{ where } \frac{1}{[y]} = [1/\overline{y}; 1/\underline{y}].$$

The distance of two intervals $[x]$ and $[y]$ is defined as the real number

$$q([x], [y]) := max\{|\underline{x} - \underline{y}|, |\overline{x} - \overline{y}|\}.$$

The absolute value of an interval $[x]$ is defined as the distance of $[x]$ from $[0] = [0;0]$

$$|[x]| := q([x], [0]) = max\{|\underline{x}|, |\overline{x}|\}.$$

The diameter (or width) of an interval $[x]$ is defined as

$$w([x]) = \overline{x} - \underline{x}.$$

The following rules for diameter hold

$$w([x] \pm [y]) = w([x]) + w([y]),$$

$$w(x[y]) = |x| w([y]) \text{ for } x \in \mathbf{R},$$

$$w([x][y]) \ge max(|[x]| w([y]), w([x])|[y]|).$$

We can also define minimum and maximum of an interval $[x]$ is defined as

$$\min([x]) = \inf([x]) = \underline{x},$$
$$\max([x]) = \sup([x]) = \overline{x}.$$

Basic properties of interval arithmetic

Besides of these four basic operations we consider unary operations in **IR**. Let r be a real continuos function defined on **R** or a subset of **R**. Then for $[a]$ contained in the domain of r we can define

$$r([a]) = \{r(a) \mid a \in [a]\} \in \mathbf{IR}.$$

Standard interval functions $r \in F = \{\sin, \cos, tan, arctan, \exp, \ln, abs, sqr, sqrt\}$ are defined in the same way.

We can define for a real-valued function $f(a,b,...,u,v)$ the so-called interval arithmetic evaluation of f by $\overline{f}([a],[b],...,[u],[v])$.

For interval arithmetic evaluations the following properties hold:

- If $[a] \subseteq [a_0], [b] \subseteq [b_0], ..., [v] \subseteq [v_0]$ then $\overline{f}([a],[b],...,[v]) \subseteq \overline{f}([a_0],[b_0],...,[v_0])$. This property is called inclusion monotonicity.
- If $a \in [a], b \in [b], ..., v \in [v]$ then $f(a,b,...,v) \in \overline{f}([a],[b],...,[v])$.

This property is a special case of the preceding one and is called inclusion property. It means that the interval arithmetic evaluation $\overline{f}([a],[b],...,[u],[v])$ always contains the range of the real function f defined

$$R(f;[a],[b],...,[u],[v]) = \{f(a,b,...,u,v) \mid a \in [a], b \in [b], ..., u \in [u], v \in [v]\}.$$

This is the property, which makes interval arithmetic so important in applications

$$R(f;[a],[b],...,[u],[v]) \subseteq \overline{f}([a],[b],...,[u],[v]).$$

Let consider the example of the interval arithmetic evaluation and the range of given real function f. Let

$$f(x) = \frac{x}{1-x}, \; x \neq 1$$

and

$$[x] = [2;3].$$

Then

$$R(f;[x]) = \left[-2; -\frac{3}{2}\right]$$

and

$$\overline{f}([x]) = \frac{[x]}{1-[x]} = \frac{[2;3]}{1-[2;3]} = [-3;-1]$$

so we have

$$R(f;[x]) \subseteq \overline{f}([x]).$$

For $x \neq 0$ we can rewrite $f(x)$ as

$$f(x) = \frac{x}{1-x} = \frac{1}{\frac{1}{x} - 1} = g(x).$$

For

$$[x] = [2;3]$$

we obtain

$$\overline{g}([x]) = \frac{1}{\frac{1}{[x]} - 1} = \frac{1}{\frac{1}{[2;3]} - 1} = \left[-2; -\frac{3}{2}\right] = R(f;[x]).$$

This example shows that the overestimation of the range of a given function by the interval arithmetic expression is strongly dependent on the arithmetic expression of the given function. The reason for this is based on the fact that interval arithmetic does not follow the same rules as the arithmetic for real numbers.

For $[x],[y],[z] \in \mathbf{IR}$ we have

$$[x]([y]+[z]) \subseteq [x][y]+[x][z].$$

This property is called subdistributivity. It is worth noticing that equality holds in some important particular cases. For instance, if $x \in \mathbf{R}$ we have

$$x([y]+[z]) = x[y]+x[z].$$

We have also for $[x] \in \mathbf{IR}$: $[x]-[x] \neq 0$ and $[x]/[x] \neq 1$ for $0 \notin [x]$ if $[x]$ is a proper interval.

We have seen that the overestimation of the range of real function by the interval arithmetic evaluation is dependent on the arithmetic expression, which is used for the interval arithmetic operations. Moore has shown that under reasonable assumptions the following inequality holds for the distance between $R(f;[x])$ and $\overline{f}([x])$

$$q\left(R(f;[x]), \overline{f}([x])\right) \leq \gamma w([x]),\ \gamma \geq 0$$

where $[x] \subseteq [x^0]$ and $[x^0]$ is some fixed interval. This inequality means that the overestimation of $R(f;[x])$ by $\overline{f}([x])$ goes linearly to zero with the diameter of $[x], w([x])$. This estimation holds analogously for the interval arithmetic evaluation of functions of several variables.

Floating-point arithmetic

The arithmetic performed in a machine (computer) involves numbers with only a finite number of digits (Kulisch, Miranker, 1981). It results that many calculations are performed with approximate representations of the actual numbers. In typical computer, that support IEEE standard arithmetic (Stevenson, 1985), such as most PC's and workstations, only a relatively small subset of the real numbers system is used for the representation of all real numbers. This subset contains only rational numbers, booth positive and negative, and stores a fractional part, called the mantisa, together with an exponential part, called the characteristic.

The result of any operation on fixed-point number is typically stored in a register that is no longer than the original format of number. When the result is put back into the original format, the extra bits must be disposed of. That is, the result must be rounded. Rounding involves going from high precision to lower precision and produces quantization errors and computational noise. There are four rounding modes available in most of programming languages: round toward zero, round towards nearest, round toward ceilling (positive infinity) and round toward floor (negative infinity).

- Round toward zero. The computationally simplest rounding mode is to drop all digits beyond the number required. This mode is referred as rounding toward zero. It results in a

number whose magnitude is always less than or equal to the more precise original value. That is, all positive numbers are rounded to smaller positive number, while all negative numbers are rounded to smaller negative numbers.

- Round toward nearest. When rounding toward nearest, the number is rounded to the nearest representable value. This mode has the smallest errors associated with it and these errors are symmetric. As a result, rounding towards nearest is the most useful approach for most applications. This rounding mode is default in most languages.
- Round toward ceilling (positive infinity). When rounding toward ceilling, both positive and negative numbers are rounded toward positive infinity.
- Round toward floor (negative infinity). When rounding toward floor, both positive and negative numbers are rounded toward negative infinity.

Floating-point interval arithmetic (FPIA)

The power of the interval arithmetic lay in implementation of interval arithmetic on computers (floating-point arithmetic). In particular, outwardly rounded interval arithmetic allows rigorous enclosures for the ranges of operations and functions. This makes a qualitative difference in scientific computations, since the results are now intervals in which the exact result must lie. It also enables use of computations for automated theorem proving.

All floating-point arithmetic operations during interval arithmetic computations on computer have to been made with correct rounding mode. All round-off errors associated with floating-point arithmetic have to be enclosed by the machine interval arithmetic in the result of any computation to ensure validated results. For example, take $[x]+[y]=\left[\underline{x}+\underline{y};\overline{x}+\overline{y}\right]$. If $\underline{x}+\underline{y}$ is rounded down after computation and $\overline{x}+\overline{y}$ is rounded up after computation, then the resulting interval $[z]=[\underline{z};\overline{z}]$ that is represented in the machine must contain the exact range of $x+y$ for $x\in[x]$ and $y\in[y]$.

Good enclosures for the ranges of standard function can be computed. The computer interval arithmetic can be carried out for virtually any expression that can be evaluated with floating point arithmetic. However, since interval arithmetic only subdistributive, expressions that are equivalent in the real arithmetic differ in interval arithmetic. In particular, computations should be arranged so that overestimation of ranges is minimized. The fact that naively arranged computations do not always give adequately narrow bounds on the range has been source of controversy, but there have been advances in recent years in the astute use of interval arithmetic. The sets of theory and tools are too large to be fully described in this paper. Advanced details and explanations of interval arithmetic can be found in the references.

Examples: solution of a set of linear equations

In this subsection we present only example of solution of a set of linear equations with standard floating point arithmetic with difference available rounding modes and computer interval arithmetic.

Let's consider the problem of solution of real linear system

$$\mathbf{Ax} = \mathbf{b},$$

where

$$A_{ij} = \frac{1}{i+j-1} \text{ and } b_i = \sum_{j=1}^{n} j \cdot A_{ij} ,$$

with Gaussian elimination algorithm, where n is the number of equations. The correct solutions for this system of equations are successive natural numbers from 1 to n.

Table 1 and Table 2 present results of solutions of considered real linear system (with $n = 10$) with rounding to negative and positive infinity, and with rounding to nearest and zero, respectively. All real-valued calculations in solving of this problem were made with MATLAB (MATrix LABoratory).

Listing 1
MATLAB M-file with implementation of algorithm for solution of considered real linear system with selected rounding mode available in MATLAB

```
clear all
format long e
n=10;
%rounding to nearest: setround(0)
%rounding to negative infinity: setround(-1)
%rounding to positive infinity: setround(1)
%rounding to zero: setround(2)
setround(2)
for i=1:n
  for j=1:n
    A(i,j)=1/(i+j-1);
  end;
  b(i)=sum(A(i,:).*[1:n]);
end;
x=b/A; x1=x'
setround(0)
```

Table 1
Results of solutions of considered real linear system with rounding to negative and positive infinity

Nr	Rounding to negative infinity	Rounding to positive infinity
1	1.000000005310460e+000	1.000000005053460e+000
2	1.999999540862342e+000	1.999999544236507e+000
3	3.000009793624234e+000	3.000010006968410e+000
4	3.999910813494155e+000	3.999906941462206e+000
5	5.000426145579732e+000	5.000451732848140e+000
6	5.998826559116256e+000	5.998740682937394e+000
7	7.001928291899554e+000	7.002089885060843e+000
8	7.998133809985666e+000	7.997961108609275e+000
9	9.000981050565368e+000	9.001079010113854e+000
10	9.999783985701424e+000	9.999761078165776e+000

Table 2
Results of solutions of considered real linear system with rounding to nearest and zero

Nr	Rounding to nearest	Rounding to zero
1	1.000000002292007e+000	1.000000005310462e+000
2	1.999999802158318e+000	1.999999540862339e+000

3	3.000004206701762e+000	3.000009793624240e+000
4	3.999961835015409e+000	3.999910813494132e+000
5	5.000181647215761e+000	5.000426145579755e+000
6	5.999501728244047e+000	5.998826559116256e+000
7	7.000815802077731e+000	7.001928291899554e+000
8	7.999213197295525e+000	7.998133809985666e+000
9	9.000412283324703e+000	9.000981050565368e+000
10	9.999909494163667e+000	9.999783985701424e+000

After a series of arithmetic operations we may not know the exact answer because limitations in the representation of numbers. Even if every operation in the series is performed twice, once rounding to negative infinity and once rounding to positive infinity, we can not be sure that correct answer belongs to obtained interval.

Table 3 presents results of solutions of considered real linear system with computer interval arithmetic. All calculations were made with INTLAB – MATLB library for interval arithmetic routines (Rump, 1999). All INTLAB code is written in MATLAB for best portability. Major objective of INTLAB is speed and ease of use. For more details see http://www.ti3.tu-harburg.de/rump/intlab/.

Table 3
Results of solutions of considered real linear system with CIA

Nr	Computer interval arithmetic
1	[9.999999589140342e-001, 1.000000052175456e+000]
2	[1.999995501688966e+000, 2.000003536262676e+000]
3	[2.999924843990240e+000, 3.000095699824235e+000]
4	[3.999130622482604e+000, 4.000682310243689e+000]
5	[4.996748157080561e+000, 5.004145082874566e+000]
6	[5.988608255622893e+000, 6.008934491748045e+000]
7	[6.985346197430993e+000, 7.018687219372125e+000]
8	[7.981944658466329e+000, 8.014156648679050e+000]
9	[8.992568970889897e+000, 9.009478204091966e+000]
10	[9.997915900946573e+000, 1.000163388243674e+001]

It is well noticed that correct mathematical results of solution belong to obtained interval in each case.

Interval overestimation and functional dependency

The primary motivation for using interval analysis in almost all applications is that the interval extension of a function provides bounds for the variation of the function. This comes from the fundamental property of interval arithmetic mentioned above.

The conventional simple interval methods do not carry the information on functional dependency. Let's consider simple computation of powers. Note that the computation of powers can be done in several ways. To see this, consider $[x]=[-1,2]$. Computing $[x]\cdot[x]$ interval arithmetic gives $[-2,4]$, but the exact range for $[x]^2$ is $[0,4]$.

The dependency problem in interval arithmetic is typically caused by cancellation effects. For example, if $[x]=[a,b]$, then $[x]-[x]$, if not recognised to represent the same

number, is computed as $[a,b]-[a,b]=[a,b]+[-b,-a]=[a-b,b-a]$, resulting in a width that is not zero, but twice as large as before. Such a situation can not be avoided in conventional interval methods in practice if the cancelling terms appear as a result of preceding complicated science computations.

As we can notice a fundamental problem in interval methods is computing the ranges of values of real functions (Cornelius, Lohner, 1984). Interval methods provide rigorous enclosures of functions, however the limitation of the methods is the overestimation mostly caused by the lack of information on functional dependency.

The one of tool to control overestimation was proposed by Moore (Moore, 1966, Moore, 1979). Author proposed the tool of subdivision to compute the range of function to a desired accuracy. Moore actually suggested two different strategies for subdivision called uniform subdivision and adaptive subdivision. Both these strategies are well known and are extensively used in various applications of interval analysis. Nataraj and Sheela in the paper (Nataraj, Sheela, 2002) presented a new subdivision strategy in interval analysis for computing the ranges of functions. The authors showed through several real-world examples that the proposed in the paper parallel-adaptive strategy is more efficient than the widely used uniform and adaptive subdivision strategies. In this work, parallel means simultaneous processing of all boxes present in a subdivision partition. Unfortunately, in case of multivariate functions, the computational expense by simple interval methods increases astronomically, especially with subdivision strategy.

To efficient control overestimation and functional dependency a brand new method, the Taylor model approximation method, has been developed and implemented in the code COSY Infinity (Berz, 1997; Makino, 1998). The method models a function f in domain D by a high order multivariate Taylor polynomial p and the remainder error interval r (Makino, Berz, 1999; Makino, Berz, 2001).

Additionally operations with Taylor models were presented. Details of addition, multiplication, divisions, elementary functions of Taylor models and calculation of Taylor models of complicated functions one can find in references (e.g. Berz, Hoffstätter, 1998).

Taylor model approximation method

We can assume that the real function $f(x_1,x_2,...,x_s)$, $f:[x_{11},x_{12}]\times...\times[x_{s1},x_{s2}]\to\mathbf{R}^s$ is $n+1$ times differentiable on the s-dimensional interval $[x_{11},x_{12}]\times...\times[x_{s1},x_{s2}]$ and that $(x_1,x_2,...,x_s)\in[x_{11},x_{12}]\times...\times[x_{s1},x_{s2}]$. The Taylor expansion of f is then

$$f(x_1,x_2,...,x_s)=p(x_1,x_2,...,x_s)+r(\xi_1,\xi_2,...,\xi_s)$$

where

$$p(x_1,...,x_s)=\sum_{i_1+...+i_s=0}^{n}a_{i_1...i_s}(x_{10},...,x_{s0})(x_1-x_{10})^{i_1}....(x_s-x_{s0})^{i_{1s}}$$

$$(x_{10},...,x_{s0})\in[x_{11},x_{12}]\times...\times[x_{s1},x_{s2}]$$

$$a_{i_1...i_s}(z_1,...,z_s)=\frac{1}{i_1!...i_s!}\frac{\partial f^{(i_1+...+i_s)}(z_1,...,z_s)}{\partial x_1^{i_1}...\partial x_s^{i_s}}$$

$$r(\xi_1,...,\xi_s)=\sum_{i_1+...+i_s=n+1}a_{i_1...i_s}(\xi_1,...,\xi_s)(x_1-x_{10})^{i_1}....(x_s-x_{s0})^{i_s}$$

$$(\xi_1,...,\xi_s)\in[x_{11},x_{12}]\times...\times[x_{s1},x_{s2}].$$

For $[x_1]\times...\times[x_s]\subseteq[x_{11},x_{12}]\times...\times[x_{s1},x_{s2}]$, the Taylor form can be expressed as

$$\overline{f}([x_1],...,[x_s]) = \overline{p}([x_1],...,[x_s]) + \overline{r}([x_1],...,[x_s]),$$

where

$$\overline{p}([x_1],...,[x_s]) \text{ is the range of } p \text{ over } [x_1]\times...\times[x_s]$$

$$\overline{r}([x_1],...,[x_s]) = \sum_{i_1+...+i_s=n+1} a_{i_1...i_s}([x_1],...,[x_s])([x_1]-x_{10})^{i_1}....([x_s]-x_{s0})^{i_{1s}} \ .$$

The above approximation formula has order $n+1$. Details and proof of this theorem one can find in (Lin, Rokne, 1996). Unfortunately, it is difficult to realize in practice, because it requires the computation of the range of an *n-th* degree polynomial, a difficult problem in its own right. For this reason the Taylor model approximation is combined with the Bernstein form of polynomial (Lin, Rokne, 1995; Lin, Rokne, 1996).

The multivariate Bernstein form

Let $p(x_1,...,x_s)$ be a polynomial in s real variables with the maximum degree $n_1+...+n_s$, that is,

$$p(x_1,...,x_s) = \sum_{i_1=0}^{n_1}...\sum_{i_s=0}^{n_s} a_{i_1...i_s} x^{i_1}...x^{i_s} \ ,$$

where $(x_1,x_2,...,x_s)\in[x_{11},x_{12}]\times...\times[x_{s1},x_{s2}]$. We can also assume that $[x_{11},x_{12}]=...=[x_{s1},x_{s2}]=[0,1]$. Any finite interval $[a,b]$ can be mapped to $[0,1]$ by a linear transformation $\frac{x-a}{b-a}$: $\frac{[a,b]-a}{b-a}=[0,1]$ (or any $[0,1]$ to $[a,b]$ by $(b-a)x+a$: $(b-a)[0,1]+a=[a,b]$).

We introduce the Bernstein basis function

$$B_j^k(x) = \binom{k}{j} x^j (1-x)^{k-j}, \ x\in[0,1].$$

It is easily shown that

$$B_j^k(x)\ge 0\,,\ \sum_{j-0}^{k} B_j^k(x) = 1,\ x\in[0,1],$$

$$x^i = \sum_{j=0}^{k} \frac{\binom{j}{i}}{\binom{k}{i}} B_j^k(x),\ x\in[0,1],\ i=0,1,...,n\le k\ .$$

We can write polynomial $p(x_1,...,x_s)$ as

$$p(x_1,...,x_s) = \sum_{i_1=0}^{n_1}...\sum_{i_s=0}^{n_s} a_{i_1...i_s} \sum_{j_1=0}^{k_s} \frac{\binom{j_1}{i_1}}{\binom{k_1}{i_1}} B_{j_1}^{k_1}(x_1)...\sum_{j_s=0}^{k_s} \frac{\binom{j_s}{i_s}}{\binom{k_s}{i_s}} B_{j_s}^{k_s}(x_s).$$

After some mathematical manipulations we can write polynomial in the Bernstein form

$$p(x_1,...,x_s) = \sum_{j_1=0}^{k_1}...\sum_{j_s=0}^{k_s} b_{j_1...j_s} B_{j_1}^{k_1}(x_1)...B_{j_s}^{k_s}(x_s)$$

where

$$b_{j_1\dots j_s} = \sum_{i_1=0}^{\min(j_1,n_1)} \dots \sum_{i_s=0}^{\min(j_s,n_s)} a_{i_1\dots i_s} \frac{\binom{j_1}{i_1}}{\binom{k_1}{i_1}} \dots \frac{\binom{j_s}{i_s}}{\binom{k_s}{i_s}}$$

with the assumption that $k_1 \ge n_1, \dots, k_s \ge n_s$.

For $b_{j_1\dots j_s}$, where $j_1 = 0,\dots,k_1; \dots; j_s = 0,\dots,k_s$, we have that (Lin, Rokne, 1996)

$$\left| b_{j_1,\dots,j_s} - p\left(\frac{j_1}{k_1},\dots,\frac{j_s}{k_s}\right) \right| = O\left(\frac{1}{k_1} + \dots + \frac{1}{k_s}\right).$$

It means that the quantities $b_{j_1\dots j_s}$ can be used to construct a Bernstein form for multivariate polynomials for approximating the range $\overline{p}([0,1],\dots,[0,1])$ of the polynomial $p(x_1,\dots,x_s)$. It can be prove that (Lin, Rokne, 1995; Lin, Rokne, 1996).

$$\overline{p}([0,1],\dots,[0,1]) \subseteq B_p^{k_1\dots k_s}([0,1],\dots,[0,1]),$$

where $B_p^{k_1\dots k_s}([0,1],\dots,[0,1]) = \left[\min_{j_1\dots j_s} b_{j_1\dots j_s}, \max_{j_1\dots j_s} b_{j_1\dots j_s}\right]$.

X. Verified integration

One of the important fields in validated methods is solving initial value problems (IVP) of ordinary differential equations (ODEs). Interval techniques for solving IVP of ODEs provide enclosures for errors in the initial values, mathematical truncation, and roundoff errors. This way, for each time point intervals are produced that contain the actual solution of the given problem.

In the case of solving a system of multidimensional ODEs, there arises the so-called wrapping effect, which is caused by the inflation of the size of the geometric set enclosing the validated solution set at each time step. The wrapping effect is a particular form of the dependency problem based on the connection of current dynamical values on initial condition, which often is more dramatic than the other sources of overestimation. The typical problem of overestimation in interval arithmetic is mostly caused by the lack of information of functional dependency.

The wrapping effect is due to the fact that the image of an interval vector under a map is not an interval vector and there is thus overestimation in enclosing the image with an interval vector. This effect can be ameliorated with changes of variables or other techniques and methods (Nedialkov, Jackson, Corlis, 1999).

We consider validated numerical methods for the solution of the autonomous IVP

$$\frac{\partial y(t)}{\partial t} = f(y),\ y(t_0) = y_0$$

where $t \in [t_0, t_1]$ for some $t_1 > t_0$. Here $t_0, t_1 \in \mathbf{R}$, $f \in C^{k-1}(D)$, $D \subseteq \mathbf{R}$ is an open set, $f : D \to \mathbf{R}$ and $y_0 \in D$. Our goal is to compute, at given points $\{t_j\}$ in $[t_0, t_1]$, an interval vector $[y_j]$ that is guaranteed to contain the true solution $y(t_j)$ of given IVP.

Standard numerical methods for IVPs for ODEs attempt to compute an approximate solution that satisfies a user specified tolerance. These methods are usually robust and reliable

for most applications. However, it is easy to find examples for which they return very inaccurate results.

If the validated method (also called an interval method) for IVPs for ODEs returns successfully, it not only produces a guaranteed error bound for the true solution, but also verifies that a unique solution to the problem exists.

X.1. The standard Taylor method

Assuming that the solution $u(t)$ to the initial value problem

$$\frac{\partial u(t)}{\partial t} = f(t,u(t)),\ t \in [t_j, t_{j+1}],\ u(t_j) = u_j$$

has (n+1) continuous derivatives and expanding that solution, $u(t)$, in terms of its nth degree Taylor polynomial about t_j, we obtain

$$u(t_{j+1}) = u(t_j) + \sum_{i=1}^{n} u^{[i]}(t_j)\frac{h^i}{i!} + u^{[n+1]}(\xi_j)\frac{h^{n+1}}{(n+1)!}$$

for some $\xi_j \in [t_j, t_{j+1}]$ and $h = t_{j+1} - t_j$.

Successive differentiation of the solution, $u(t)$, gives in general

$$u^{[k]}(t) = f^{[k-1]}(t,u(t)),\ k = 1,2,...,n+1.$$

Substituting these results into expanding of $u(t)$ gives

$$u(t_{j+1}) = u(t_j) + \sum_{i=1}^{n} f^{[i-1]}(t_j, u(t_j))\frac{h^i}{i!} + f^{[n]}(\xi_j, u(\xi_j))\frac{h^{n+1}}{(n+1)!}.$$

The difference method corresponding to above equation is obtained by neglecting the remainder term involving ξ_j. This method is called the standard Taylor method of order n

$$w_0 = u(t_0),$$

$$w_{j+1} = w_j + hT_n(t_j, w_j) \text{ for each } j = 0,1,2,...,$$

where $T_n(t_j, w_j) = f(t_j, u(t_j)) + \sum_{i=2}^{n} f^{[i-1]}(t_j, u(t_j))\frac{h^{i-1}}{i!}$. The methods which takes into consideration also the remainder term can be used to produce the verified integration process (Berz, Makino, 1998).

In the paper (Lara, Elipe, Palacios, 1999) problem of the integration of differential equations by Taylor series was presented. This method was applied in two applications the Henon-Heiles and the van der Pol's oscillator. Numerical comparisons between the recurrent power series method and well-known Runge-Kutta method were made.

X.2. Verified integration with Taylor Models

One of the important fields in validated methods is solving initial value problems in ordinary differential equations. Verified integration of ODEs with Taylor models is based on local modeling with high-order Taylor polynomials with remainder bound. The use of such Taylor models of order n allows convenient automated verified inclusion of functional dependencies (Berz, Makino, 1998; Berz, Makino, Hoefkens, 2001).

In this section, a technique for the recursive generation of Taylor coefficients of arbitrary order for functions defined by ordinary differential systems is presented. The method

is applicable to a wide class of problems and can be programmed for a computer in such the way that only the differential system with initial values need to be given. We consider only autonomous systems, but each time dependent system of ODEs can be converted into an autonomous system.

Denote the ith Taylor coefficient of $u(t)$ evaluated at some point t_j by

$$\left(u_j\right)_i = \frac{u^{(i)}\left(t_j\right)}{i!}$$

where $u^{(i)}(t)$ is the ith derivative of $u(t)$. Let $\left(u_j\right)_i$ and $\left(v_j\right)_i$ be the i-th Taylor coefficients of $u(t)$ and $v(t)$ at t_j. It can be shown that

$$\left(u_j \pm v_j\right)_i = \left(u_j\right)_i \pm \left(v_j\right)_i$$

$$\left(u_j v_j\right)_i = \sum_{r=0}^{i} \left(u_j\right)_r \left(v_j\right)_{i-r}$$

$$\left(\frac{u_j}{v_j}\right)_i = \frac{1}{v_j}\left\{\left(u_j\right)_i - \sum_{r=1}^{i}\left(v_j\right)_r \left(\frac{u_j}{v_j}\right)_{i-r}\right\}.$$

Similar formulas can be derived for the generation of Taylor coefficients for the standard functions (Rihm, 1994; Makino, 1998).

X.2.1. Solution of first order initial value problem

Consider the autonomous differential equation

$$\frac{\partial y(t)}{\partial t} = f(y),$$

with initial condition $y\left(t_j\right) = y_j$.

We introduce the sequence of functions

$$f^{[1]} = f,$$

$$f^{[i]} = \frac{1}{i}\frac{\partial f^{[i-1]}}{\partial y} f \quad \text{for } i \geq 2.$$

The Taylor coefficient of $y(t)$ at t_j satisfy

$$\left(y_j\right)_0 = y_j,$$

$$\left(y_j\right)_1 = f^{[1]}\left(y_j\right) = f\left(y_j\right),$$

$$\left(y_j\right)_i = f^{[i]}\left(y_j\right) = \left(\frac{1}{i}\frac{\partial f^{[i-1]}}{\partial y} f\right)\left(y_j\right) \text{ for } i \geq 2.$$

We can write solution of IVP of ODE $y(t)$ at t_j in the form

$$y(t) = \left(y_j\right)_0 + \sum_{i=1}^{k-1}\left(y_j\right)_i (t-t_j)^i + \left(y_\xi\right)_k (t-t_j)^k,$$

where $\left(y_\xi\right)_k = f^{[k]}\left(y_\xi\right) = \left(\frac{1}{i}\frac{\partial f^{[i-1]}}{\partial y} f\right)\left(y_\xi\right)$ and $y_\xi = y\left(t_\xi\right),\ t_\xi \in [t_j, t]$.

After introducing $t = t_j + h$ we obtain

$$y(t_j + h) = (y_j)_0 + \sum_{i=1}^{k-1} (y_j)_i h^i + (y_\xi)_k h^k .$$

Let $y(t_j) = y_j \in [y_j]$. If we have a procedure for computing the point Taylor coefficient of $y(t)$ and perform the computations in interval arithmetic with $[y_j]$ instead of y_j, we obtain a procedure for computing the interval Taylor coefficients of $y(t)$. If we denote the i-th interval coefficient of $y(t)$ at t_j by $[y_j]_i$, then we have

$$[y_j]_0 = [y_j],$$
$$[y_j]_1 = f^{[1]}([y_j]) = f([y_j]),$$
$$[y_j]_i = f^{[i]}([y_j]) = \left(\frac{1}{i}\frac{\partial f^{[i-1]}}{\partial y} f\right)([y_j]) \text{ for } i \geq 2 .$$

We can write solution of IVP of ODE $y(t)$ at t_j in the form

$$y(t) = [y_j]_0 + \sum_{i=1}^{k-1} [y_j]_i (t - t_j)^i + [y_j]_k ([t_j, t] - t_j)^k .$$

X.2.2. Solution of system of first order initial value problems

Let us consider now case of system of the autonomous differential equations

$$\frac{\partial \mathbf{y}(t)}{\partial t} = \mathbf{f}(\mathbf{y}),$$

with initial condition $\mathbf{y}(t_j) = \mathbf{y}_j = [y_{1j}, y_{2j}, ..., y_{nj}]$, where $\mathbf{f} = [f_1, f_2, ..., f_n]^T$ and $\mathbf{y} = [y_1, y_2, ..., y_n]$. This system of ODEs we can write in the form

$$\frac{\partial y_i}{\partial t} = f_i(y_1, y_2, ..., y_n),\ y_i(t_j) = y_{ij},\ i = 1,2,...,n .$$

We introduce the sequence of functions

$$\mathbf{f}^{[1]} = \mathbf{f}$$
$$\mathbf{f}^{[i]} = \frac{1}{i} J(\mathbf{f}^{[i-1]}, \mathbf{y}) \cdot \mathbf{f} \text{ , for } i \geq 2$$

where

$$J(\mathbf{g}, \mathbf{y}) = \begin{bmatrix} \frac{\partial g_1}{\partial y_1} & \frac{\partial g_1}{\partial y_2} & \cdots & \frac{\partial g_1}{\partial y_n} \\ \frac{\partial g_2}{\partial y_1} & \frac{\partial g_2}{\partial y_2} & \cdots & \frac{\partial g_2}{\partial y_n} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial g_n}{\partial y_1} & \frac{\partial g_n}{\partial y_2} & \cdots & \frac{\partial g_n}{\partial y_n} \end{bmatrix}$$

is the Jacobian matrix and $\mathbf{g} = [g_1, g_2, ..., g_n]^T$ and $\mathbf{y} = [y_1, y_2, ..., y_n]$..

The Taylor coefficient of $\mathbf{y}(t)$ at t_j satisfy

$$(\mathbf{y}_j)_0 = \mathbf{y}_j ,$$
$$(\mathbf{y}_j)_1 = \mathbf{f}^{[1]}(\mathbf{y}_j) = \mathbf{f}(\mathbf{y}_j),$$
$$(\mathbf{y}_j)_i = \mathbf{f}^{[i]}(\mathbf{y}_j) = J(\mathbf{f}^{[i-1]}, \mathbf{y})(\mathbf{y}_j) \text{ for } i \geq 2 .$$

We can write solution of IVP of ODE $\mathbf{y}(t)$ at t_j in the form

$$\mathbf{y}(t) = (\mathbf{y}_j)_0 + \sum_{i=1}^{k-1} (\mathbf{y}_j)_i (t - t_j)^i + (\mathbf{y}_\xi)_k (t - t_j)^k .$$

Let $\mathbf{y}(t_j) = \mathbf{y}_j \in [\mathbf{y}_j]$, where $[\mathbf{y}_j] = [[y_1], [y_2], ..., [y_n]]$. Performing the computations in interval arithmetic, we obtain a procedure for computing the interval Taylor coefficients of $\mathbf{y}(t)$. If we denote the i-th interval coefficient of $\mathbf{y}(t)$ at t_j by $[\mathbf{y}_j]_i$, then we have

$$[\mathbf{y}_j]_0 = [\mathbf{y}_j],$$
$$[\mathbf{y}_j]_1 = \mathbf{f}^{[1]}([\mathbf{y}_j]) = \mathbf{f}([\mathbf{y}_j]),$$
$$[\mathbf{y}_j]_i = \mathbf{f}^{[i]}([\mathbf{y}_j]) = J(\mathbf{f}^{[i-1]}, \mathbf{y})([\mathbf{y}_j]) \text{ for } i \geq 2 .$$

We can write solution of IVP of ODEs $\mathbf{y}(t)$ at t_j in the form

$$\mathbf{y}(t) = [\mathbf{y}_j]_0 + \sum_{i=1}^{k-1} [\mathbf{y}_j]_i (t - t_j)^i + [\mathbf{y}_j]_k ([t_j, t] - t_j)^k .$$

X+1. Symbolic calculation with Taylor model

References – Interval arithmetic

- Alefeld, G., Claudio, D.: The basic properties of interval arithmetic, its software realization and some applications, *Computer and Structures,* 67, 3-8, 1998.
- Alfeld, G., Herzberger, J.: *Introduction to Interval Computations,* Wissenschaftsverlag, Mannheim, Wien, Zürich, 1974.
- Alfeld, G., Herzberger, J.: *Introduction to Interval Computations,* Academic Press, New York, 1983.
- Alfeld, G., Mayer, G., Interval analysis: theory and applications, *Journal of Computational and Applied Mathematics,* 121, 421-464, 2000.
- Burkill, J.C.: Function of intervals. *Proccedings of the London Mathematical Society,* 22, 375-446, 1924.
- Cornelius, H., Lohner, R. Computing the range of values of real functions with accuracy higher than second order, *Computing,* 33, 331-347, 1984.
- Kaucher, E., Kulisch, U., Ullrich, Ch., editors. *Computer Arithmetic – Scientific Computation and Programming Languages,* Stuttgar, 1987.
- Kulisch, U.W., Miranker, W.L. *Computer Arithmetic in Theory and Practice,* Academic Press, New York, 1981.
- Moore, R.E.: *Interval Arithmetic and Automatic Error Analysis in Digital Computing.* PhD thesis, Stanford University, October 1962.
- Moore, R.E.: *Interval Analysis,* Prentice Hall, Englewood Cliffs, 1966.
- Moore, R.E.: *Methods and applications of interval analysis,* SIAM, Philadelphia, 1979.
- Rump, S.M.: *INTLAB –INTerval LABoratory,* www.ti3.tu-harburg.de/rump/intlab, 1999.
- Stevenson, D., chairman: Floating-Point Working Group, Microprocessor Standards Subcommittee. *IEEE standard for binary floating point arithmetic* (IEEE/ANSI 754-1985). Technical report, IEEE, 1985

- Sunaga, T.: Theory of an interval algebra and its application to numerical analysis, RAAG Memories 2, 29-46, 1958.
- Young, R.C.: The algebra of many-valued quantities. *Math. Ann.*, 104, 260-290, 1931.

References – Taylor method

- Berz, M.COSY INFINITY Version 8. Reference manual. Technical Report MSCUCL-1088, National Superconducting Cyclotron Laboratory, Michigan State University, East Lansing, Michigan, USA, 1997.
- Berz, M., Hoffstätter, G. Computation and Application of Taylor Polynomials with Interval remainder Bounds, *Reliable Computing,* 4, 83-97, 1998.
- Lara, M., Elipe, A., Palacios, M., Automatic programming of recurrent power series, *Mathematics and Computers in Simulations,* 49, 351-362, 1999.
- Lin, Q., Rokne, J. G. Methods for bounding the range of a polynomial, *Journal of Computational and Applied Mathematics,* 58, 193-199, 1995.
- Lin, Q., Rokne, J. G. Interval Approximation of Higher Order to the Ranges of Functions, *Computers and Math. Applic.,* Vol. 31, No. 7, 101-109, 1996.
- Makino, K., Berz, M. Higher Order Verified Inclusions of Multidimensional Systems by Taylor Models, *Nonlinear Analysis,* 47, 3503-3514, 2001.
- Makino, K., Berz, M. Efficient Control of the Dependency Problem based on Taylor Model Methods, *Reliable Computing,* 5, 3-12, 1999.
- Makino, K. Rigorous Analysis of Nonlinear Motion in Particle Accelerators, PhD thesis, Technical report MSUCL-1093, Michigan State University, East Lansing, Michigan, USA, 1998
- Nataraj, P. S. V., Sheela S. M. A New Subdivision Strategy for range Computations, *Reliable Computing,* 8, 83-92, 2002.
- Neher, M. Validated Bounds for Taylor Coefficient of Analytic Functions, *Reliable Computing,* 7, 307-319, 2001.

References – ODE -Taylor model

- Alfeld, G., Mayer, G. Interval analysis: theory and applications, *Journal of Computational and Applied Mathematics,* 121, 421-464, 2000.
- Berz, M., Makino, K. Verified Integration of ODEs and Flows using Differential Algebraic Methods on Higher-Order taylor Models, *Reliable Computing,* 4, 361-369, 1998.
- Berz, M., Makino, K. Hoefkens, J. Verified Integration of Dynamics in the Solar Systems, *Nonlinear Analysis,* 47, 179-190, 2001.
- Nedialkov, N.S., Jackson, K.R., Corlis, G.F. Validated solutions of initial value problems for ordinary differential equations, *Applied Mathematics and Computation,* 105, 21-68, 1999.
- Rihm, R. Interval methods for initial value problems in ODEs, *Topics in Validated Computations, Proceedings of the IMACS-GAMM International Conference on Validated Computations,* University of Oldenburg, Elsevier Studies in Computational Mathematics, Elsevier, Amsterdam, 1994.