LAA 324, Issue 1-3, pp. 3-13

SELF-VALIDATING METHODS

SIEGFRIED M. RUMP*

Abstract. We present some ideas on self-validating (SV) methods. This note is especially intended to give some background for the articles in this special volume of LAA. It is not intended to be a survey of the big variety of all possible aspects of SV-methods but rather a summary of some basic concepts. Especially, some common misconceptions are mentioned and explored.

1. Introduction.

The basic goals of self-validating methods are

i) to deliver rigorous results
ii) in a computing time not too far from a pure numerical algorithm
iii) including the proof of existence (and possibly uniqueness) of a solution.

The first goal means absolutely rigorous results, of the same nature as mathematical theorems. This includes all possible conversion, rounding or algorithmic errors. In one word, SV-methods deliver true results. Other names for SV-methods are found in the literature, for example verification methods or automatic result verification. Text books include [11, 1, 6, 12], a number of methods and algorithms can be found in [8]. Many more references can be found in the cited literature.

Recently, a number of interesting mathematical problems have been solved with the aid of self-validating methods. Those include the celebrated Kepler conjecture [5], minimal sets of the Jouanolou foliation [3], existence of eigenvalues of the Sturm-Liouville problem below the essential spectrum [2], bounds for the Feigenbaum constant [4], the double bubble conjecture [7], verification of chaos [10, 13], and others.

SV-methods apply to locally continuous problems. For example, the problem "Is a given matrix singular?" is in a certain sense ill-posed. For if the given matrix is indeed singular, then an arbitrary small perturbation changes the answer from yes to no. In this case only exact computation could give a correct answer.

In order to meet the second objective, computations in SV-methods are performed in floating point with some rounding error control (see Section 2.2). Such a computation may be considered as an exact computation with slightly perturbed input data. In this sense an SV-method can never give an answer "yes" to the above question. But a given matrix can be identified very well - using pure floating point arithmetic with some rounding error control - to be definitely not singular, thereby proving that a small neighbourhood of matrices is nonsingular as well.

A simple example for such a proof is the following. Let a real or complex matrix A and some preconditioner R, which could be an approximate inverse of A, be given. Then $I - RA$ is small, and if it can be verified that $\rho(I - RA)$, the spectral radius, is less than one, then obviously A (as well as R) is nonsingular. This in turn is true by Perron-Frobenius theory if, for example, $|I - RA|x < x$ for some positive vector x. And this can obviously be verified in pure floating point arithmetic with some rounding error control.

The above is a simple example of a self-validating method. Basically it is the verification of the assumptions of certain mathematical theorems, the latter being formulated in a suitable way to be applicable for that purpose. Then the assertions are true which include existence and possibly uniqueness of the solution of a given problem within computed bounds. This is the third goal formulated above.

In the following we will sketch two major ingredients for SV-methods, namely, first, arithmetical issues in order to guarantee a safe validation process of assumptions, and, second, more details and examples of such theorems.

* Inst. f. Informatik III, Technical University Hamburg-Harburg, Schwarzenbergstr. 95, 21071 Hamburg, Germany (rump@tu-harburg.de).

2. Arithmetical issues. In order to meet the first and second objective formulated in the previous section we need a fast way to compute rigorous error bounds. A simple way to do this is to keep track of the intermediate errors of computations, and an elegant way to achieve this is interval arithmetic. This is not the only way and, moreover, there is quite a potential for misuse. This is partly the reason for mixed reputation of so-called "interval methods". We will come to that in the next section; here we already want to stress that this has not much to do with self-validating methods.

We will sketch interval arithmetic in two steps, first the theoretical definition and second some practical implementation issues. For better readability, all interval quantities will be in bold face.

2.1. Theoretical definition. A real interval $\mathbf{X} = [x_1, x_2] = \{x \in \mathbb{R} : x_1 \le x \le x_2\}$ is a segment of the real line. We denote the set of real intervals by $\mathbb{IR}$ It may be represented by its end points or by midpoint and radius. All operations between interval quantities satisfy the fundamental property of *inclusion isotonicity*, namely

$$\forall \mathbf{X} \ \forall \mathbf{Y} \ \forall x \in \mathbf{X} \ \forall y \in \mathbf{Y} : \quad x \circ y \in \mathbf{X} \circ \mathbf{Y} \quad \text{for } \circ \in \{+, -, \cdot, /\}. \tag{1}$$

Operations with at least one interval operand are by definition interval operations, although using the same symbol. It is easy to see that the set of all possible results $x \circ y$ for $x \in \mathbf{X}$ and $y \in \mathbf{Y}$ forms a closed interval (for 0 not in a denominator interval), and the end points can be calculated by

$$\mathbf{X} \circ \mathbf{Y} = [\min x_i \circ y_j, \ \ \max x_i \circ y_j].$$

It can also be seen that the only case where more than two operations are necessary is the multiplication with both operands containing 0 as an inner point; in all other cases the two pairs of operands yielding the lower and upper bound can be determined immediately or by case distinctions. So the theoretical overhead for most operations is a factor of two.

The above definition has its pros and cons. A significant drawback is that intervals "have no memory". For example, $\mathbf{X} - \mathbf{X}$ will not result in zero unless $\mathbf{X} = [0, 0]$. Moreover, for $\mathrm{diam}(\mathbf{X}) := x_2 - x_1$ it is easy to see that

$$\mathrm{diam}(\mathbf{X} + \mathbf{Y}) = \mathrm{diam}(\mathbf{X} - \mathbf{Y}) = \mathrm{diam}(\mathbf{X}) + \mathrm{diam}(\mathbf{Y}). \tag{2}$$

On the one hand, the result of $\mathbf{X} \pm \mathbf{Y}$ is narrowest possible, and this is always true when every interval quantity occurs at most once in an expression; on the other hand it is clear that repeated occurence of an interval quantity may imply significant overestimation of the result due to data dependencies. There are a number of new and promising techniques to reduce overestimation using linear programming, special Taylor expansions, Bernstein expansions, convex and quasi-convex envelopes and others.

Real numbers are naturally embedded into the space of intervals $\mathbb{IR}$. In an expression, say $f(x) : \mathbb{R} \to \mathbb{R}$, every operation may be replaced by its corresponding interval operation. Call the resulting function $\mathbf{F} : \mathbb{IR} \to \mathbb{IR}$. Then

$$\forall x \in \mathbb{R} : \quad f(x) \in \mathbf{F}(x)$$

follows from the fundamental principle of inclusion isotonicity (1). Moreover, the argument x may be replaced by an interval $\mathbf{X}$, and again (1) implies the remarkable property that the range of a function given by an arithmetic expression can be estimated without further knowledge of the function:

$$\forall \mathbf{X} \in \mathbb{IR} : \quad \{f(x) : x \in \mathbf{X}\} \subseteq \mathbf{F}(\mathbf{X}). \tag{3}$$

The derived function $\mathbf{F}$ is often called the *natural interval extension* of f. This principle can be extended to elementary standard functions. A straightforward way to include their range is to use a truncated Taylor

series; more subtle methods guarantee very sharp inclusions for the range of all elementary functions for arbitrary (real or complex) input interval.

The set of interval vectors $\mathbb{I}\mathbb{R}^n$ and interval matrices $\mathbb{I}M_{n,k}(\mathbb{R})$ is defined as Cartesian products of $\mathbb{I}\mathbb{R}$. An interval matrix $\mathbf{A}\in \mathbb{I}M_{n,k}(\mathbb{R})$ comprises of all real matrices $A \in M_{n,k}(\mathbb{R})$ with $A_{ij} \in \mathbf{A}_{ij}$ for all i, j. Operations between such quantities derive naturally from the corresponding real operations. However, for given interval matrices $\mathbf{A} \in \mathbb{I}M_{m,k}(\mathbb{R})$, $\mathbf{B} \in \mathbb{I}M_{k,n}(\mathbb{R})$, the set of all products $A \cdot B, A \in \mathbf{A}, B \in \mathbf{B}$, is, in general, no interval matrix. Therefore we define the interval matrix product $\mathbf{A} \cdot \mathbf{B}$ to be the intersection of all interval matrices $\mathbf{C} \in \mathbb{I}M_{m,n}(\mathbb{R})$ containing this set of all products. This is best possible to satisfy inclusion isotonicity. The interval operation $\mathbf{A} \cdot \mathbf{B}$ is not to be confused with the power set operation. One easily shows that

$$\mathbf{A} \cdot \mathbf{B} \in \mathbb{I}M_{m,n}(\mathbb{R}) \quad \text{with} \quad (\mathbf{A} \cdot \mathbf{B})_{ij} := \sum_{\nu=1}^{k} \mathbf{A}_{i\nu} \cdot \mathbf{B}_{\nu j},$$

using scalar interval addition and interval multiplication in the last expression. The definition extends to non-interval factors using the natural embedding $M(\mathbb{R}) \subseteq \mathbb{I}M(\mathbb{R})$.

There are cases where the power set operation yields the same result as the interval operation, for example, if $\mathbf{A}\in \mathbb{I}M_n(\mathbb{R}), x \in \mathbb{R}^n$. Indeed, $\forall y \in \mathbf{A} \cdot x \; \exists A \in \mathbf{A}$ with $y = Ax$. This is because each entry $\mathbf{A}_{ij}$ is used only once to compute $\mathbf{A} \cdot x$. On the other hand, for $A \in M_n(\mathbb{R}), \mathbf{X} \in \mathbb{I}\mathbb{R}^n$ and $y \in A \cdot \mathbf{X}$, there need not exist some $x \in \mathbf{X}$ with $y = Ax$. Geometrically, the power set product $\{Ax : x \in \mathbf{X}\}$ is the linear image of an n-dimensional rectangle, i.e. a parallel-epiped, whereas $A \cdot \mathbf{X}$ is the smallest enclosing interval vector. In $A \cdot \mathbf{X}$, components $\mathbf{X}_j$ are used several times. Such data dependencies are a major reason for overestimations.

Automatic differentiation is a well known way to compute gradients of functions given by means of an arithmetical expression including elementary functions or, more general by some computational scheme. Replacing every operation by its corresponding interval operation (as for the natural interval extension) leads to inclusions of gradients in the same natural way. As before, the range may be overestimated, especially for large input intervals, and SV-methods try to diminish this effect.

Definition of complex interval arithmetic works in a quite similar way, either by infimum/supremum representation and partial ordering or, seemingly more appropriately, by midpoint/radius representation and arithmetic. In contrast to the real case these two representations are not equivalent but have quite different properties.

2.2. Rounding control. The result of arithmetic operations with floating point numbers is, in general, not a floating point number. In order to meet inclusion isotonicity (1) for an interval arithmetic with floating point endpoints and floating point operations we need rounding error control. This can be achieved by multiplying the result by some $1 \pm \varepsilon$, ε denoting the relative rounding error unit. Since the establishment of the IEEE 754 arithmetic standard, optimal floating point operations in a specifiable rounding mode are available. For the following the important rounding modes are $\bigtriangledown$ towards $-\infty$, and $\bigtriangleup$ towards $+\infty$.

Frequently, a processor may be switched into such a rounding mode. This means that subsequent operations are performed in that mode until the next switch. We use the notation that an arithmetic expression in parentheses preceded by a rounding symbol implies that all operations are performed in floating point in the specified rounding mode. Then for a set $\mathbb{F}$ of floating point numbers, e.g. single or double precision including $\pm\infty$, IEEE 754 defines

$$\begin{aligned} \forall a, b \in \mathbb{F} \;\; \forall \circ \in \{+, -, \cdot, /\} \quad & \bigtriangledown(a \circ b) = \max\{x \in \mathbb{F} : x \le a \circ b\} \text{ and} \\ & \bigtriangleup(a \circ b) = \min\{x \in \mathbb{F} : a \circ b \le x\}. \end{aligned} \tag{4}$$

Thus rounding is correct and best possible. Note that this is true for all floating point operands and operations. It holds similarly for the square root. An immediate consequence is

$$\forall a, b \in \mathbb{F} \;\; \forall \circ \in \{+, -, \cdot, /\} : \bigtriangledown(a \circ b) = \bigtriangleup(a \circ b) \;\Leftrightarrow\; a \circ b \in \mathbb{F},$$

where the rightmost operation is the real operation (over $\mathbb{R}$). Establishing inequalities including rounding modes is sometimes a little tricky. For example, for $a, b, c \in \mathbb{F}$,

$$\bigtriangledown(b \cdot c - a) \leq b \cdot c - a, \quad \bigtriangledown(a + b \cdot c) \leq a + b \cdot c \quad \text{and} \quad \bigtriangledown\,(a + (-b) \cdot c) \leq a - b \cdot c$$

using symmetry of $\mathbb{F}$, but $\bigtriangledown(a - b \cdot c) \leq a - b \cdot c$ need not be true for positive quantities a, b, c. However, addition and multiplication can safely be staggered, and by repeatedly using (4) we obtain for matrices $A \in M_{m,k}(\mathbb{F}), B \in M_{k,n}(\mathbb{F})$

$$\bigtriangledown(A \cdot B) \leq A \cdot B \leq \bigtriangleup(A \cdot B). \tag{5}$$

Now the simple example in the previous section can already be formulated as a first self-validating application. Given $R, A \in M_n(\mathbb{F})$ and $0 \leq x \in \mathbb{F}^n$, define

$$C_1 = \bigtriangledown(R \cdot A - I); \quad C_2 = \bigtriangleup(R \cdot A - I); \quad C = \max(|C_1|, |C_2|).$$

Then, using entrywise comparison, $C_1 \leq R \cdot A - I \leq C_2$ and $|I - RA| \leq C$, such that

$$\bigtriangleup(C \cdot x) < x$$

implies R and A to be nonsingular.

A real and complex interval arithmetic can be implemented along the sketched lines . A simple access through Matlab [9] is provided in the recent interval package INTLAB [14]. It is a library (toolbox) entirely written in Matlab and thus easily portable to many platforms. It provides real and complex interval arithmetic including standard functions, automatic differentiation, slopes and more. Due to the operator concept available in Matlab it is easy to use in a readable code, close to mathematical notation. For instance, the above example could be written as follows.

```
if( all( abs( eye(n)-R*intval(A) )*x < x ) )
  disp('R and A are nonsingular')
end
```

Finally we mention that interval arithmetic may be used to compute with non floating point numbers. For example, the transcendental number π may be replaced by a narrow interval $\mathbf{P}$ containing π. Then, any computation involving $\mathbf{P}$ and possible conclusions are valid for replacing $\mathbf{P}$ by any number within $\mathbf{P}$, including π.

3. Common misuse - an example. Consider a standard numerical algorithm, for example Gaussian elimination for the solution of a linear system of equations. It is true that when replacing every operation by its corresponding interval operation, then the true solution will be included in the final result intervals. This is called naive interval arithmetic. However, it is also most likely for a general linear system that the inclusion will be extremely pessimistic and/or the algorithm will end prematurely because a pivot element contains zero.

We will illustrate that with a simple example. Consider

$$A = \begin{pmatrix} 1 & & & 0 \\ & 1 & & \\ & & \ddots & \\ 1 & & & 1 \end{pmatrix} \quad \text{and} \quad b = 0.1 \cdot (1), \text{ where } (1) := \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix}.$$

The solution of the linear system $Ax = b$ is obviously 0.1 times the first unit vector. We try to solve the system by naive interval arithmetic. To see the effect we replace the right hand side by $\mathbf{b} = \mathbf{f} \cdot (1)$ where

$\mathbf{f} := [0.1 - 1e\text{-}10, 0.1 + 1e\text{-}10]$, in short notation $\mathbf{f} = 0.1 \pm 10^{-10}$. Then forward substitution yields

$$\mathbf{X} = \begin{pmatrix} 0.1 & \pm & 1 \cdot 10^{-10} \\ 0 & \pm & 2 \cdot 10^{-10} \\ 0 & \pm & 4 \cdot 10^{-10} \\ 0 & \pm & 8 \cdot 10^{-10} \\ \cdots & & \cdots \end{pmatrix}$$

and $\forall b \in \mathbf{b}$ it is $A^{-1}b \in \mathbf{X}$. Obviously diameters of $\mathbf{X}$ grow exponentially. On the other hand,

$$A^{-1} = \begin{pmatrix} 1 & & & \\ -1 & 1 & & 0 \\ & -1 & \ddots & \\ & & \ddots & 1 \\ 0 & & & -1 \end{pmatrix} \quad \text{and} \quad \mathbf{Y} := A^{-1}\mathbf{b} = \begin{pmatrix} 0.1 & \pm & 1 \cdot 10^{-10} \\ 0 & \pm & 2 \cdot 10^{-10} \\ 0 & \pm & 2 \cdot 10^{-10} \\ & \vdots & \\ 0 & \pm & 2 \cdot 10^{-10} \end{pmatrix},$$

and $\forall b \in \mathbf{b}$ it is $A^{-1}b \in \mathbf{Y}$. In other words, $\mathbf{X}$ is a true inclusion, but a vast overestimation as the result of data dependencies. Mathematically, naive interval forward substitution is equivalent to

$$\mathbf{X} = A^{-1} \cdot \mathrm{mid}(\mathbf{b}) \pm \langle A \rangle^{-1} \cdot \mathrm{rad}(\mathbf{b}),$$

where $\langle A \rangle$ denotes Ostrowski's comparison matrix (i.e. the matrix with diagonal entries $|A_{ii}|$ and off-diagonal entries $-|A_{ij}|$). In our example, the entries of $\langle A \rangle^{-1}$ grow exponentially, and this is not untypical. Note that the above behaviour occurs in exact computation, it has nothing to do with rounding errors but is purely an effect of data dependencies. Also note that the real number 0.1 is purposely chosen to be not finitely exactly representable in binary floating point. Therefore, in order to solve the original linear system, 0.1 has to be replaced by a small interval with the same exponential overestimation. An INTLAB code to try this is the following.

```
n = 20;  A = tril(ones(n));  b = intval('0.1')*ones(n,1);  X = b;
for i=1:n
  X(i) = b(i) - A(i,1:i-1)*X(1:i-1);
end
X
```

A common misconception about what is sometimes called "interval mathematics" is to suppose that the above approach of naive interval arithmetic is all there is to self-validating methods. Almost the opposite is true. SV-methods use the *possibility* to estimate the range of a function in order to verify validity of the assumptions of certain theorems. Because for general data the above behaviour of overestimation is typical, special care is necessary to formulate theorems in such a way to diminish this effect.

4. A simple SV-approach. One typical approach of SV-methods to find a zero of a function $f : \mathbb{R}^n \to \mathbb{R}^n$, described and used several times in the literature, is to transform an equation $f(x) = 0$ into a fixed point equation $g(x) = x$ and to use Brouwer's fixed point theorem. For a nonsingular preconditioning matrix R it is

$$f(x) = 0 \quad \Leftrightarrow \quad g(x) = x, \quad \text{where} \quad g(x) := x - R \cdot f(x).$$

The function g may be considered as a (simplified) Newton operator. Now Brouwer's fixed point theorem and

$$g(\mathbf{X}) \subseteq \mathbf{X} \quad \text{for some} \quad \mathbf{X} \in \mathbb{IR}^n \quad \text{imply} \quad \exists\, x \in \mathbf{X} : \quad f(x) = 0,$$

provided R is nonsingular. This approach uses the possibility (3) to estimate the range of a function. We will illustrate the process with a simple example, a linear system $Ax = b$. An obvious preconditioner is an approximate inverse $R \approx A^{-1}$. The equation $f(x) = Ax - b = 0$ is transformed into $g(x) = x - R(Ax - b)$. It follows that

(6) $\qquad g(\mathbf{X}) \subseteq \mathbf{X}$ for some $\mathbf{X} \in \mathbb{IIR}^n$ and R being nonsingular implies $\exists\, x \in \mathbf{X} : Ax = b$.

However, the application of (2) to interval vectors implies $\mathrm{diam}(g(\mathbf{X})) \geq \mathrm{diam}(\mathbf{X})$, and (6) is, in general, never satisfied. One key point of self-validating methods is not to rearrange interval expressions but the original expression they arose from. Here we want to verify $g(\mathbf{X}) \subseteq \mathbf{X}$ for some $\mathbf{X} \in \mathbb{IIR}^n$. By definition it is $g(x) = Rb + (I - RA)x$, and therefore

$$Rb + (I - RA)\mathbf{X} \subseteq \mathbf{X} \quad \Rightarrow \quad g(\mathbf{X}) \subseteq \mathbf{X}. \tag{7}$$

If $\mathbf{X}$ is a small interval around Rb and R is a reasonably good approximation to the inverse of A, then $I - RA$ is small and (7) is likely to be satisfied. The left hand side of (7) is a simplified form of the frequently used Krawczyk operator. For the assertion $\exists\, x \in \mathbf{X} : Ax = b$ we still need to prove R to be nonsingular. This can be achieved, for example, by the following lemma.

LEMMA 4.1. *Let $\mathbf{Z}, \mathbf{X} \in \mathbb{IIR}^n$ and $\mathbf{C} \in \mathbb{I}M_n(\mathbb{R})$ be given. Suppose (using interval operations)*

$$\mathbf{Z} + \mathbf{C} \cdot \mathbf{X} \subseteq \mathrm{int}(\mathbf{X}). \tag{8}$$

Then every $C \in \mathbf{C}$ is convergent, i.e. $\rho(C) < 1$.

Proof. For every fixed $Z \in \mathbf{Z}, C \in \mathbf{C}$, the inclusion isotonicity (1) implies $Z + C \cdot \mathbf{X} \subseteq \mathrm{int}(\mathbf{X})$. For the midpoint vector $m \in \mathbb{R}^n$ and radius vector $r \in \mathbb{R}^n$ of $\mathbf{X}$, it is $\mathbf{X} = [m - r, m + r] = \{x \in \mathbb{R}^n : m - r \leq x \leq m + r\}$ with entrywise comparisons. Therefore, using entrywise absolute values, it is $C \cdot \mathbf{X} = C \cdot m + [-|C| \cdot r, |C| \cdot r]$. Hence, (8) implies $m - r < Z + C \cdot m - |C| \cdot r \leq Z + C \cdot m + |C| \cdot r < m + r$, and therefore $|C| \cdot r < r$. By Perron-Frobenius theory, $\rho(C) \leq \rho(|C|) < 1$. ∎

This proof is a nice example of the use of matrix theory in the context of self-validating methods. This combined with (7) yields a simple example of an SV-method.

THEOREM 4.2. *Let $A, R \in M_n(\mathbb{R}), b \in \mathbb{R}^n$ and $\mathbf{X} \in \mathbb{IIR}^n$ be given. If*

$$Rb + (I - RA)\mathbf{X} \subseteq \mathrm{int}(\mathbf{X}), \tag{9}$$

then R and A are nonsingular and the unique solution of $Ax = b$ satisfies $A^{-1}b \in \mathbf{X}$.

Coming back to our objectives in Section 1, first the verification method covers all possible conversion, rounding or algorithmic errors, in one word, it delivers true results. Second, measured computing time for an algorithm in INTLAB based on Theorem 4.2 for a linear system with 500 unknowns on a 300 MHz Laptop is 27 sec for the SV-method compared to 4.4 sec for the built-in linear system solver. Recent results by Oishi reduce this computing time in many cases to about 10 seconds. This is a price to pay; on the other hand the SV-method delivers safe information. The third objective is also met because Theorem 4.2 verifies existence and uniqueness of the solution within the computed error bounds.

5. Self-validating methods. In the previous section we gave a simple example of an SV-method, and there is much room for improvement. For example, it turns out to be superior to calculate an inclusion of the difference to an approximate solution, a suitable inclusion $\mathbf{X}$ can be constructed, an interval iteration may be applied and more. The main point is that condition (9) can be rigorously verified on a computer using interval arithmetic as described in Section 2. Rigour includes that *every* computed result is correct. If, due to an ill-conditioned matrix A or a poor preconditioner R, computation of rigorous error bounds is not possible, then a corresponding message is given (rather than an erroneous result).

We have still to comment on the factor 6 to 7 in computing time. For most cases, self-validating methods are not intended to compete with or even replace traditional numerical methods. This also follows by the principle of many SV-methods, that is that error bounds are constructed and verified based upon an approximation. The *raison d'être* of SV-methods is to provide reasonably fast methods to deliver *correct* results (within the computed bounds), for example to solve the mathematical problems mentioned in the introduction or to give certainty if there is doubt about the accuracy of a numerical method.

Self-validating methods can be used to solve classes of problems by inserting interval data. For example, given an interval matrix $\mathbf{A} \in \mathbb{I}M_n(\mathbb{R})$ and an interval right hand side $\mathbf{b} \in \mathbb{I}\mathbb{R}^n$, we may be interested in i) is every $A \in \mathbf{A}$ nonsingular, and ii) what are bounds for $\sum(\mathbf{A}, \mathbf{b}) = \{x \in \mathbb{R}^n \mid \exists\ A \in \mathbf{A}\ \ \exists\ b \in \mathbf{b} : Ax = b\}$. A surprisingly simple solution is based on Theorem 4.2.

THEOREM 5.1. *Let* $\mathbf{A} \in \mathbb{I}M_n(\mathbb{R}),\ R \in M_n(\mathbb{R}),\ \mathbf{b}, \mathbf{X} \in \mathbb{I}\mathbb{R}^n$ *be given. If*

$$R\mathbf{b} + (I - R\mathbf{A})\mathbf{X} \subseteq \mathrm{int}(\mathbf{X}), \tag{10}$$

then R *and every matrix* $A \in \mathbf{A}$ *is nonsingular, and*

$$\sum(\mathbf{A}, \mathbf{b}) = \{x \in \mathbb{R}^n \mid \exists\ A \in \mathbf{A}\ \ \exists\ b \in \mathbf{b} : Ax = b\} \subseteq \mathbf{X}.$$

The PROOF is a typical example for SV-methods and also surprisingly simple. Let fixed but arbitrary $A \in \mathbf{A}, b \in \mathbf{b}$ be given. Then (10) and inclusion isotonicity (1) imply (9), and Theorem 4.2 finishes the proof.

Among other techniques this is a method for proving nonsingularity of every matrix within an interval matrix, thus proving a lower bound for the componentwise distance of the midpoint matrix to the nearest singular matrix weighted by the radius matrix. This problem is known to be NP-hard.

The basic approach of many SV-methods is computation of an approximate solution, local linearization and estimation of linearization and numerical errors by means of suitable theorems the assumptions of which are verified on the computer. Methods for infinite dimensional problems frequently also follow these lines using in addition approximation by a finite dimensional problem together with estimation of the introduced error.

The assumptions to be verified are frequently the inclusion of sets like $Y \subseteq \mathrm{int}(x)$ or, more generally, the verification of an inequality $y < x$. If y and x denote expressions and $\triangle(y) < \triangledown(x)$ is true (cf. Section 2.2), this certainly implies $y < x$. This is a typical step in self-validating methods.

Quite different approaches are used in the promising area of unconstrained and constrained global optimization. Here the capability of estimation of the range of a function over a certain domain proves to be advantageous. Let a function $f : \mathbf{X} \subseteq \mathbb{R}^n \to \mathbb{R}$ to be minimized over $\mathbf{X}$ and a certain (possibly local) minimum $\widetilde{x} \in \mathbf{X}$ be given. If for a subbox $\mathbf{Y} \subseteq \mathbf{X}$ the range is estimated by some $Z = [z_1, z_2] \supseteq f(\mathbf{Y})$ and $f(\widetilde{x}) < z_1$, then $\mathbf{Y}$ does definitely not contain a global minimum. By exploiting this principle together with clever new ways of estimating the range of functions certain subboxes can be excluded from further investigation. This is in fact the biggest problem in global optimization, to be sure that a certain box does *not* contain a minimizer.

6. Conclusion. This short summary of some basic ideas does not, by any means, cover the variety of self-validating methods. Much more can be found in the literature and in this special issue.

SV-methods yield a number of promising results and ideas, but there are many open and untouched problems. For example, algorithms for large, sparse problems are still in a premature status, there are good results for ordinary and partial differential equations but SV-methods are still far from the cutting edge of what may be solved by state-of-the-art numerical algorithms, the treatment of double or nearly double zeros of nonlinear systems ought to be improved, and much more. For the time being, self-validating methods seem

to be an interesting supplement to numerical algorithms, and more and more they have achieved a deserved place in producing rigorous and undoubtedly correct results. And that is what we are after in mathematics.

Acknowledgement. The author wants to thank Jiri Rohn, Beresford Parlett, Hans Schneider and Tetsuro Yamamoto for their thorough reading and valuable comments.

REFERENCES

[1] G. Alefeld and J. Herzberger. *Introduction to Interval Computations*. Academic Press, New York, 1983.

[2] B.M. Brown, D.K.R. McCormack, and A. Zettl. On the existence of an eigenvalue below the essential spectrum. *Proc. R. Soc. Lond.*, 455:2229–2234, 1999.

[3] C. Camacho and L.H. de Figueiredo. The dymanics of the Jouanolou foliation on the complex project 2-space. to appear in: Ergod. Th. & Dynam. Sys.

[4] J.-P. Eckmann and P. Wittwer. *Computer Methods and Borel Summability Applied Feigenbaum's Equation*, volume 227 of *Lecture Notes in Physics*. Springer Verlag, Berlin Heidelberg New York Tokyo, 1985.

[5] T.C. Hales. Cannonballs and Honeycombs. *Notices of the AMS*, 47(4):440–449, 2000.

[6] E.R. Hansen. *Global Optimization using Interval Analysis*. Marcel Dekker, New York, 1992.

[7] J. Hass, M. Hutchings, and R. Schlafly. The Double Bubble Conjecture. *Elec. Res. Announcement of the Am. Math. Soc.*, 1, No. 3:98–102, 1995.

[8] J. Herzberger, editor. *Topics in Validated Computations — Studies in Computational Mathematics*. Elsevier, Amsterdam, 1994.

[9] MATLAB User's Guide, Version 5. The MathWorks Inc., 1997.

[10] K. Mischaikow and M. Mrozek. Chaos in the Lorenz equations: A computer assisted proof. Part II: Details. *Math. Comput.*, 67:1023–1046, 1998.

[11] R.E. Moore. *Methods and Applications of Interval Analysis*. SIAM, Philadelphia, 1979.

[12] A. Neumaier. *Interval Methods for Systems of Equations*. Encyclopedia of Mathematics and its Applications. Cambridge University Press, 1990.

[13] A. Neumaier and Th. Rage. Rigorous Chaos Verification in Discrete Dynamical Systems. *Physica D*, 67:327–346, 1993.

[14] S.M. Rump. INTLAB - INTerval LABoratory. In Tibor Csendes, editor, *Developements in Reliable Computing*, pages 77–104. Kluwer Academic Publishers, 1999. http://www.ti3.tu-harburg.de/rump/intlab/index.html.