The Abdus Salam
International Centre for Theoretical Physics

United Nations
Educational, Scientific and
Cultural Organization

IAEA
International Atomic Energy Agency

**2247-40**

# School and Conference on Computational Methods in Dynamics

***20 June - 8 July, 2011***

## On the Ubiquity of the Wrapping Effect in the Computation of Error Bounds

Presented by: TUCKER Warwick
Lohner R.J.
*University of Uppsala, S-751 06*
*Uppsala*
*Sweden*

# On the Ubiquity of the Wrapping Effect in the Computation of Error Bounds

Rudolf J. Lohner

Institut für Angewandte Mathematik, Universität Karlsruhe (TH),
D-76128 Karlsruhe, Germany.
rudolf.lohner@math.uni-karlsruhe.de

**Abstract.** Historically, the wrapping effect was discovered and named in the context of solving ordinary initial value problems in interval arithmetic. Its explanation was obviously geometric: rotations of interval vectors enclosing the set of solutions catch excessive points into the enclosure which may eventually 'explode' exponentially. Also discrete dynamical systems share this undesirable behaviour. In the literature the wrapping effect has been discussed primarily in this context.

However, the wrapping effect is not confined to the computation of bounds for dynamical systems – it is much more a phenomenon which occurs concealed within many other problems such as difference equations, linear systems with full or with banded and even triangular matrix, similarly in non-linear systems and even in automatic differentiation. It appears that almost any algorithm which computes rigorous error bounds in some 'iterative' or 'recurrent' fashion may become a victim of the wrapping effect. This paper gives an overview on many such wrapping prone problems as well as on old and new methods designed to eliminate or at least diminish the wrapping effect.

## 1 Introduction or What is the Wrapping Effect?

The wrapping effect was discovered and named by R.E. Moore [14,5] in the context of solving ordinary initial value problems in interval arithmetic. The classical model problem for the explanation of the wrapping effect is the harmonic oscillator with initial values taken in some box:

$$u" + u = 0,\ u(0) \in [u_0],\ u'(0) \in [u_1].$$

The exact solution set in the phase plane is indicated in Fig. 1 by the dashed square rotating clockwise around the origin. A straight forward interval enclosure method takes some time step size $h > 0$ and successively computes enclosures at $t_j = jh$ by starting with the enclosure at $t_{j-1}$ and wrapping the propagated solution set to an interval vector at $t_j$ as is also sketched in Fig. 1.

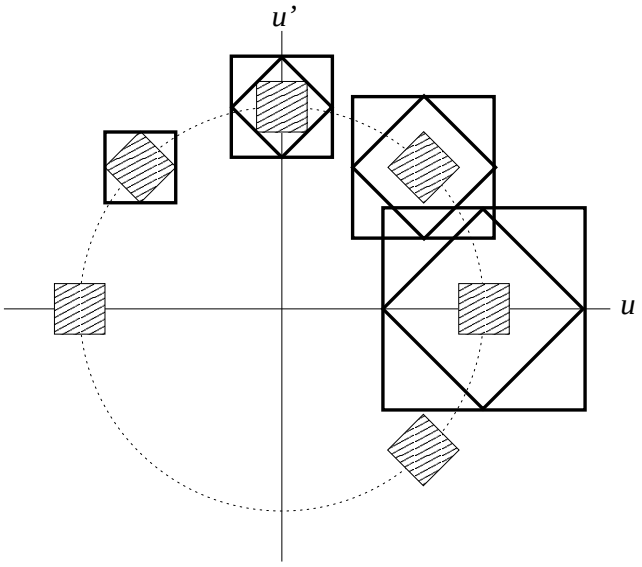


**Figure 1:** Wrapping effect for the harmonic oscillator.

Thus the explanation for the rapid growth of the enclosure set is geometrically obvious: rotations of interval vectors enclosing the solution set catch excessive points into the enclosure which may eventually 'explode' exponentially. Another observation is that no numerical errors such as roundoff or discretization errors are involved in the wrapping operation – it is solely due to the enclosure in interval vectors.

Therefore, we do not even need the context of a differential equation to examine the wrapping effect: A simple matrix vector iteration shows all important details.

If for the rotation $A$ with angle $\phi$ and initial condition $[x_0]$

$$A = \begin{pmatrix} \cos\phi & \sin\phi \\ -\sin\phi & \cos\phi \end{pmatrix}, \quad [x_0] = \begin{pmatrix} [1-\varepsilon, 1+\varepsilon] \\ [1-\varepsilon, 1+\varepsilon] \end{pmatrix}$$

we perform the interval iteration

$$[x_{n+1}] = A[x_n].$$

then the sequence of the diameter vectors $d_n = d([x_n])$ satisfies

$$d_{n+1} = |A| d_n = \begin{pmatrix} |\cos\phi| & |\sin\phi| \\ |\sin\phi| & |\cos\phi| \end{pmatrix} d_n = (\underbrace{|\sin\phi| + |\cos\phi|}_{>1})^n \underbrace{\begin{pmatrix} 2\varepsilon \\ 2\varepsilon \end{pmatrix}}_{d_0}.$$

The $d_n$ diverge exponentially. For small $\phi$ this occurs with a blow up factor $\approx e^{2\pi} \approx 535$ per revolution.

There are other examples where the solution has no rotations in the phase plane but still the wrapping effect occurs massively (e.g. $u_1' = u_1 - 2u_2, u_2' = 3u_1 - 4u_2$, see [10,11]).

In the linear case the exact solution set is an affine image of the initial set. Computing with affine images of interval vectors, i.e. computing with parallelepipeds can completely eliminate the wrapping effect, however, only if all computations are done exactly ([4,10,11,20]). As soon as computational errors have to be taken into account the wrapping effect is still a serious issue. We will discuss this in Section 3.

For nonlinear discrete or continuous dynamical systems the situation is even less favourable. Here the nonlinear dependence on the initial data prevents the a priori choice of some 'optimal' class of enclosure sets. The exact solution set may develop a very complicated shape and become non-convex or even multiply connected. Thus parallelepipeds or even any class of convex sets may be unsuitable for the computation of enclosures. With the recently developed 'Taylor models' ([3,13]) the dependency problem can be handled to a large degree very satisfactorily, however, the computational cost may increase rapidly with the number of variables and the wrapping effect will not be completely eliminated (see Section 3).

We take this as a reason to define only loosely what we mean with the wrapping effect: It is the undesirable overestimation of a solution set of an iteration or recurrence which occurs if this solution set is replaced by a superset of some 'simpler' structure and this superset is then used to compute enclosures for the next step which may eventually lead to an exponential growth of the overestimation. Although not very precisely defined, this notion of the wrapping effect covers the observations made with the simple model problem as well as being useful for more complicated classes of enclosure sets which have been introduced by several authors [3,7–11,13,19,22].

The focus of this paper will not be the dependency problem of nonlinear problems but the fact that in each step of a 'marching algorithm' we have to take into account additional local errors (roundoff, discretization etc.) which force overestimations already in the linear case where we cannot benefit from nonlinear tools.

In Section 2 we will identify many different kinds of problems which are prone to the wrapping effect. The reason for this is basically that these problems can be reformulated as a linear or nonlinear recurrence which reveals where and how they are vulnerable. The enumeration of such problems is by far not complete. Still it gives an impression of how widespread such problems are which might suffer from the wrapping effect if enclosures for their solutions are to be computed. Section 3 gives an overview on different appoaches to eliminate or at least to reduce the wrapping effect. Most of the methods which were proposed by authors in the last four decades are of an intuitive geometric nature. However, also purely algebraic approaches have been considered. Conlusions in Section 4 discuss the present situation especially with a look on the cost of the methods proposed in Section 3.

Throughout the paper we assume that the reader has a basic knowlegde in interval analysis as presented e.g. in [1] and [18]. For brevity of the pre-

sentation we implicitly assume that necessary conditions such as smoothness of functions, existence of interval evaluations etc. are fulfilled wherever used but not stated explicitly.

## 2 Where does the Wrapping Effect appear?

### 2.1 Matrix-Vector Iterations

As in the model problem from Section 1 we consider matrix vector iterations

$$[x_{n+1}] = A_n[x_n] + b_n, \quad [x_0] \in \mathbb{IR} .$$

Here we must expect overestimations, since for the spectral radii of a matrix $A$ and the matrix of its absolute values $|A|$ we have

$$\rho(A) \leq \rho(|A|)$$

and for the diameter vectors $d_n = d([x_n])$ there holds

$$d_{n+1} = |A_n| d_n .$$

If $A_n$ and $b_n$ are allowed to contain intervals too, then

$$d(A_n)|[x_n]| + |A_n| d_n + d(b_n) \geq d_{n+1} \geq |A_n| d_n + d(b_n) .$$

If for example

$$\rho(A) < 1 \text{ and } \rho(|A|) > 1$$

then the solution set

$$\{x_{n+1} = A_n x_n + b_n | x_0 \in [x_0]\}, n \geq 0,$$

may shrink to a point whereas the interval iterates $[x_n]$ diverge.

The same may (and does) happen in matrix-products, matrix-powers as well as in more complicated matrix and matrix-vector expressions.

### 2.2 Discrete Dynamical Systems

For a nonlinear iteration with $f : \mathbb{R}^n \to \mathbb{R}^n$ sufficiently smooth

$$x_{n+1} = f(x_n), \ x_0 \text{ given}$$

or with intervals

$$[x_{n+1}] = f([x_n]), \ x_0 \in [x_0] \text{ given}$$

we can apply a mean-value form which gives tighter enclosures as long as $d([x_n])$ remains small:

$$[x_{n+1}] = f(\tilde{x}_n) + f'([x_n])([x_n] - \tilde{x}_n), \ \tilde{x} \in [x_n].$$

Formally this is a 'linear' iteration scheme as in the previous subsection, however, with an interval matrix $A_n = f'([x_n])$ depending on the iteration index $n$. Therefore we must expect the same kind of difficulties which can occur in the linear case: Iterations may diverge even though the exact solution set stays bounded.

In the mean-value representation the nonlinearity of the original iteration is hidden inside the functional matrix $A_n = f'([x_n])$ whose diameter additionally grows if $d([x_n])$ does.

### 2.3 Continuous Dynamical Systems (ODEs)

For an ordinary initial value problem with $g : \mathbb{R}^{n+1} \to \mathbb{R}^n$

$$x'(t) = g(t, x(t)), \quad x(t_0) = x_0,$$

a numerical one step method which also takes into account all roundoff and discretization errors leads to a discrete system again which basically has the form

$$[x_{n+1}] = [x_n] + h\Phi([x_n], t_n) + [z_{n+1}] \ .$$

Here $h$ is the step size, $t_n = t_0 + nh$, $x_n = x(t_n) \in [x_n]$, $\Phi$ comes from the one step method and $[z_{n+1}]$ is an interval vector containing all local errors. Since this is the type of problem for which the wrapping effect has been studied most intensively, we refer to the literature for further details ([3,4,10,11,14–16]).

As with linear and nonlinear discrete dynamical systems we see that we potentially run into problems with the wrapping effect. Any counter measures that work in the discrete case should also work in the continuous case and vice versa.

### 2.4 Difference Equations

Whereas the previously mentioned problems all were problems for systems in $\mathbb{R}^n$ the following linear difference equation

$$\begin{cases} a_0 z_n + a_1 z_{n+1} + \cdots + a_m z_{n+m} + a_{m+1} z_{n+m+1} = b_n, & n \geq 0 \\ z_0, z_1, \ldots, z_m \text{ given} \end{cases}$$

is a recurrence equation for scalar values $z_n$ only. There seems to be no reason to expect a behaviour similar to the previous cases.

However, rewriting the scalar equation as a matrix-vector iteration by use of $x_n := (z_n, z_{n+1}, \ldots, z_{n+m})^T \in \mathbb{R}^{m+1}$:

$$x_{n+1} = \begin{pmatrix} z_{n+1} \\ z_{n+2} \\ \vdots \\ z_{n+m+1} \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & 1 \\ \dfrac{-a_0}{a_{m+1}} & \cdots & \dfrac{-a_{m-1}}{a_{m+1}} & \dfrac{-a_m}{a_{m+1}} \end{pmatrix} x_n + \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ \dfrac{b_n}{a_{m+1}} \end{pmatrix}$$

it can be seen immediately that both formulations are equivalent not only theoretically but also computationally: Solving the scalar difference equation for $z_{n+m+1}$ requires precisely the same computation as doing one step of the matrix-vector form.

Unexpected overestimations in the scalar computation are often attributed to data dependency. In the vector formulation, however, it becomes evident that such overestimations are due to the wrapping effect in its classical geometric appearance. Data dependence of the initial values is linear only and can be dealt with by the use of parallelepipeds as we will see in Section 3. The techniques described there can be applied only to the matrix-vector formulation however, not to the scalar formulation. This suggests that difference equations should really be treated as matrix-vector iterations if tight error bounds are to be computed.

As an example, consider the recurrence equations for the Chebycheff polynomials

$$T_0(x) = 1, \quad T_1(x) = x, \quad T_{n+1}(x) - 2xT_n(x) + T_{n-1}(x) = 0 \ .$$

If we wish to compute an enclosure of $T_n(x_0)$ for some value $x_0$ and a large value of $n$ then we can use this difference equation and compute the desired value in interval arithmetic. However, this computation is nothing else but forward computation in interval arithmetic for the matrix-vector iteration

$$\begin{pmatrix} T_n(x_0) \\ T_{n+1}(x_0) \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ -1 & 2x_0 \end{pmatrix} \begin{pmatrix} T_{n-1}(x_0) \\ T_n(x_0) \end{pmatrix} \ .$$

Even though we have point data at the beginning interval floating-point arithmetic will introduce roundoff errors. Then a very fast blow up of the enclosures will occur due to the wrapping effect. If the argument $x_0$ is not a floating-point number, then such intervals will enter the computation right from the beginning. If we take e.g. $x_0 = 0.99$ then we obtain the following output from a short PASCAL-XSC program (IEEE double precision) for the values of $n$ and the enclosures for $T_n(0.99)$:

```
n                          T[n](0.99)

 2   [  9.601999999999999E-001,  9.602000000000007E-001 ]
 3   [   9.11195999999999E-001,   9.11196000000002E-001 ]
 4   [   8.43968079999998E-001,   8.43968080000004E-001 ]
 5   [    7.5986079839999E-001,    7.5986079840001E-001 ]
 6   [    6.6055630083198E-001,    6.6055630083202E-001 ]
 7   [     5.480406772473E-001,     5.480406772474E-001 ]
 8   [     4.245642401176E-001,     4.245642401179E-001 ]
 9   [     2.925965181856E-001,     2.925965181861E-001 ]
10   [      1.54776865889E-001,      1.54776865891E-001 ]
15   [       -5.246430527E-001,       -5.246430525E-001 ]
20   [        -9.52088247E-001,        -9.52088240E-001 ]
25   [          -9.222663E-001,          -9.222657E-001 ]
30   [             -4.496E-001,             -4.494E-001 ]
35   [                2.3E-001,                2.5E-001 ]
40   [                6.9E-001,                9.3E-001 ]
45   [               -8.0E+000,                1.0E+001 ]
50   [               -7.1E+002,                7.2E+002 ]
```

Of course the values of the Chebychev polynomials can be computed in a much easier way, since $T_n(x) = \cos(n \arccos x)$. This example should only demonstrate the difficulties that can arise with difference equations even as simple as this one. This example is also important since many other well known functions can be computed by use of certain difference equations such as three term recursions for orthogonal polynomials and special functions like Lagrange polynomials, Bessel functions and many others.

Similarly, nonlinear difference equations can be treated in an obvious way by rewriting them in an equivalent nonlinear vector formulation. This can then be evaluated by the mean-value form as was done for nonlinear discrete dynamical systems. The two representations are theoretically equivalent but no longer computationally. Nevertheless, for small diameters of the iterates we should expect better enclosures from the mean-value formulation however, this is a matrix-vector iteration again (with interval data) and such a problem has already been identified as being susceptible to the wrapping effect.

## 2.5 Linear Systems with (Banded) Triangular Matrix

Forward and backward substitution in interval arithmetic for triangular matrices are known to be mostly unstable processes resulting in large overestimations.

That this is also due to the wrapping effect becomes evident e.g. in the case of banded triangular matrices since there is an obvious equivalence with linear difference equations which in turn are equivalent to matrix-vector iterations as we have demonstrated in the previous subsection.

This is precisely the reason why verification methods for linear systems $Ax = b$ which use an $LU$-factorization usually break down for medium to

high system dimensions unless special methods against the wrapping effect are employed.

The following example is taken from [19]: The system $Ax = b$ with three band lower triangular matrix

$$A = \begin{pmatrix} 1 & 0 & 0 & 0 & \cdots & 0 \\ 1 & 1 & 0 & 0 & & 0 \\ 1 & 1 & 1 & 0 & & 0 \\ 0 & 1 & 1 & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & 1 & 1 & 1 \end{pmatrix}$$

and right hand side $(b_1, 0, 0, \ldots, 0)^T$, $b_1 = [-\varepsilon, \varepsilon]$ can be reformulated as a linear second order difference equation:

$$\begin{cases} x_1 = b_1, \ x_2 = -b_1 \\ x_{n+1} + x_n + x_{n-1} = 0, \ n \geq 2. \end{cases}$$

Computing the solution of this difference equation by solving for $x_{n+1}$ in interval arithmetic is computationally equivalent with interval forward substitution with the original matrix.

For any fixed $b_1$ the exact solution is:

$$x_1 = b_1, \quad x_2 = -b_1, \quad x_n = \begin{cases} -b_1 & \text{for } n = 3k-1 \\ 0 & \text{for } n = 3k \\ b_1 & \text{for } n = 3k+1 \end{cases}$$

Interval forward substitution, however, yields for $b_1 = [-\varepsilon, \varepsilon]$:

$$x_1 = [-\varepsilon, \varepsilon], \ x_2 = [-\varepsilon, \varepsilon], \ x_n = [-a_n\varepsilon, a_n\varepsilon], \ n \geq 3$$

where the $a_n$ are the Fibonacci numbers $a_1 = a_2 = 1$, $a_{n+1} = a_n + a_{n-1}$. Therefore, the diameters $d(x_n)$ diverge exponentially:

$$d(x_n) = 2a_n\varepsilon > \text{const} \cdot 1.62^n.$$

On the other hand the optimal solution set stays bounded:

$$x_n = \begin{cases} 0 & \text{for } n = 3k \\ [-\varepsilon, \varepsilon] & \text{for } n = 3k+1. \end{cases}$$

Again we have seen from the theoretical and computational equivalence of two problems, that susceptibility for the wrapping effect in one problem class induces the same risk for the other problem class. Since the bandwidth itself does not play any role and also the resulting difference equations need not necessarily have constant coefficients this observation holds for any triangular matrix.

### 2.6 Automatic Differentiation

The computation of Taylor coefficients of a scalar function by automatic differentiation (see [10,11,15,18,21]) often involves forward/backward substitution for a triangular matrix. As we know from the previous subsection we must be suspicious and expect to get difficulties here also. Examples are division, square root and virtually all elementary function.

To compute Taylor coefficients $(w)_k$ of a quotient $w = u/v$ of two functions $u, v$ with Taylor coefficients $(u)_k, (v)_k$ we have to solve the triangular Toeplitz system

$$\begin{pmatrix} (v)_0 & 0 & 0 & \cdots & 0 \\ (v)_1 & (v)_0 & 0 & & 0 \\ (v)_2 & (v)_1 & (v)_0 & & 0 \\ \vdots & \ddots & \ddots & \ddots & 0 \\ (v)_n & \cdots & (v)_2 & (v)_1 & (v)_0 \end{pmatrix} \begin{pmatrix} (w)_0 \\ (w)_1 \\ (w)_2 \\ \vdots \\ (w)_n \end{pmatrix} = \begin{pmatrix} (u)_0 \\ (u)_1 \\ (u)_2 \\ \vdots \\ (u)_n \end{pmatrix}$$

for $(w)_0, \ldots (w)_n$. Here, in general, all quantities are intervals due to roundoff errors.

The way how this is usually done is by interval forward substitution which results in the well known and widely used recurrence formulae (see [21]). Here often large overestimations are produced as can be seen from the simple example $f(x) = \sin(e^x)/e^{-x}$. Computing an enclosure of the 30th Taylor coefficient $(f)_{30}$ at $x = -8$ by the recurrence formula in interval arithmetic (IEEE double floating-point) yields $(f(-8))_{30} \in [-6, 6] \cdot 10^{-17}$.

If instead we first precondition the triangular system with an approximate inverse of the mid-point matrix and solve the resulting system by interval forward substitution, then we get practically no wrapping effect and as an enclosure of the 30th Taylor coefficient at $x = -8$ we get $(f(-8))_{30} \in [-8.708693, -8.708691] \cdot 10^{-30}$.

In general this preconditioning technique is quite expensive. However, here we are dealing with a Toeplitz matrix the consequence of which is that all matrix operations can be done at a cost being only quadratic in the system dimension.

## 3 How can we Reduce the Wrapping Effect?

In Section 2 we demonstrated that the straight forward interval solution of many important classes of problems may yield bounds which are too bad by an exponentially growing overestimation. In order to get satisfactory bounds new algorithms have to be designed which are much less vulnerable to the wrapping effect. In this section we give a short overview over different such techniques which have been created by different authors.

We formulate all methods for the special case of matrix-vector iterations. As we have seen in the previous section this is the central problem class in

the sense that once we have a means to reduce the wrapping effect for this special problem class, then we can reformulate other problem classes in order to take advantage from the method for matrix-vector iterations.

We note that a typical property of many of these methods is, that they try to separate (up to a certain degree) the computation of the operator (i.e. the matrix product) and its action on the initial set $x_0$. A successful separation of both has a good chance to avoid many unnecessary wrapping operations – at least if the operator itself does not involve large perturbations (i.e. interval entries with large diameter).

### 3.1 Rearranging Expression Evaluation

We start with a purely algebraic approach to reduce the wrapping effect, see Gambill and Skeel [5]. It is based on the idea of minimizing the depth of the computational graph induced by the matrix-vector iteration thereby having the (somewhat vague) intention to reduce the number of operations with dependent variables:

Instead of computing

$$x_{n+1} = A_n(A_{n-1}(A_{n-2}(\cdots A_1(A_0x_0)\cdots)))$$

which would be the normal order of operations we compute the product of the matrices first and do this in such a way that the expression is not very deeply nested. If $n = 2^m$ then this can be achieved for example by grouping pairs of factors recursively:

$$x_{n+1} = (\cdots((A_nA_{n-1})(\cdots))\cdots((A_3A_2)(A_1A_0))\cdots)x_0$$

Modifications for other values of $n$ can be obtained easily. Also many variations are possible such as taking groups of more than two factors or even varying the number of factors in different groups.

Numerical experiments often show astonishingly narrow enclosures. One reason for this behaviour is, that the method first computes the operator, i.e. the matrix product, before it is applied to the argument $x_0$. For initial values with large diameter and matrices with very small diameters (ideally point matrices) this is a favorable situation.

Nevertheless, the arbitrariness in grouping the terms gives rise to the possibility to construct counterexamples for each given grouping strategy such that the method yields exponentially growing overestimations. For grouping in pairs a simple counterexample is given with all matrices equal to the $2 \times 2$ rotation matrix with $30^o$ rotation: The group pairing will always compute rotations whose angles are a multiple of $30^o$ but never a multiple of $90^o$.

For nonlinear problems this method does not behave satisfactorily since now the matrices are interval matrices which depend on the iterates. This causes the matrix products to blow up rather soon in most cases.

### 3.2 Coordinate Transformations

The oldest approach which also has been continuously modified and developed further by different authors uses enclosures in suitably transformed coordinate systems instead of just taking interval vectors in the original Cartesian system.

A more geometrical interpretation of this approach is that now the enclosing sets are suitably chosen parallelepipeds (i.e. affine images of interval vectors). As already discussed in Section 1 this class of sets is optimal in the linear case if the initial set belongs already to this class and if computations are done exactly. Because of this reason this approach is rather successful also if computation is not exact and generates additional error intervals in each iteration.

A parallelepiped can be represented as $P = \{c + Bx | x \in [x]\}$ with a point vector $c$, a point matrix $B$ and an interval vector $[x]$.
Then the original iteration

$$x_{n+1} = A_n x_n + b_n$$

is replaced by the equivalent one:

$$\begin{cases} x_{n+1} = B_{n+1} y_{n+1}, \quad B_0 := I, \quad y_0 := x_0 \\ y_{n+1} = (B_{n+1}^{-1} A_n B_n) y_n + B_{n+1}^{-1} b_n \end{cases} .$$

In interval arithmetic:

$$\begin{cases} [x_{n+1}] = B_{n+1}[y_{n+1}], \quad B_0 := I, \quad [y_0] := [x_0] \\ [y_{n+1}] = (B_{n+1}^{-1} A_n B_n)[y_n] + B_{n+1}^{-1} b_n \end{cases} .$$

The crucial point in this approach is the choice of basis matrices $B_{n+1}$. Several possibilities will be discussed now.

**Coordinate Transformations – non orthogonal** Choose the basis matrices as good approximations of the iteration map [14,15,4,10,11] .

This means $B_{n+1} \approx A_n A_{n-1} \cdots A_1 A_0$, which can be achieved by choosing $B_{n+1} \approx A_n B_n$ in each step.

An advantage of this choice is that the edges of the enclosing set are roughly parallel to those of the solution set itself (at least in the linear case). This however is at the same time also a disadvantage: If there are dominating eigenvalues (e.g. all $A_i$ are constant with a largest simple eigenvalue) then $B_{n+1}$ may become ill conditioned and usually becomes even singular numerically. In this case the method breaks down.

Unfortunately this is the most common situation: the method behaves similar to the power method for the computation of the dominant eigenvector in $B_{n+1}$ with the consequence that the columns of $B_{n+1}$ become linearly dependent very soon. Then $B_{n+1}$ rapidly becomes ill-conditioned or even singular such that $B_{n+1}^{-1}$ will cause the enclosures to blow up.

This limited applicability may be considerably broadened if we restrict the pairwise angles between the columns of the basis matrix $B_n$ to stay above a minimal prescribed angle. The resulting matrices have bounded condition numbers and offer a great flexibility in choosing an appropriate basis. Up to now this variant has not yet been tested thoroughly, so there is not yet practical experience with this modification.

**Coordinate Transformations – orthogonal** Choose orthogonal basis matrices to keep them well conditioned and invertible [10,11].

The choice of an orthogonal matrix $B_n$ guarantees that it has much more favorable properties than with the previous choice. A suitable orthogonal matrix can be obtained from a $QR$-factorization of $A_nB_n$: Compute $A_nB_n = Q_nR_n$ and choose $B_{n+1} \approx Q_n$. (If computations are done exactly then $B_{n+1}^{-1}A_nB_n = R_n$ is upper triangular.)

It is advisable to apply a pivoting strategy prior to the $QR$-factorization by sorting the columns of $A_nB_n$ in descending order according to the lengths of the columns of $A_nB_n\mathrm{diag}(d([y_n]))$. The columns of this matrix span a good approximation of the exact solution set. The advantage of this pivoting becomes clear from Figure 2: If $a_1, a_2$ are the columns which span an approximation of the solution set (dashed) and which are to be orthogonalized then obviously proceeding in decreasing order of their length (right picture) yields much narrower enclosures than taking a different order (left picture).

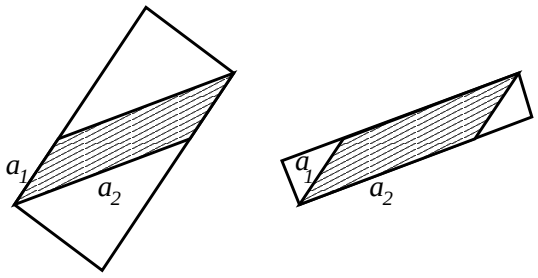


**Figure 2:** Sorting edges prior to orthogonalization.

A big advantage of this method is that it never breaks down because of a singular matrix. In practical computations is has proven to be very robust and mostly delivers astonishingly narrow enclosures.

However, also this method is not infallible. Kühn [7] has constructed an example where this choice of orthogonal basis matrices leads to an exponential overestimation of the solution set:

$$A_0 = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix},\ A_1 = A_0^{-1},\ A_2 = \begin{pmatrix} 2 & 0 \\ 0 & 1 \end{pmatrix},\ A_3 = A_0^T,$$
$$A_4 = A_3^{-1},\ A_5 = A_2^{-1},\ A_{n+6} = A_n,\ n \geq 5\ .$$

After each six steps this iteration is the identity. However, starting the iteration with $[x_0] = ([-1,1],[-1,1])^T$ and choosing orthogonal basis matrices as

described above doubles the initial interval after each six iterations eventually producing an exponential blow up whereas the exact solution set stays bounded.

**Coordinate Transformations – Constant Matrix Case ($A_n = A$)** If the iteration matrices $A_n$ are all identical, $A_n = A$, then it is possible to obtain some quantitative information about its behaviour. A very detailed discussion in this direction can be found in Nedialkov and Jackson [17] in this volume.

Because of the general rule $d(A[x]) \geq |A|d([x])$ the growth of the diameter of $[x_n]$ for the non orthogonal choice of the matrices $B_n$ is dictated by the spectral radius $\rho(|A|)$ which may be considerably larger than $\rho(A)$.

The situation is much more complicated in the case of orthogonal matrices $B_n$. Here, writing $C_{n+1} := B_{n+1}^{-1} A B_n$, the growth of the diameters of $[x_n]$, $[y_n]$ is dictated by $\rho(|C_{n+1}|)$. Since $C_{n+1}$ is upper triangular we have

$$\rho(|C_{n+1}|) = \rho(C_{n+1}) \ .$$

Moreover, since $B_n$ and $B_{n+1}$ are orthogonal it follows that

$$C_{n+1}^T C_{n+1} = B_n^T A^T A B_n \Rightarrow \sigma(C_{n+1}) = \sigma(A)$$

where $\sigma(\cdot)$ denotes the spectral norm (i.e. the largest singular value).

For orthogonal $A$ we trivially obtain always $\rho(|C_{n+1}|) = 1$ which demonstrates that this choice of $B_n$ is ideal in this case.

For symmetric $A$ we obtain the result

$$\rho(|C_{n+1}|) = \rho(C_{n+1}) \leq \sigma(C_{n+1}) = \sigma(A) = \rho(A)$$

which shows that here the orthogonal choice of $B_n$ also yields optimal results concerning the growth of the enclosure. For more and deeper results see [17].

### 3.3 Ellipsoids

Neumaier [19] proposes the use of ellipsoids as enclosure sets. An ellipsoid is represented as

$$E(z, L, r) := \{z + Lx | x \in \mathbb{R}^n, \ ||x||_2 \leq r\}$$

where the $n \times n$-matrix $L$ is chosen to be lower triangular.

In [19] an algorithm is developed which computes enclosures for $x_n$ as ellipsoids. Some numerical examples for constant $2 \times 2$ (interval) iteration matrices $A$ demonstrate the applicability of the algorithm. The results are mostly superior to naive interval arithmetic. There do not, however, exist comparisons with methods that use affine coordinate transformations as discussed earlier. The algorithm for computing with ellipsoids is more complicated than computing with parallelepipeds.

An advantage of the use of ellipsoids could be the fact that ellipsoids have a smooth boundary which make them perhaps easier to handle analytically. Also some applications can be treated in a more natural way as with parallelepipeds (e.g. stability regions and confidence regions, [19]).

### 3.4 Zonotopes

Proposing zonotopes as enclosing sets Kühn has introduced another promising class of sets into the arena against the wrapping effect [7–9]. Zonotopes are the Minkowski sum of line segments and thus are convex polyhedrons. The sum of three line segments and the resulting zonotope is depicted in Figure 3:

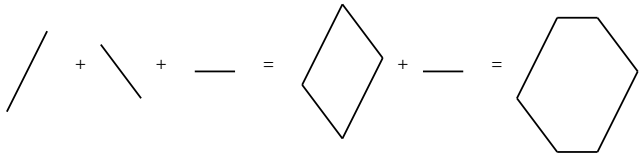

**Figure 3:** Zonotope as sum of three line segments.

A parallelepiped in $\mathbb{R}^n$ is a zonotope: The Minkowski sum of $n$ line segments. This is used by Kühn to represent zonotopes as a sum of $m$ parallelepipeds

$$z := \sum_{k=1}^{m} B_k[z_k] := \{\sum_{k=1}^{m} B_k z_k | z_k \in [z_k], k = 1 \dots m\}$$

which enables comfortable computation:

$$Az + b = \sum_{k=1}^{m} AB_k[z_k] + b = \sum_{k=1}^{m} B'_k[z_k] + b \ .$$

The number $m$ of terms to represent a zonotope is kept constant by wrapping the smallest term in the sum into an interval vector and adding it to the next larger term. Kühn discusses many possible strategies how this can be done in detail. The key point is that wrapping operations are done only with small terms which effectively delays or almost eliminates a blow up of the enclosing zonotope.

The method has a large degree of flexibility due to the free choice of the number $m$ of terms in a zonotope and many possible strategies of wrapping and combining different terms in the representation of a zonotope. For satisfactory results $m$ must not be too small, usually $m = 5, \dots, 10$.

An advantage over the coordinate transformation approach is that no inverse matrices have to be computed. On the other hand, however, the zonotope and coordinate transformation approaches can be combined by using coordinate transformations whenever several terms in the zonotope representation have to be added together. Then the necessary wrapping operations can be performed with much less overestimation than would be obtained by pure interval arithmetic.

### 3.5 Taylor Models

Taylor models were introduced by Berz [3] and Makino/Berz [13]. A Taylor model is an enclosure of a function $f(x)$ by use of a polynomial $T(x)$ with floating-point coefficients and a corresponding remainder term $[I_f]$ which is an interval vector.

$$f(x) \in T(x) + [I_f]$$

Usually $T(x)$ is a (very close) approximation of the Taylor polynomial of $f(x)$ at some specified expansion point and $[I_f]$ contains the corresponding remainder term and all other kinds of errors (round-off, approximation).

Berz and Makino report very successful applications of Taylor models to different types of problems such as computing range of values, univariate and multivariate verified integration, ordinary initial value problems and others.

In the linear case (i.e. $T(x)$ is linear) Taylor models represent a special zonotope, the sum of a parallelepiped plus an interval vector:

Computing a matrix-vector iteration $x_{n+1} = Ax_n$ by use of such a linear Taylor model yields

$$\begin{aligned}
x_0 &= \tilde{x}_0 + (x - \tilde{x}_0), \quad x \in [x_0], \\
x_1 &= Ax_0 = A(\tilde{x}_0 + (x - \tilde{x}_0)) = A\tilde{x}_0 + A(x - \tilde{x}_0) \\
&= \tilde{x}_1 + A_1(x - \tilde{x}_0) + [I_1], \quad x \in [x_0], \\
x_2 &= Ax_1 = A(\tilde{x}_1 + A_1(x - \tilde{x}_0) + [I_1]) \\
&= \tilde{x}_2 + A_2(x - \tilde{x}_0) + [I_2], \quad x \in [x_0], \\
&\vdots \\
x_{n+1} &= Ax_n = A(\tilde{x}_n + A_n(x - \tilde{x}_0) + [I_n]) \\
&= \tilde{x}_{n+1} + A_{n+1}(x - \tilde{x}_0) + [I_{n+1}], \quad x \in [x_0],
\end{aligned}$$

where $\tilde{x}_k$ and $A_k$ are floating-point quantities, $A_0 = I$, and the interval vectors $[I_{k+1}] = A[I_k] + (A\tilde{x}_k - \tilde{x}_{k+1}) + (AA_k - A_{k+1})(x - \tilde{x}_0)$, $[I_0] = 0$ contain all errors. We see that the wrapping effect is not completely eliminated: It is just moved to the smallest term $A[I_k]$ within the Taylor model representation which here at the same time is a zonotope representation.

Taylor models have their strong point in nonlinear problems. To a high degree they solve the dependency problem which we have identified in Section 1 because they approximate this dependence up to the degree of the approximating polynomial $T(x)$. However, the cost for computing such Taylor models can be very high as compared to other methods. Also, as can be seen in the linear case that the wrapping effect is still not completely eliminated. Nevertheless, because of the successful treatment of the dependency problem Taylor models are a valuable new tool.

## 4 Conclusion

We have identified many situations where the wrapping effect may severely blow up computed error bounds if no suitable measures are taken against it. Unfortunately these situations can be problems in which it is not at all obvious that the wrapping effect may be the cause of large overestimations (as e.g. in automatic differentiation).

On the other hand we also gave an overview over different methods which can be used with more or less success to reduce or almost eliminate the influence of the wrapping effect. However, at the present stage no method can be recommended to be always superior to all others or even just to be successful in all applications.

Whereas coordinate transformations and parallelepipeds have proven to be rather robust for some time now, new candidates such as zonotopes and Taylor models are an interesting and promising step ahead.

In summary we state that among the presently available methods there is no general rule available to choose the 'right' one. All have their advantages and disadvantages. There is one thing. however, that is common to all the methods: The cost for computing tight enclosures for an $n$-dimensional vector iteration is at least of order $O(n^3)$ whereas the cost for simple floating-point computation and naive (but usually unacceptable) interval arithmetic is only $O(n^2)$.

Therefore, the main problem which still is unsolved is the question if there are any methods that compute tight bounds without or with only negligible wrapping effect whose cost is only of order $O(n^2)$.

## References

1. Alefeld, G., Herzberger, J.: *An Introduction to Interval Computations.* Academic Press, New York, 1983.
2. Anguelov, R., Markov, S.: *Wrapping effect and wrapping function.* Reliab. Comput. 4, No.4, 311-330 (1998).
3. Berz, M.: *Verified integration of ODEs and flows using differential algebraic methods on high-order Taylor models.* Reliab. Comput. 4, No.4, 361-369 (1998).
4. Eijgenraam, P.: *The Solution of Initial Value Problems Using Interval Arithmetic.* Math. Centre Tracts 144, Mathematisch Centrum, Amsterdam (1981).
5. Gambill, T.N. Skeel, R.D.: *Logarithmic Reduction of the Wrapping Effect with Applications to Ordinary Differential Equations.* SIAM J. Numer. Anal. 25, 153–162 (1988).
6. Jackson, L.W.: *Interval Arithmetic Error-Bounding Algorithms.* SIAM J. Numer. Anal. 12, 223–238 (1975).
7. Kühn, W.: *Rigorously Computed Orbits of Dynamical Systems Without the Wrapping Effect.* Computing 61, No.1, 47-67 (1998).
8. Kühn, W.: *Zonotope Dynamics in Numerical Quality Control.* In: Hege, H-Ch. (ed.) et al.: Mathematical visualization. Algorithms, applications, and numerics. International workshop Visualization and mathematics, Berlin, Germany, September 16-19, 1997. Berlin: Springer. 125-134 (1998).

9. Kühn, W.: *Towards an Optimal Control of the Wrapping Effect.* In: Csendes, Tibor (ed.), Developments in reliable computing. SCAN-98 conference, 8th international symposium on Scientific computing, computer arithmetic and validated numerics. Budapest, Hungary, September 22-25, 1998. Dordrecht: Kluwer Academic Publishers. 43-51 (1999).
10. Lohner, R.J.: *Enclosing the Solutions of Ordinary Initial- and Boundary-Value Problems.* In: Kaucher, E., Kulisch, U., Ullrich Ch. (eds.): Computerarithmetic, pp. 225–286, Teubner Stuttgart (1987).
11. Lohner, R.J.: *Einschließung der Lösung gewöhnlicher Anfangs- und Randwertaufgaben und Anwendungen.* Dissertation, University of Karlsruhe (1988).
12. Lohner, R.J.: *Verified Computing and Programs in Pascal-XSC.* Habilitationsschrift, University of Karlsruhe (1994).
13. Makino, K., Berz, M.: *Efficient control of the dependency problem based on Taylor model methods.* Reliab. Comput. 5, No.1, 3-12 (1999).
14. Moore, R.E.: *Automatic local coordinate transformations to reduce the growth of error bounds in interval computation of solutions of ordinary differential equations.* Error in Digital Comput. 2, Proc. Symp. Madison 1965, 103-140 (1965).
15. Moore, R.E.: *Interval Analysis.* Englewood Cliffs, N.J. Prentice-Hall (1966).
16. Nedialkov, N.S., Jackson, K.R., Corliss, G.F.: *Validated solutions of initial value problems for ordinary differential equations.* Appl. Math. Comput. 105, No.1, 21-68 (1999).
17. Nedialkov, N.S., Jackson, K.R.: *A New Perspective on the Wrapping Effect in Interval Methods for Initial Value Problems for Ordinary Differential Equations.* This Volume. (2001)
18. Neumaier, A.: *Interval methods for systems of equations.* Encyclopedia of Mathematics and its Applications, 37. Cambridge etc.: Cambridge University Press. xvi, (1990).
19. Neumaier, A.: *The Wrapping Effect, Ellipsoidal Arithmetic, Stability and Confidence Regions*, Computing, Suppl. 9, 175–190 (1993).
20. Nickel, K.: *How to fight the wrapping effect.* Lect. Notes Comput. Sci. 212, 121-132 (1986).
21. Rall, L.B.: *Automatic Differentiation: Techniques and Applications.* Lecture Notes in Computer Science, No. 120, Springer-Verlag, Berlin, 1981.
22. Stewart, N.P.: *A heuristic to reduce the wrapping effect in the numerical solution of* $x' = f(t, x)$. BIT 11, 328-337 (1971).