ZOR-Methods and Models of Operations Research (1991) 35:87-111

Rigorous Solution of Linear Programming Problems with Uncertain Data

By C. Jansson and S.M. Rump[1]

Abstract. This note gives a synopsis of new methods for solving linear systems and linear programming problems with uncertain data. All input data can vary between given lower and upper bounds. The methods calculate very sharp and guaranteed error bounds for the solution set of those problems and permit a rigorous sensitivity analysis. For problems with exact input data in general the calculated bounds differ only in the last bit in the mantissa, i.e. they are of maximum accuracy.

Keywords: linear programming, programming in conditions of uncertainty, sensitivity, interval and finite arithmetic, systems of equations.

AMS Subject Classification: 90C05, 90C15, 90C31, 65G10, 65H10.

0 Introduction

There are three major sources of errors present when solving a problem on a computer. First, in practical applications the input data are afflicted with tolerances. Secondly, during execution of an algorithm rounding errors occur. Finally, even if some input data are known exactly conversion errors during the input process may occur.

Modern applications of Interval Mathematics to be presented in the following allow to estimate and to control those errors for many problems. The tools of Interval Mathematics are described in several text books ([ALE74], [ALE83], [BAU87], [KUL76], [KUL83], [MOO79], [NEU90]).

In floating-point computations real values are replaced on the computer by approximative floating-point numbers and the real operations by corresponding

[1] This is a written account of an invited lecture delivered by the second author on occasion of the 14. Symposium über Operations Research, Ulm, 6.–8.9.1989.
Dr. C. Jansson and Prof. Dr. S. M. Rump, Informatorik III, Technische Universität Hamburg-Harburg, Eißendorfer Straße 38, 2100 Hamburg 90.

floating-point operations. In Interval Mathematics a real value is replaced by a real compact *interval*

$$[a] := [\underline{a}, \overline{a}] := \{a \in \mathbb{R} \mid \underline{a} \le a \le \overline{a}\} \tag{0.1}$$

where $\underline{a} \le \overline{a}$. The values $\underline{a}$ resp. $\overline{a}$ are called *lower* resp. *upper bound*. On the computer $\underline{a}$ and $\overline{a}$ are floating-point numbers whereas $[a] = [\underline{a}, \overline{a}]$ still represents the set of all *real* numbers between $\underline{a}$ and $\overline{a}$. The specific details of floating-point interval arithmetic will be discussed in section 3. The *radius* of [a] is defined by $\text{rad}([a]) := 0.5 \cdot (\overline{a} - \underline{a})$ and the *midpoint* by $\text{mid}([a]) := 0.5 \cdot (\underline{a} + \overline{a})$. By $\mathbb{IR}$ we denote the set of all real compact intervals. The interval operations on $\mathbb{IR}$ are defined by

$$[a] * [b] := \{a * b \mid a \in [a], b \in [b]\} \tag{0.2}$$

with $[a], [b] \in \mathbb{IR}$ where $* \in \{+, -, \cdot, /\}$ on the right hand side of (0.2) is a real binary operation. In the case of division $0 \notin [b]$ is assumed. In the following the type of the operation (real or interval) will be clear out of its context.

The interval operations can easily be calculated by using the identities

$$\begin{aligned} &[a]+[b] = [\underline{a}+\underline{b}, \overline{a}+\overline{b}] \\ &[a]-[b] = [\underline{a}-\overline{b}, \overline{a}-\underline{b}] \\ &[a]\cdot[b] = [\text{Min}\{\underline{a}\underline{b}, \underline{a}\overline{b}, \overline{a}\underline{b}, \overline{a}\overline{b}\}, \text{Max}\{\underline{a}\underline{b}, \underline{a}\overline{b}, \overline{a}\underline{b}, \overline{a}\overline{b}\}] \\ &[a]/[b] = [\underline{a}, \overline{a}]\cdot[1/\overline{b}, 1/\underline{b}] \end{aligned} \tag{0.3}$$

The set of real n-dimensional vectors x resp. real $m \times n$ matrices A is denoted by $\mathbb{R}^n$ resp. $\mathbb{R}^{mn}$. An interval vector $[x] = ([x_j])$ resp. an interval matrix $[A] = ([a_{ij}])$ is a vector resp. a matrix the coefficients of which are intervals. By $\mathbb{IR}^n$ we denote the set of interval vectors $[x]$ with n components and by $\mathbb{IR}^{mn}$ we denote the set of $m \times n$ interval matrices $[A]$. The interval operations are extended to interval matrices and interval vectors analogously to the real vector and matrix operations:

$$\begin{aligned} &[r]\cdot[A] := ([r]\cdot[a_{ij}]),\ [r] \in \mathbb{IR},\ [A] \in \mathbb{IR}^{mn} \\ &[A]+[B] := ([a_{ij}]+[b_{ij}]),\ [A], [B] \in \mathbb{IR}^{mn} \\ &[A]\cdot[B] := \left(\sum_{l=1}^{k} [a_{il}]\cdot[b_{lj}]\right),\ [A] \in \mathbb{IR}^{mk},\ [B] \in \mathbb{IR}^{kn} \end{aligned} \tag{0.4}$$

Setting $n = 1$ or $m = 1$ this definition includes the operations for interval vectors. Diameter and midpoint of interval vectors and matrices are defined componentwise.

The operations (0.3), (0.4) can be realized on a computer in a straightforward manner if directed roundings are available. If the processor in use satisfies IEEE 754 [IEEE85], the necessary rounding modes are available. With the arithmetic of Kulisch [KUL76, KUL83] vector and matrix operations can be executed with maximum accuracy (compare section 3).

The key property of Interval Mathematics is the *isotonicity*

$$[a], [b] \in \mathbb{IR};\ a \in [a],\ b \in [b] \Rightarrow a * b \in [a] * [b] \tag{0.5}$$

which extends to the operations between interval matrices and interval vectors and all other calculations. These calculations deliver (also on a computer) *guaranteed* bounds. For example, calculating in (0.4) the product $[A]\cdot[B]$ of interval matrices the computed result $[C] = [\underline{C}, \overline{C}]$ gives a lower and a upper bound $\underline{C}, \overline{C}$ with

$$A \in [A],\ B \in [B] \Rightarrow A \cdot B \in [C].$$

Remember that $[A]$ comprises of all *real* matrices $A \in \mathbb{R}^{mn}$ with $\underline{A} \le A \le \overline{A}$. In practice systems of linear equations frequently occur the coefficients of which are not known exactly but are afflicted with tolerances. In this case one is interested in the set of all solutions for real linear systems with system matrix and right hand side within the tolerances. Using the notation of Interval Mathematics this means for some $[A] \in \mathbb{IR}^{nn}$ and $[b] \in \mathbb{IR}^n$ we are interested in guaranteed bounds for the *solution set*

$$X([A], [b]) = \{x \in \mathbb{R}^n \mid Ax = b,\ A \in [A],\ b \in [b]\}. \tag{0.6}$$

In the following we use formally the notation

$$[A]x = [b] \tag{0.7}$$

for a system of linear equations with interval input data; to "solve" this system means to calculate an interval vector $[x]$ with $X([A], [b]) \subseteq [x]$.

Replacing in Gaussian Elimination the real operations by the corresponding interval operations (0.2) an interval vector $[x]$ which contains the solution set $X([A], [b])$ is calculated. This is true because of the fundamental principle of isotonicity. Hence the interval version of Gaussian Elimination is a method for solving the system (0.7). Unfortunately dependencies of the input data during the calculation may yield to severe overestimations or to division by an interval containing zero. Using this approach this may happen even for examples of small size. The reason for this is: when the same interval occurs in a calculation several times the result tends to overestimate the true (narrowest) result. A simple example is the calculation of

$$[x]-[x]=[\underline{x}-\overline{x},\ \overline{x}-\underline{x}] \overset{\text{i.g.}}{\neq} 0 .$$

Therefore special algorithms (compare section 2) have to be developed to avoid the phenomenon of overestimation.

In section 1 we will describe a method for computing guaranteed error bounds for the solution set of a linear programming problem with uncertain data covering the *basisinstable* case. This method can be viewed as a graph-search-method on a graph where the nodes are the optimal basic-index-sets. The computed guaranteed error bounds are bounds for the solution sets of linear systems which correspond to the optimal basic-index-sets.

In section 2 methods for solving linear systems with uncertain data are described. It will be shown that the computed bounds are in general very sharp. In fact the sharpness of the bounds itself will be estimated. With these methods a complete sensitivity analysis together with guaranteed error bounds for the solution of linear systems and linear programming problems is given. It should be stressed that all input data can vary between lower and upper bounds.

In section 3 computational details and hints for an implementation on digital computers will be given. In section 4 numerical examples are demonstrated.

1 Solving Linear Programming Problems with Uncertain Data

We consider the linear programming problem in *standard form*

$$\operatorname{Max}\{c^t x \mid x \in X\},\ X := \{x \in \mathbb{R}^n \mid Ax = b,\ x \geq 0\} \tag{1.1}$$

where $A = (a_1, \ldots, a_n)$ is a real $m\times n$ matrix with column vectors $a_1, \ldots, a_n \in \mathbb{R}^m$, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$ and $m < n$. Associated with (1.1) is the corresponding *dual linear programming problem*

$$\operatorname{Min}\{b^t y \mid y \in Y\},\ Y := \{y \in \mathbb{R}^m \mid A^t y \geq c\} . \tag{1.2}$$

The input data of (1.1) and (1.2) are given by the tripel $P = (A, b, c) \in \mathbb{R}^{mn+m+n}$.

The set of indices $B = \{\beta_1, \ldots, \beta_m\} \subseteq \{1, \ldots, n\}$ is called a *basic-index-set* if the corresponding $m \times m$ submatrix $A_B = (a_{\beta_1}, \ldots, a_{\beta_m})$ is non-singular.

$N := \{\gamma_1, \ldots, \gamma_{n-m}\} := \{1, \ldots, n\} / B$ is called the set of *nonbasic indices* and $A_N := (a_{\gamma_1}, \ldots, a_{\gamma_{n-m}})$. For a given basic-index-set B the well-known *simplex tableau* $T(B)$ is defined by

$$T(B) := \left(\begin{array}{c|c} z & d_N^t \\ \hline x_B & S_N \end{array}\right) \tag{1.3}$$

where $x_B := A_B^{-1} b$, $y(B) := (A_B^t)^{-1} c_B$, $d_N := A_N^t y(B) - c_N$, $z := b^t y(B)$, $S_N := A_B^{-1}$
A_N and $c := (c_B, c_N)^t$ is partioned analogously to $A = (A_B, A_N)$.
A basic-index-set B is called *optimal* if $x_B \geq 0$ and $d_N \geq 0$. V_{opt} denotes the set
of all optimal basic-index-sets. It is well-known that $x(B) := (x_B, x_N)^t$, $x_B := A_B^{-1} b$,
$x_N := 0$ is an optimal vertex of (1.1) and $y(B)$ is an optimal vertex of (1.2) if $B \in$
V_{opt}. Moreover the optimal value $z_{\text{opt}} := \operatorname{Max}\{c^t x \mid x \in X\}$ exists and satisfies z_{opt}
$= b^t y(B) = c^t x(B)$, $\hat{X}_{\text{opt}} := \{x(B) \mid B \in V_{opt}\}$ is the set of all optimal vertices of
(1.1) and $Y_{opt} := \{y(B) \mid B \in V_{opt}\}$ is the set of all optimal vertices of (1.2).

The interesting output data are therefore the *optimal value* z_{opt}, the *set of optimal basic-index-sets* V_{opt} *and the sets of optimal vertices* $\hat{X}_{opt}$ *and* $\hat{Y}_{opt}$.

To perform a rigorous sensitivity and error analysis of linear programming problems we describe the input data by the tripel

$$[P] = ([A], [b], [c]) \tag{1.4}$$

where $[A]$ is an $m\times n$ interval matrix, $[b]$ is an interval vector with m components and $[c]$ is an interval vector with n components. By the tripel $[P]$ a set of linear programming problems $P \in [P]$ with real input data $P = (A, b, c) \in \mathbb{R}^{mn+m+n}$ is given. In the following we write $z_{opt}(P)$, $V_{\text{opt}}(P)$, $\hat{X}_{\text{opt}}(P)$, $\hat{Y}_{\text{opt}}(P)$, $T(B, P)$, $x(B, P)$, ... to indicate the dependency on $P \in [P]$.

The interesting output data of the linear programming problems with $P \in [P]$ are the sets

$$z_{\text{opt}}([P]) := \{z_{\text{opt}}(P) \mid P \in [P]\} \tag{1.5}$$

$$V_{\text{opt}}([P]) := \cup\, \{V_{\text{opt}}(P) \mid P \in [P]\} \tag{1.6}$$

$$\hat{X}_{\text{opt}}([P]) := \{x(B, P) \mid P \in [P],\ B \in V_{\text{opt}}(P)\} \tag{1.7}$$

$$\hat{Y}_{\text{opt}}([P]) := \{y(B, P) \mid P \in [P],\ B \in V_{\text{opt}}(P)\} \tag{1.8}$$

We are interested in reliable and very sharp lower and upper bounds for the output data. In this section we describe a method which allows to calculate an interval $[z]$, interval vectors $[x^1], \ldots [x^s]$, $[y^1], \ldots, [y^s]$ and a set of index-sets $Z = \{B^1, \ldots, B^s\}$ such that the following conditions are fulfilled:

1° $z_{\text{opt}}([P]) \subseteq [z]$

2° $V_{\text{opt}}([P]) \subseteq Z$

3° $\hat{X}_{\text{opt}}([P]) \subseteq [x^1] \cup \ldots \cup [x^s]$

4° $\hat{Y}_{\text{opt}}([P]) \subseteq [y^1] \cup \ldots \cup [y^s]$

The method either computes the above inclusions and guarantees that for all real problems $P \in [P]$ there exists an optimal solution or, a warning is given that no inclusions could be calculated. In the first case the computed interval vectors give guaranteed bounds for the variation of the output data for P in $[P]$. Therefore a complete sensitivity and error analysis is given for a linear programming problem where all input data can vary between lower and upper bounds. In section 2 we will show that the computed bounds are in general very sharp.

The presented method can be viewed as a graph-search-method (cf. [PAP82]) applied to the graph:

$$G_{\text{opt}}([P]) := (V_{\text{opt}}([P]), E_{\text{opt}}([P])) \tag{1.9}$$

with the node set $V_{\text{opt}}([P])$ and the edge set

$$E_{\text{opt}}([P]) := \left\{ \{B, B'\} \;\middle|\; \begin{array}{l} B, B' \text{ differ in exactly one index and there} \\ \text{exists a } P \in [P] \text{ with } B, B' \in V_{\text{opt}}(P) \end{array} \right\} \tag{1.10}$$

We call $G_{\text{opt}}([P])$ the *representation graph* of $[P]$. Two nodes B^1, B^2 are called to be *neighboured* if $\{B^1, B^2\} \in E_{\text{opt}}([P])$. For $B \in V_{\text{opt}}([P])$ the set $N(B)$ denotes the set of all nodes B' in $V_{\text{opt}}([P])$ which are neighboured to B.

In the first step of the method a starting node B' in $V_{\text{opt}}([P])$ is computed. This can be done for example using the classical simplex algorithm. Following all nodes in $N(B_{\text{start}})$ are computed. Subsequently all nodes in $N(B')$ with $B' \in N(B_{\text{start}})$ are calculated and so forth.

To calculate the sets $N(B)$ with $B = \{\beta_1, \ldots, \beta_m\} \in V_{\text{opt}}([P])$ we need an inclusion of all simplex tableaus $T(B, P)$ with $P \in [P]$. Solving the system of linear interval equations

$$\begin{aligned} &[A_B] x_B = [b] \\ &[A_B]^t y(B) = [c_B] \\ &[A_B] s_\gamma = [a_\gamma],\ \gamma \in N := \{\gamma_1, \ldots, \gamma_{n-m}\} \end{aligned} \tag{1.11}$$

interval vectors $[x_B]$, $[y(B)]$, $[s_\gamma]$ are obtained with $\gamma \in N$.

Now we define

$$[d_N] := [A_N]^t [y(B)] - [c_N], \tag{1.12}$$

$$[x(B)] := ([x_B], x_N)^t,\ x_N := 0, \tag{1.13}$$

$$[S_N] := ([s_{\gamma_1}], \ldots, [s_{\gamma_{n-m}}]), \tag{1.14}$$

$$[z(B)] := [c_B]^t [x_B] \cap [b^t][y(B)], \tag{1.15}$$

$$[T(B)] := \left(\begin{array}{c|c} [z(B)] & [d_N]^t \\ \hline [x_B] & [S_N] \end{array} \right) = \left(\begin{array}{c|ccc} [z(B)] & [d_{\gamma_1}] & \cdots & [d_{\gamma_{n-m}}] \\ \hline [x_{\beta_1}] & [s_{\beta_1\gamma_1}] & \cdots & [s_{\beta_1\gamma_{n-m}}] \\ \vdots & \vdots & & \vdots \\ [x_{\beta_m}] & [s_{\beta_m\gamma_1}] & \cdots & [s_{\beta_m\gamma_{n-m}}] \end{array} \right). \tag{1.16}$$

$[T(B)]$ is called the *interval tableau* corresponding to B. Obviously the simplex tableaus $T(B, P) \in [T(B)]$ for $P \in [P]$. The set $N(B)$ can be effectively calculated according to the following theorem.

Theorem 1.1.: Let B be a basic-index-set and $[T(B)]$ the corresponding interval tableau with $\underline{x}_B \geq 0$, $\underline{d}_N \geq 0$. Let $\tilde{N}(B)$ be defined as the set of all index-sets

$$B' = (B \setminus \{\beta\}) \cup \{\gamma\},\ \beta \in B,\ \gamma \in N \tag{1.17}$$

satisfying one of the following conditions

$$0 \in [d_\gamma],\ \bar{s}_{\beta\gamma} > 0 \text{ and } \frac{\underline{x}_\beta}{\bar{s}_{\beta\gamma}} \leq \operatorname{Min}\left\{ \frac{\bar{x}_{\beta'}}{\underline{s}_{\beta'\gamma}} \;\middle|\; \underline{s}_{\beta'\gamma} > 0,\ \beta' \in B \right\} \tag{1.18}$$

$$0 \in [x_\beta],\ \underline{s}_{\beta\gamma} < 0 \text{ and } \frac{\underline{d}_\gamma}{\underline{s}_{\beta\gamma}} \geq \operatorname{Max}\left\{ \frac{\bar{d}_{\gamma'}}{\bar{s}_{\beta\gamma'}} \;\middle|\; \bar{s}_{\beta\gamma'} < 0,\ \gamma' \in N \right\}. \tag{1.19}$$

Then $\tilde{N}(B) \supseteq N(B)$ if $B \in V_{\text{opt}}([P])$.

It should be remarked that $\tilde{N}(B)$ and $N(B)$ are empty if $[x_B] > 0$ and $[d_N] > 0$. In this case we call $[P]$ *basisstable*. With theorem 1.1 we can describe the following method:

```
(1): compute $B_{\text{start}} \in V_{\text{opt}}([P])$;
     $S := \emptyset$, $Z := \emptyset$, $B := B_{\text{start}}$
     goto (4);
(2): if $S = \emptyset$ then STOP;
```

```
(3): choose $B \in S$; $S := S \setminus \{B\}$;

(4): a) compute the interval vectors of the systems (1.11); if some interval
        vectors can not be calculated then STOP (WARNING);

     b) compute $[T(B)]$;
        if one of the following conditions
           (i)  $0 \in [x_\beta]$ with $\beta \in B \Rightarrow$ there exists a
                $\gamma \in N$ with $[s_{\beta\gamma}] < 0$
           (ii) $0 \in [d_\gamma]$ with $\gamma \in B \Rightarrow$ there exists a
                $\beta \in B$ with $[s_{\beta\gamma}] > 0$
           is not true
        then STOP (WARNING);

     c) if {there exists a $\beta \in B$ with $[x_\beta] < 0$ or there exists a
              $\gamma \in N$ with $[d_\gamma] < 0$}
        then goto (2);

     d) compute $\tilde{N}(B)$ using theorem 1.1;
        $Z := Z \cup \{B\}$; $S := S \cup \{N(B) \setminus Z\}$;
        goto (2);
```

Theorem 1.2.: If the algorithm described above stops with step (2) then

1. The sets of optimal solutions of the standard form (1.1) and the dual problem (1.2) are nonempty and bounded for every $P \in [P]$. Moreover the function of optimal values $z_{\text{opt}} : [P] \to \mathbb{R}$ with $z_{\text{opt}}(P) := \text{Max}\{c(P)^t x \mid x \in X(P)\}$ is continuous and the representation graph $G_{\text{opt}}([P])$ is connected.
2. The computed set of basic-index-sets $Z := \{B^1, \ldots, B^s\}$, the computed interval vectors $[x(B^1)], \ldots, [x(B^s)], [y(B^1)], \ldots, [y(B^s)]$ and the interval $[z] := [z(B^1)] \cup \ldots \cup [z(B^s)]$ satisfy 1°, 2°, 3°, 4°.

The proofs of theorem 1.1 and theorem 1.2 are far too lengthy to be included in these notes (cf. [JAN85]). There are some papers treating the basisstable case (cf. [BEE78], [KRA75], [MAC70], [ROH84], [RU85], [STEU81]).

Executing the algorithm above on a computer the first part of theorem 1.2 implies in particular the existence of optimal solutions for all $P \in [P]$.

The warnings in step (4) either point out the existence of singular or almost singular matrices A_B with $B \in V_{\text{opt}}([P])$ or, that the sets of feasible solutions $X(P)$, $Y(P)$ may be empty resp. that the objective function is unbounded for some $P \in [P]$.

Obviously $B \notin V_{\text{opt}}([P])$ if condition (4) c), is true. Hence B is not admitted to Z.

The main effort in the algorithm is to compute inclusion vectors of (1.11) and the interval tableaus $[T(B)]$. With $[T(B)]$ the set $N(B)$ is computed according to theorem 1.1.

Obviously it is sufficient to obtain inclusion vectors $[x_B]$, $[d_N]$ and inclusion vectors of the columns of $[S_N]$ with $0 \in [d_\gamma]$ and the rows of $[S_N]$ with $0 \in [x_\beta]$. In practical applications $[d_\gamma] > 0$ and $[x_\beta] > 0$ for most indices $\gamma \in N$, $\beta \in B$. The computational costs can then be reduced significantly as in the revised simplex method.

The sharpness of the computed bounds for the linear programming problem $[P]$ depends only on the algorithm for calculating the bounds of the linear systems (1.11). We proceed by describing this solution process.

2 Solving Linear Systems with Uncertain Data

According to the results presented in the last section solving linear programming problems the data of which are afflicted with tolerances requires the possibility to bound the solution set of a linear system with uncertain data. That means for some $[A] \in \mathbb{IR}^{nn}$ and $[b] \in \mathbb{IR}^n$ we are interested in very sharp bounds $[x]$ with $X([A], [b]) \subseteq [x]$. In general the solution set $X([A], [b])$ consists of infinitely many points $x \in \mathbb{R}^n$.

For simplicity we will use in addition the notation

$$a \pm \delta := [a - \delta, a + \delta] \text{ with } \delta \ge 0 \tag{2.1}$$

for intervals, interval vectors and interval matrices.

Consider a simple example. Let

$$[A] := \begin{bmatrix} 0.73 & 0.76 \\ -2.80 & 0.86 \end{bmatrix} \pm \begin{bmatrix} 0.003 & 0.001 \\ 0.002 & 0.001 \end{bmatrix} \text{ and } \quad [b] := \begin{bmatrix} 0.3 \\ -2.7 \end{bmatrix} \pm \begin{bmatrix} 0.01 \\ 0.03 \end{bmatrix} \tag{2.2}$$

The tolerances in the matrix $[A]$ are between 0.07% and 0.41%, the tolerances in $[b]$ between 1.1% and 3.3%.

Then the solution set $X([A], [b])$ can be calculated explicitly by a number of linear programming problems. For nonnegative $x \in \mathbb{R}^n$

$$x \ge 0 : x \in X([\underline{A}, \overline{A}], [\underline{b}, \overline{b}]) \Leftrightarrow \underline{A}x \le \overline{b} \text{ and } \overline{A}x \ge \underline{b} \tag{2.3}$$

(cf. [ROH84], [DEI86]). Checking all combinations of signs for the components of x results in 2^n linear programming problems all together exactly describing $X([A], [b])$. Moreover, the intersection of $X([A], [b])$ with every orthant is a polyhedron consisting, in general, of exponentially many vertices. The result for example (2.2) is depicted in the following figure.

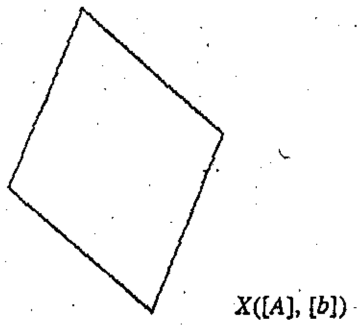


Fig. 2.1. Solution complex for (2.2)

The computational costs may be reduced a little bit for special cases. In general this approach is not practicable even for problems of small size.

According to the previous section it is not necessary to know the precise shape of $X([A], [b])$ but bounds for it, e.g. it would suffice to know some interval vector $[x] \in \mathbb{IIR}^n$ with $X([A], [b]) \subseteq [x]$ where the distance between the boundaries of $X([A], [b])$ and $[x]$ is small. Following we will describe a solution to this problem and a corresponding algorithm will be given as well. The computational costs are $2n^3 + 0(n^2)$ operations (1 operation = 1 addition + 1 multiplication).

The theoretical foundation to be described was given in [RU80] and [RU83] and a number of papers being referenced over there. There the general case of systems of nonlinear equations with uncertain data was treated. In the following we restrict our attention to systems of linear equations. This allows a very simple formulation and short and elementary proofs.

Theorem 2.1.: Let $[A] \in \mathbb{IIR}^{nn}$, $[b] \in \mathbb{IIR}^n$ be given and let for some $R \in \mathbb{R}^{nn}$, $\tilde{x} \in \mathbb{R}^n$ and $[x] \in \mathbb{IIR}^n$

$$R \cdot ([b] - [A] \cdot \tilde{x}) + \{I - R \cdot [A]\} \cdot [x] \subseteq \text{int}([x]) \tag{2.4}$$

Then R and every matrix $A \in [A]$ are non-singular and for all $A \in [A]$, $b \in [b]$ the corresponding linear system $Ax = b$ is uniquely solvable with solution $\hat{x}$ satisfying $\hat{x} \in \tilde{x} + [x]$.

Therefore

$$X([A], [b]) \subseteq \tilde{x} + [x]. \tag{2.5}$$

Remark. All operations in (2.4) are interval operations, I denotes the $n \times n$ identity matrix and int ([x]) denotes the interior of [x].

Proof. Using the central property of interval arithmetic, the isotonicity (0.5) we get by (2.4) for all $A \in [A]$, $b \in [b]$

$$R \cdot (b - A\tilde{x}) + \{I - RA\} \cdot x \subseteq \text{int}([x]) \quad \text{for all } x \in [x]. \tag{2.6}$$

In (2.6) all operations are the real matrix and vector operations. The function $f(x) : \mathbb{R}^n \to \mathbb{R}^n$ with

$$f(x) := R \cdot (b - A\tilde{x}) + \{I - RA\} \cdot x = x + R \cdot (b - A \cdot (\tilde{x} + x)) \tag{2.7}$$

is continuous and by (2.6) f maps [x] into itself. In fact according to (2.6) we even have

$$f([x]) \subseteq \text{int}([x]). \tag{2.8}$$

Therefore Brouwer's Fixed Point Theorem implies the existence of some $\hat{x} \in [x]$ with

$$f(\hat{x}) = \hat{x}, \tag{2.9}$$

hence $R \cdot (b - A(\tilde{x} + \hat{x})) = 0$.

Assume $A \cdot y = 0$ for some $y \in \mathbb{R}^n$. Then by (2.7) and (2.9) for every $\lambda \in \mathbb{R}$

$$f(\hat{x} + \lambda y) = \hat{x} + \lambda y - RAy = \hat{x} + \lambda y. \tag{2.10}$$

For $y \neq 0$ there would be some $\hat{\lambda}$ with $\hat{x} + \hat{\lambda} y = f(\hat{x} + \hat{\lambda} y) \in \partial[x]$ which contradicts (2.8) and implies $y = 0$. Hence $Ay = 0$ is only true for $y = 0$ which means A is not singular. Assume $Ry = 0$ for some $y \in \mathbb{R}^n$. Then $A^{-1}y$ is well-defined and by (2.7) and (2.9) for every $\lambda \in \mathbb{R}$

$$f(\hat{x} + \lambda A^{-1}y) = \hat{x} + \lambda A^{-1}y - RAA^{-1}y = \hat{x} + \lambda A^{-1}y.$$

For $y \neq 0$ implying $A^{-1}y \neq 0$ there would be some $\hat{\lambda} \in \mathbb{R}$ with $\hat{x}+\hat{\lambda}A^{-1}y = f(\hat{x}+\hat{\lambda}y) \in \partial[x]$ which contradicts (2.8) and implies $y = 0$. Hence R is not singular. Then by (2.9) follows

$$b - A(\tilde{x}+\hat{x}) = 0$$

and using $\hat{x} \in [x]$ we have $A^{-1} \cdot b \in \tilde{x} + [x]$. These conclusions hold for all $A \in [A]$, $b \in [b]$ finishing the proof. □

In practical applications R is an approximate inverse of A (which may be replaced by an LU-decomposition) and $\tilde{x}$ is an approximate solution of $Ax = b$. It should be stressed that there are neither a priori assumptions on the quality of R resp. $\tilde{x}$ in terms of a maximum distance to the exact values A^{-1} resp. $A^{-1}b$ nor even the existence of A^{-1} for every $A \in [A]$ is assumed *a priori*. The only assumption is (2.4) which implies all assertions of theorem 2.1.

In the literature frequently $\| I-RA \| < 1$ is assumed for some norm to conclude the non-singularity of R and A. Our assumption (2.4) requires for the spectral radius $\rho(|I-RA|) < 1$, where the absolute value is to be taken componentwise. This is shown in [RU83]. Over there examples are given where $\| I-RA \| > 10^6$ for 1–, 2–, ∞– and Frobenius-Norm whereas (2.4) is still satisfied. Later on we will see that $\rho(|I-RA|) > 1$ is a necessary and sufficient condition to find some $[x] \in \mathbb{R}^n$ with (2.4).

For the moment we postpone the problem to check (2.4) on the computer (which is, as we will see, very simple to solve) and concentrate on analyzing (2.4). Theorem 2.1 can be regarded as a check on the validity of an error bound $[x]$ for the solution set $X([A],[b])$ w.r.t. an approximation $\tilde{x}$. The main problem is then to find an appropriate $[x]$. One may try an iteration of the form

$$[x]^{k+1} := R \cdot ([b]-[A]\cdot\tilde{x}) + \{I - R\cdot[A]\}\cdot[x]^k \tag{2.11}$$

for some $[x]^0 \in \mathbb{R}^n$. If $[x]^{k+1} \subsetneq \operatorname{int}([x]^k)$ then (2.6) is satisfied for $[x] := [x]^k$. However, it turns out very quickly that in trivial examples $[x]^{k+1} \subsetneq \operatorname{int}([x]^k)$ will never be satisfied. For example it is frequently a good test for an algorithm to insert the exact solution. In our case omitting the uncertainties in the data that means $[A] = A \in \mathbb{R}^{n\times n}$, $[b] = b \in \mathbb{R}^n$, $\tilde{x} = A^{-1}b$ and $[x]^0 := 0$. Obviously we have to ensure that the $[x]^k$ always maintains a finite diameter to allow $[x]^{k+1} \subsetneq \operatorname{int}([x]^k)$. Therefore in [RU80] the so-called ε-inflation $[x] \circ \varepsilon$ was introduced. There are many possible definitions resp. practical incarnations of it. We use

$$[x]\circ\varepsilon := [x]\cdot[1-\varepsilon, 1+\varepsilon] + [-\mu, +\mu] \quad \text{for some } 0 < \varepsilon, \mu;\ \varepsilon \in \mathbb{R}, \mu \in \mathbb{R}^n \tag{2.12}$$

and define for given $[A] \in \mathbb{IIR}^{nn}$, $[b] \in \mathbb{IIR}^n$, $\tilde{x} \in \mathbb{R}^n$, $R \in \mathbb{R}^{nn}$ and $[x]^0 \in \mathbb{IIR}^n$ the following iteration:

$$\begin{aligned} &[y] := [x]^k \circ \varepsilon; \\ &[x]^{k+1} := R\cdot([b]-[A]\cdot\tilde{x}) + \{I - R\cdot[A]\}\cdot[y]. \end{aligned} \tag{2.13}$$

One immediately verifies that $[x]^{k+1} \subsetneq \operatorname{int}([y])$ implies (2.4) for $[x] := [y]$ and hence the assertions of theorem 2.1 are valid. Using the iteration schema (2.13) allows a complete analysis of the conditions under which (2.13) stops with $[x]^{k+1} \subsetneq \operatorname{int}([y])$ for some $k \in \mathbb{N}$. Without proof we state:

Theorem 2.2.: Let $[A] \in \mathbb{IIR}^{nn}$, $[b] \in \mathbb{IIR}^n$ be given and let $\tilde{x} \in \mathbb{R}^n$, $R \in \mathbb{R}^{nn}$ and $[x]^0 \in \mathbb{IIR}^n$. Then the following is equivalent:

(i) Using (2.13) with (2.12) for $0 < \varepsilon, \mu$ with $\varepsilon \in \mathbb{R}$, $\mu \in \mathbb{R}^n$ there is some $k \in \mathbb{N}$ with $[x]^{k+1} \subsetneq \operatorname{int}([y])$ hence implying the non-singularity of R and every $A \in [A]$ and $A^{-1}b \in \tilde{x} + [x]^{k+1}$ for all $A \in [A]$, $b \in [b]$

(ii) $\rho(|I-R\cdot[A]|) < 1$.

Remark: For $[C] \in \mathbb{IIR}^{nn}$ the absolute value is defined to be the matrix of absolute values of the components of $[C]$ where the absolute value of an interval $[x] \in \mathbb{IIR}$ is defined by $|[x]| := \max\{|x| \mid x \in [x]\}$.

It is well-known that the conventional residual iteration

$$x^{k+1} := x^k + R\cdot(b - Ax^k)$$

for a linear system $Ax = b$, $A \in \mathbb{R}^{nn}$, $b \in \mathbb{R}^n$ converges for every starting value $x^0 \in \mathbb{R}^n$ iff $\rho(I-RA) < 1$. However in very many practical examples not a single case was observed where $\rho(I-RA) < 1$ but $\rho(|I-RA|) \geq 1$. That means in practice, roughly speaking, an inclusion of the solution will be obtained if the real residual iteration converges. Using floating-point arithmetic it is sometimes hard to decide whether a residual iteration actually converges or not.

The remaining question is the quality of an inclusion $\tilde{x} + [x]$ for the solution set $X([A],[b])$. The quality is determined by the distance of $\tilde{x} + [x]$ to the interval hull of the solution set

$$[X([A],[b])] := \cap\{[w] \in \mathbb{IIR}^n \mid X([A],[b]) \subseteq [w]\}.$$

This distance can be estimated if some interval vector contained in $[X([A],[b])]$ could be found. Such an interval vector can also serve to bound the sensitivity of the solution of some $Ax = b$ for $A \in [A]$, $b \in [b]$ w.r.t. perturbations of A and b within $[A]$, $[b]$. In [RU90] the following theorem is shown.

Theorem 2.3.: Let $[A] \in \mathbb{I}\mathbb{R}^{nn}$, $[b] \in \mathbb{I}\mathbb{R}^{n}$ and $\tilde{x} \in \mathbb{R}^{n}$, $R \in \mathbb{R}^{nn}$, $[x] \in \mathbb{I}\mathbb{R}^{n}$ be given with

$$[z] := R \cdot ([b] - [A] \cdot \tilde{x}),\ [\Delta] := \{I - R \cdot [A]\} \cdot [x] \text{ and}$$
$$[z] + [\Delta] \subseteq \operatorname{int}([x]). \tag{2.14}$$

Then $[X([A], [b])]$ satisfies

$$[z] \mp [\Delta] \subseteq [X[A], [b])] - \tilde{x} \subseteq [z] + [\Delta]. \tag{2.15}$$

Remark: For interval vectors $[z], [\Delta] \in \mathbb{I}\mathbb{R}^{n}$

$$[z] \mp [\Delta] := [\inf([z]) + \sup([\Delta]),\ \sup([z]) + \inf([\Delta])]. \tag{2.16}$$

Let us demonstrate theorem 2.3 by means of our first example (2.2). We first compute the midpoints mid([A]), mid([b]) of [A], [b] to

$$\operatorname{mid}([A]) = \begin{bmatrix} 0.73 & -0.76 \\ -2.8 & 0.86 \end{bmatrix}, \operatorname{mid}([b]) = \begin{bmatrix} 0.3 \\ -2.7 \end{bmatrix}.$$

An approximate inverse R (three significant figures) of mid ([A]) and an approximate solution $\tilde{x}$ of mid $([A]) \cdot x = \operatorname{mid}([b])$ computes to

$$R = \begin{bmatrix} 0.312 & -0.276 \\ 1.016 & -0.265 \end{bmatrix} \text{ and } \tilde{x} = \begin{bmatrix} 0.838 \\ -0.410 \end{bmatrix}.$$

Note that no *a priori* information is required on the quality of R or $\tilde{x}$, even the non-singularity of R or matrices $A \in [A]$ is not required but verified by theorem 2.1. According to (2.14) we get

$$[z] = \begin{bmatrix} \pm 0.01289 \\ \pm 0.02164 \end{bmatrix} \text{ and } [C] := I - R \cdot [A] = \begin{bmatrix} \pm 1.49 \cdot 10^{-3} & \pm 5.88 \cdot 10^{-4} \\ \pm 3.58 \cdot 10^{-3} & \pm 1.29 \cdot 10^{-3} \end{bmatrix}.$$

The sizes of the components of [z] and [C] are typical. [z] is a correction of the size of one step of the residual iteration whereas [C] is the residual of R w.r.t. matrices $A \in [A]$. We set

$$[x] := [z] \cdot [0.9, 1.1] = \begin{bmatrix} \pm 0.014179 \\ \pm 0.023804 \end{bmatrix}$$

and verify

$$[z] + [\Delta] = \begin{bmatrix} \pm 0.012918 \\ \pm 0.021714 \end{bmatrix} \subseteq \operatorname{int}([x])$$
$$\text{with } [\Delta] = \begin{bmatrix} \pm 3.058 \cdot 10^{-5} \\ \pm 8.119 \cdot 10^{-5} \end{bmatrix}$$

The values of $[\Delta]$ are also typical as long as rad([A]) is not too big (e.g. every matrix $A \in [A]$ must be non-singular). Thus the assumptions of theorem 2.1 are valid implying the non-singularity of R and of every matrix $A \in \mathbb{R}^{nn}$ with $A \in [A]$. Moreover (2.15) gives

$$\begin{bmatrix} \pm 0.17008 \\ \pm 0.21281 \end{bmatrix} \subseteq [X([A], [b])] - \tilde{x} \subseteq \begin{bmatrix} \pm 0.17015 \\ \pm 0.21298 \end{bmatrix}.$$

We define the relative variation of the solution set $X([A], [b])$ by the vector σ with components

$$\sigma_i := \left\{ \frac{|\hat{x}_i - x_i|}{|\hat{x}_i|} \mid Ax = b \text{ for } A \in [A], b \in [b] \text{ and } \operatorname{mid}([A]) \cdot \hat{x} = \operatorname{mid}([b]) \right\},$$

provided $\hat{x}_i \neq 0$. That means we are looking for relative variations w.r.t. the solution of the „midpoint-system" $\operatorname{mid}([A]) \cdot x = \operatorname{mid}([b])$. σ is a vector giving the sensitivity for every component of the solution set and can be estimated by theorems 2.1 and 2.3

$$\frac{\operatorname{rad}([z] \mp [\Delta])_i}{|\hat{x}_i|} \leq \sigma_i \leq \frac{\operatorname{rad}([z] + [\Delta])_i}{|\hat{x}_i|}.$$

In our example we get

$$1.53\% \leq \sigma_1 \leq 1.54\%$$
$$5.25\% \leq \sigma_2 \leq 5.29\%.$$

That means there exists a linear system $Ax = b$ with $A \in [A]$, $b \in [b]$ such that e.g. the relative distance of the first component of $A^{-1}b$ to the first component of $\hat{x} = (\operatorname{mid}([A]))^{-1} \cdot \operatorname{mid}([b])$ is equal to a value between 1.53% and 1.54%. This looks like

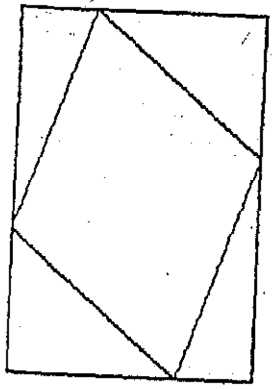


Fig. 2.2. Solution complex of (2.2) with inclusion

According to the scale we do not see a difference between $[z] \mp [\Delta]$ and $[z] + [\Delta]$. Amplifying the scale by a factor 100 we get for the left-most boundary of $X([A], [b])$ the following graph.

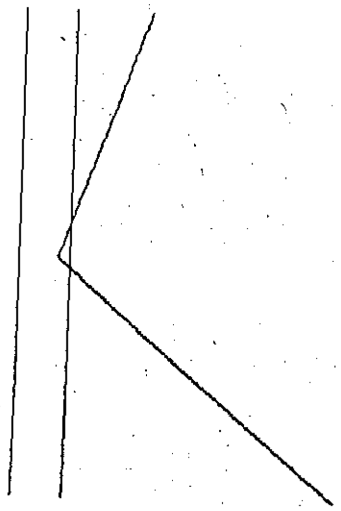


Fig. 2.3. Detail of figure 2.2, scale 1: 100

The left-most two vertical lines are the boundaries of $[z] + [\Delta]$ and $[z] \mp [\Delta]$ where the vertex between them belongs to $X([A], [b])$.

For smaller diameters of $[A]$ and $[b]$ the difference between $[z] \mp [\Delta]$ and $[z] + [x]$ becomes even smaller. This depends only on the radius of $[\Delta] = \{I - R \cdot [A]\} \cdot$

$[x]$. But $I - R \cdot [A]$ is the residual of $R \approx A^{-1}$ for some $A \in [A]$ and $[x]$ is an inclusion of the error of $\tilde{x}$. Both quantities are usually small and the product of them is very small, the latter determining the distance between the interior $[z] \mp [\Delta]$ and the exterior $[z] + [\Delta]$ of $X([A], [b])$ and therefore the quality and sharpness of the solution.

3 Implementation on Digital Computers

For a practical implementation of the methods described in sections 1 and 2 first of all we need a floating-point interval arithmetic. The central property to be preserved is the isotonicity (0.5). For this purpose we may use the IEEE 754 arithmetic standard [IEEE85] which is nowadays implemented in a number of digital computers, in personal computers frequently by means of a coprocessor or a software emulation. It provides so-called directed rounding modes which we call ∇ for the rounding downward and Δ for the rounding upward. In the ∇-mode the computed floating-point result is always less than or equal to the true real result of an operation, in the Δ-mode it is greater than or equal to the true real result.

Denote the set of floating-point numbers in a digital computer by F. Then $|F| < \infty$ and $F \subseteq \mathbb{R}$. Following [KUL80] a floating-point operation $* \in \{+, -, \cdot, /\}$ in the ∇-mode resp. Δ-mode is denoted by ∇ resp. Δ. Therefore

$$\nabla : F \times F \to F \text{ with } a, b \in F : a \nabla b \leq a * b$$
$$\Delta : F \times F \to F \text{ with } a, b \in F : a \Delta b \leq a * b$$

for $* \in \{+, -, \cdot, /\}$. Thus defining a floating-point interval $[a]$ by $[a] := [\underline{a}, \overline{a}] := \{x \in \mathbb{R} \mid \underline{a} \leq x \leq \overline{a}\}$ where $\underline{a}, \overline{a} \in F$ and floating-point interval operations $*$ by

$$[a] + [b] = [\underline{a} \nabla \underline{b}, \overline{a} \Delta \overline{b}]$$
$$[a] - [b] = [\underline{a} \nabla \underline{b}, \overline{a} \Delta \underline{b}],$$
$$[a] \cdot [b] = [\mathrm{Min}\{\underline{a} \nabla \underline{b}, \underline{a} \nabla \overline{b}, \overline{a} \nabla \underline{b}, \overline{a} \nabla \overline{b}\}, \mathrm{Max}\{\underline{a} \Delta \underline{b}, \underline{a} \Delta \overline{b}, \overline{a} \Delta \underline{b}, \underline{a} \Delta \overline{b}\}]$$
$$[a] / [b] = [\underline{a}, \overline{a}] \cdot [1 \nabla \overline{b}, 1 \Delta \underline{b}]$$

the key property of interval arithmetic, the isotonicity, is preserved. Vector and matrix operations are defined similar to those in section 0 by replacing the interval operations for real intervals by the corresponding interval operations for floating-point intervals.

Hence evaluating (2.4) by replacing the interval operations by the corresponding floating-point interval operations assures the validity of (2.4) and therefore implies that all assertions of theorem 2.1 are true.

In [KUL76, KUL81] an interval arithmetic is defined with the additional feature of a scalar product operation. Kulisch uses a precise scalar product in the sense that for any scalar product of floating-point numbers resp. intervals the immediate left and right floating-point neighbour of the real result resp. of the lower and upper bound of the real result are calculated. This gives especially in the calculation of the residuals $[b]-[A]\cdot\tilde{x}$ and $I-R\cdot[A]$ sharper results namely in the case of data with small uncertainties.

In the following we give an algorithm for the calculation of guaranteed bounds of the solution set $X([A],[b])$ for linear systems $Ax=b$, $A\in[A]$, $b\in[b]$.

```
1.  calculate an approximation of $R \approx (\mathrm{mid}([A]))^{-1}$ using some standard
    algorithm;
2.  $\tilde{x} := R\cdot\mathrm{mid}([b])$;
    optionally apply a residual iteration to improve $\tilde{x}$;
3.  $[z] := R\cdot([b]-[A]\cdot\tilde{x})$; $[C] := I-R\cdot[A]$;
    $[x] := [z]$; $k := 0$;
4.  repeat   $[y] := [x]\cdot[0.9, 1.1] + [-\mu, +\mu]$; $k := k+1$;
             $[x] := [z] + [C]\cdot[y]$
    until    $[x] \subseteq \mathrm{int}([y])$ or $k > k\mathrm{max}$
5.  if       $[x] \subseteq \mathrm{int}([y])$
             then   {all $A\in[A]$ are non-singular and $X([A],[b]) \subseteq \tilde{x}+[x]$}
             else   {no inclusion computed};
```

Algorithm 3.1.: Solving linear systems for uncertain data

The operations in steps 1) and 2) are ordinary floating-point operations whereas the operations in step 3) to 5) are floating-point interval operations. The μ in step 4) is the smallest positive floating-point number. A proper value for kmax is for example 5. In general only one or two iterations in (4) are necessary.

Algorithms similar to 3.1 for many numerical standard problems have been implemented in a number of programming packages, both being commercially and non-commercially available. There is the commercial package ACRITH by IBM (cf. [IBM84], [IBM86a], [IBM86b]) in 3 releases available under VM operating system being supported by interval arithmetic described in [KUL81] in hardware on 4361and 9370 engines and software supported otherwise. ARITHMOS by SIEMENS (cf. [SIE86]) is also hardware supported on many small and medium size computers and software supported otherwise. There are a number of non-commercial packages like Abacus, Calculus, Fortran-SC, Hificomp, Pascal-SC, TPX (cf. [HUS88], [HUS89b], [RU89], [BLE87], [MAR89], [KUL87], [HUS89a]) and others. The first and second one is an interactive programming environment supporting all interval operations and including many high-level problem solving routines for numerical problems with uncertain data. It runs on PC's and IBM /370 under VM operating system.

4 Numerical Examples

In this section we discuss some numerical examples demonstrating our earlier analysis. The examples are executed on a machine with base 16 and machine unit eps $:= 16^{-13} = 0.22\ldots 10^{-16}$. We display in the following seven figures of the calculated values. All calculated values are guaranteed in the above sense.

The following coefficients

$$A = \left[\begin{array}{cccccccc|cccccccc}
-7 & 7 & 13 & -8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -16 & -2 & 4 & -17 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
-2 & 1 & -3 & 14 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & -2 & 1 & -3 & 14 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
\hline
1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0
\end{array}\right]$$

$c = (6.45, 6.45, 6.45, 6.45, 5.91, 5.91, 5.91, 5.91, 4.83, 4.83, 4.83, 4.83, 0, 0, 0, 0)^t$

$b = (0, 0, 0, 0, 3814, 2666, 4016, 1300)^t$

are the input data of the standard form for a small blending problem ([CHV83], p. 10). The unique optimal vertices of (1.1), (1.2) are

$$x(B_1) = (3754.000, 2666.000, 920.0000, 543.0000, 60.00000, 0, 3096.000, 672.0000, 0, 0, 0, 85.00000, 0, 0, 0, 0)^t$$

$$y(B_1) = (0.06000000, 0.06000000, 0.1500000, 7.170000, 5.880000, 6.120000, 4.830000)^t$$

and the optimal value $z_{\mathrm{opt}} = 73879.38$.

All components of b, c and the coefficients a_{ij} with $i = 1, \ldots, 4$, $j = 1, \ldots, 8$ were replaced by

$$b_i = b_i \pm (|b_i|/200),\ c_j = c_j \pm (|c_j|/200),$$
$$a_{ij} = a_{ij} \pm (|a_{ij}|/200).$$

Hence the diameter of the intervals is 1% of the midpoint. This interval linear programming problem $[P]$ was solved by the method of section 1. The algorithm described over there stopped with step (2). Three basic-index-sets $Z = \{B^1, B^2, B^3\}$ were calculated with

$$B^1 = \{1, 2, 3, 4, 5, 7, 8, 12\}$$
$$B^2 = \{1, 2, 3, 4, 6, 7, 8, 14\}$$
$$B^3 = \{1, 3, 4, 6, 7, 8, 12, 14\}.$$

In Table 4.1 the corresponding inclusion vectors $[x(B)]$ with $B \in Z$ (compare (1.13)) are displayed. For every component of these interval vectors the outer bounds (compare theorem 2.1) and the inner bounds (compare theorem 2.3) of the solution set $X([A], [b])$ are given.

The results are displayed in the following way. The first component of $[x(B_1)]$ has outer bounds

$$[3719.765, 3788.234]$$

and inner bounds

$$[3720.851, 3787.149]$$

and this is displayed in table 4.1 as

$$\left[37^{19.765}_{20.851}, 378^{8.234}_{7.149}\right].$$

Note that the difference of outer and inner bounds is small compared to the diameter of the solution set $X([A], [b])$ although the diameter of the interval input data is 1%. Hence the outer bounds alone describe the corresponding solution sets very well. The bounds $[y(B)]$ for the solution of the dual problem behave similar. Moreover $z_{opt}(P) \in [72771.55, 75310.79]$ for all $P \in [P]$. The optimal value varies at most by 3.4% if the input data vary 1%. The calculated intervals perform a rigorous sensitivity analysis of a linear programming problem. Several further applications of the method described in section 1 can be found in ([JAN85], [JAN88], [MAI88]).

Table 4.1. Inner and outer bounds for the linear programming problem

$[x(B_1)]$	$[x(B_2)]$	$[x(B_3)]$
$[37^{19.765}_{20.851}, 378^{8.234}_{7.149}]$	$[3794.92^{8}_{9}, 3833.07^{1}_{0}]$	$[3794.92^{8}_{9}, 3833.07^{1}_{0}]$
$[2652.67^{0}_{1}, 2679.33^{1}_{0}]$	$[1^{370.559}_{415.334}, 1^{834.774}_{789.998}]$	0
$[85^{3.0805}_{5.6083}, 98^{6.9195}_{4.3916}]$	$[1^{493.169}_{526.002}, 18^{60.164}_{27.332}]$	$[269^{7.654}_{8.979}, 280^{6.093}_{4.767}]$
$[51^{5.2013}_{6.0831}, 5^{70.7987}_{69.5803}]$	$[7^{23.9173}_{35.1909}, 8^{55.4160}_{44.1424}]$	$[110^{8.617}_{9.341}, 11^{60.471}_{59.747}]$
$[4^{4.83516}_{5.92064}, 7^{5.16483}_{4.07936}]$	0	0
0	$[8^{38.3265}_{83.1013}, 12^{88.341}_{43.565}]$	$[2652.6^{69}_{70}, 2679.33^{1}_{0}]$
$[30^{09.000}_{11.529}, 318^{2.999}_{0.471}]$	$[21^{45.140}_{77.973}, 25^{33.526}_{00.694}]$	$[11^{89.827}_{91.153}, 133^{8.426}_{7.101}]$
$[64^{8.0945}_{9.0071}, 69^{5.9054}_{4.9929}]$	$[3^{66.0404}_{77.2330}, 4^{84.6263}_{73.4337}]$	$[6^{0.65665}_{1.34069}, 1^{00.2548}_{99.5707}]$
0	0	0
0	0	0
0	0	0
$[5^{5.92167}_{6.92991}, 11^{4.0784}_{3.0700}]$	$[5^{5.15242}_{7.67319}, 11^{4.8476}_{2.3268}]$	$[56.^{01120}_{85114}, 113^{.9888}_{.1488}]$
0	0	0
0	0	$[1^{493.805}_{508.293}, 17^{92.676}_{78.188}]$
0	0	0
0	0	0

The determination of the input data is a very hard problem for many practical applications. The above algorithm may be used for example to get guaranteed informations how the precision of the input data should be to ensure that the optimal value varies between given bounds. This can be done in an interactive manner by varying some or all input data.

The above small-size example should only demonstrate how the method works. In the following we will give some results for systems of linear equations. We do so because the bounds for linear programming problems are the calculated bounds of systems of linear equations (compare (1.11)).

In the following tests the coefficients of the $n \times n$ matrix A are uniformly distributed in $[-1,1]$. We choose the right hand side b such that the components of the exact solution are all equal to one. The corresponding interval matrix $[A]$ and the interval right hand side are defined by

$$[A] = A \pm 10^{-7} \cdot |A|,\ [b] = b \pm 10^{-7} \cdot |b|.$$

In the following table we display the maximal inner bound

$$\underline{\sigma} := \text{Max}\left\{\frac{\text{rad}([z] \mathbin{\overset{-}{+}} [\Delta])_i}{|\hat{x}_i|} \mid i = 1, \dots, n\right\}$$

and the maximal outer bound

$$\overline{\sigma} := \text{Max}\left\{\frac{\text{rad}([z] + [\Delta])_i}{|\hat{x}_i|} \mid i = 1, \dots, n\right\}.$$

of the solution set $X([A], [b])$.

Table 4.2. Inner aund outer bounds of the solution set $X([A], [b]) \subseteq \mathbb{R}^n$

n	$\underline{\sigma}$	$\overline{\sigma}$
10	$5.529790 \cdot 10^{-5}$	$5.529875 \cdot 10^{-5}$
50	$1.197947 \cdot 10^{-3}$	$1.198098 \cdot 10^{-3}$
100	$3.374497 \cdot 10^{-3}$	$3.376395 \cdot 10^{-3}$
200	$5.653805 \cdot 10^{-2}$	$5.681218 \cdot 10^{-2}$
300	$2.729809 \cdot 10^{-2}$	$2.734967 \cdot 10^{-2}$
500	$5.139332 \cdot 10^{-2}$	$5.159357 \cdot 10^{-2}$

This test demonstrates that $\underline{\sigma}, \overline{\sigma}$ are very close. The outer bounds alone give the exact number of significant figures of the solution set. For other uncertainties in the input data the results look very similar.

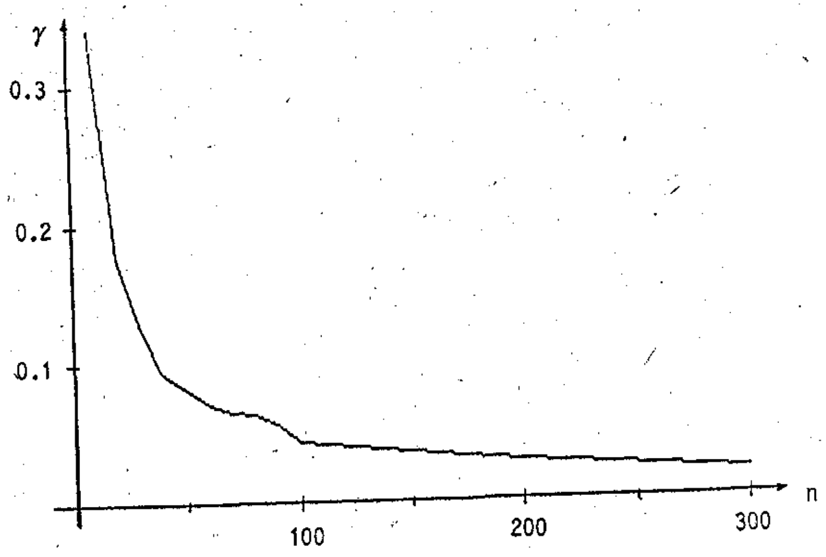
A frequently used heuristic approach to error and sensitivity analysis is to run a calculation several times with varying input data. The number of figures of the solution which agree in all calculations are taken to indicate the precision resp. width of the solution set. For example Bellmann [BEL61] writes: „Considering the many assumptions that go into the construction of mathematical models, the many uncertainties that are always present, we must view with some suspicion at any particular prediction. One way to contain confidence is to test the consequences of various changes in the basic parameters“.

For dimension n we have randomly chosen n matrices $A \in [A]$ and right hand sides $b \in [b]$, being uniformly distributed within $[A]$ resp. $[b]$, where $[A]$, $[b]$ are defined as in the above example. The measure

$$\gamma := \text{Max}\left\{\frac{rad([v_i])}{rad([z]+[\Delta])_i} \mid i = 1, \dots, n\right\}$$

($[v]$ is the smallest interval vector containing the n solutions for the n runs) describes the underestimation of this Monte-Carlo experiment w.r.t. the true solution set $X([A], [b])$, compare table 4.3.

Table 4.3. True span of the solution set compared to Monte Carlo span



Obviously, with growing dimension there is a severe underestimation of the solution set $X([A], [b])$ using the Monte Carlo method. For example, for $n = 300$ for the Monte Carlo estimation of the radius of the solution set is only 3% of the true radius. We agree with the opinion of Bellmann. If, on the other hand, methods are available being faster and calculating guaranteed bounds, those may be used.

More applications and examples can be found in [RU80], [RU83]. The described theory and the algorithms can be used on the one hand to obtain a rigorous sensitivity analysis for problems with interval input data. On the other hand they produce high accurate solutions for problems with real input data. The algorithms in ACRITH are based on the method described in section 2. On a computer using the optimal arithmetic of Kulisch [KUL76], [KUL81] with only about 16 significant decimals for all intermediate results a 21 x 21 Hilbert matrix was accurately inverted. The condition number of this matrix is about 10^{29}.

Acknowledgement

The authors wish to thank Prof. H. Brakhage and Prof. U. Kulisch for initiating this research.

References

[ALE74] Alefeld G, Herzberger J (1974) Einführung in die Intervallrechnung, B.I. Wissenschaftsverlag

[ALE83] Alefeld G, Herzberger J (1983) Introduction to Interval Computations, Academic Press

[BAU87] Bauch H, Jahn KU, Oelschlägel D, Süsse H, Wiebigke V (1987) Intervallmathematik, Theorie und Anwendungen, Mathematische-Naturwissenschaftliche Bibliothek, Band 72, B.G. Teubner, Leipzig

[BEE78] Beeck H (1978) Linear Programming with Inexact Data, Bericht Nr. 7830 der Abteilung Mathematik TU München

[BEL61] Bellmann R (1961) Adaptive Control Processes, Princeton University Press

[BLE87] Bleher JH Rump SM, Kulisch U, Metzger M, Ullrich Ch, Walter W (1987) FORTRAN-SC: A Study of a FORTRAN Extension for Engineering/Scientific Computation with Access to ACRITH, Computing 39: 93–110, Springer

[BOH81] Bohlender G, Kaucher E, Klatte R, Kulisch U, Miranker WL, Ullrich Ch, Wolff v Gudenberg J (1981) FORTRAN for Contemporary Numerical Computation, IBM Research Report RC 8348 (1980). Computing 26: 277–314

[CHV83] Chvatal V (1983) Linear Programming, WH Freemann and Company

[DEI86] Deif A (1986) Sensitivity Analysis in Linear Systems, Springer, New York

[HUS88] Husung D (1988) – ABACUS – Programmierwerkzeug mit hochgenauer Arithmetik für Algorithmen mit verifizierten Ergebnissen, Diplomarbeit, Karlsruhe

[HUS89a] Husung D (1989) Precompiler for Scientific Computation (TPX), Informatik III, TU Hamburg-Harburg

[HUS89b] Husung D, Rump SM (1989) ABACUS, Proceedings SCAN'89, Tagung „Wissenschaftliches Rechnen und Programmiersprachen", Basel

[IBM84] IBM System/370 RPQ High-Accuracy Arithmetic (1984) SA 22–7093–0

[IBM86a] IBM High-Accuracy Arithmetic Subroutine Library (ACRITH) (April 1986) General Information Manual, GC 33–6163–02, 3rd Edition

[IBM86b] IBM High-Accuracy Arithmetic Subroutine Library (ACRITH), Program Description and User's Guide (April 1986) SC 33–6164–02, 3rd Edition

[IEEE85] IEEE Standard for Binary Floating-Point Arithmetic (1985), ANSI/IEEE Standard 754

[JAN85] Jansson C (1985) Zur linearen Optimierung mit unscharfen Daten, Dissertation, Kaiserslautern

[JAN88] Jansson C (1988) A Selv-Validating Method for Solving Linear Programming Problems, Computing, Suppl. 6: 33–46

[KRA75] Krawzcyk R (1975) Fehlerabschätzung bei linearer Optimierung, Interval Mathematics: 215–227, Springer Lecture Notes in Computer Science 29

[KUL76] Kulisch U W (1976) Grundlagen des numerischen Rechnens, B.I. Wissenschaftsverlag

[KUL81] Kulisch UW (1981) Computer Arithmetic in Theory and Practice, Academic Press, New York

[KUL83] Kulisch UW, Miranker WL (1983) A New Approach to Scientific Computation, Academic Press New York

[KUL87] Kulisch U (1987) (ed.): PASCAL-SC: A PASCAL Extension for Scientific Computation, Information Manual and Floppy Disks, Version IBM PC. Stuttgart, B.G. Teubner; Chichester: John Wiley & Sons

[MAC70] Machost B (1970) Numerische Behandlung des Simplexverfahrens mit intervallmathematischen Methoden, Berichte der GMD Bonn, Nr. 30

[MAI88] Maichle U (1988) Lineare Intervalloptimierung und Anwendungen, Diplomarbeit, Kaiserslautern

[MAR89] Markov S, Velichkov T (1989) HIFICOMP, a subroutine library for highly efficient and accurate computation, Proceedings SCAN'89, Tagung „Wissenschaftliches Rechnen und Programmiersprachen", Basel

[MOO79] Moore RE (1979) Methods and Applications of Interval Analysis, SIAM Philadelphia

[NEU90] Neumaier A (1990) Interval Methods for Systems of Equations, Cambridge University Press, to be published.

[PAP82] Papadimitrion CH, Steiglitz K (1982) Combinatorial Optimization: Algorithms and Complexity, Prentice-Hall

[ROH84] Rohn J (1984) Solving Interval Linear Systems, Freiburger Intervallberichte 84/7: 1–30

[RU80] Rump SM (1980) Kleine Fehlerschranken bei Matrixproblemen, Dr.-Dissertation, Institut für Angewandte Mathematik, Universität Karlsruhe

[RU83] Rump SM (1983) Solving Algebraic Problems with High Accuracy, Habilitationsschrift, in: A New Approach to Scientific Computation, Hrsg UW Kulisch und WL Miranker, ACADEMIC PRESS: 51–120

[RU89] Rump SM(1989) CALCULUS, in „Wissenschaftliches Rechnen mit Ergebnisverifikation", ed U Kulisch, Vieweg und Akademie Verlag Berlin

[RU90] Rump SM (1990) Rigorous Sensitivity Analysis for Systems of Linear and Nonlinear Equations, to appear in MATH of COMP

[SIE86] Siemens ARITHMOS, (BS 2000) Benutzerhandbuch (1986) SIEMENS AG, U2900–I-Z87–1

[STEU81] Steuer RE (1981) Algorithms for Linear Programming Problems with Interval Objective Function Coefficients, Mathematics of Oper. Res., Vol. 6: 222–248

Received December 1989