A Simplex-Like Algorithm for Interval Linear Systems

Olivier Beaumont

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

A Simplex-Like Algorithm for Interval Linear Systems.

O. Beaumont

N ° 3153

Avril 1997

THÈME 4

ISSN 0249-6399

INRIA
RENNES

A Simplex-Like Algorithm for Interval Linear Systems.

O. Beaumont*

Thème 4 — Simulation et optimisation
de systèmes complexes
Projet Aladin

Rapport de recherche n°3153 — Avril 1997 — 18 pages

Abstract: In this paper, we show how it is possible to use convex polyhedra for solving linear interval systems without using preconditionning. We first show how to derive, from an enclosure of $\square\Sigma([A],[b])$, a polyhedron which contains the convex hull of the solution set. Then, a simplex-like method enables us to find a new outer inclusion. Moreover, the constraints obtained may be used to compute an inner inclusion of $\square\Sigma([A],[b])$.

Key-words: interval linear systems, simplex algorithm

(Résumé : tsvp)

* Institut de Recherche en Informatique et Systèmes Aléatoires - email:obeaumon@irisa.fr

Unité de recherche INRIA Rennes
IRISA, Campus universitaire de Beaulieu, 35042 RENNES Cedex (France)
Téléphone : (33) 02 99 84 71 00 – Télécopie : (33) 02 99 84 71 71

Un algorithme de type Simplexe pour résoudre les systèmes linéaires avec intervalles.

Résumé : Dans cet article, nous montrons comment il est possible d'utiliser des polyhèdres convexes pour résoudre les systèmes linéaires d'intervalles sans utiliser de préconditionnement. Nous montrons tout d'abord comment trouver, à partir d'un sur-ensemble de $\square\Sigma([A],[b])$, un polyhèdre qui contient l'ensemble solution. Nous utilisons alors un algorithme de type simplexe pour déterminer un nouveau sur-ensemble de la solution. De plus, les contraintes obtenues peuvent être utilisées pour determiner un sous-ensemble de $\square\Sigma([A],[b])$.

Mots-clé : systèmes linéaires d'intervalles, algorithme du simplexe

1 Introduction

A lot of work has been done in order to solve interval linear systems $[A]x = [b]$ [6]. Rohn and Kreinovich [10],[4] has proved that the calculation of $\Box\Sigma([A],[b])$, the smallest box that contains all the solutions of $[A]x = [b]$, is a NP-hard problem. On the other hand, several algorithms obtain good results, especially when the diameter of $[A]$ is small [8],[6].
In this paper, we propose a new algorithm which is based on linear programming.It consists of an iterative scheme which considers as input an enclosure of the solution of $\Box\Sigma([A],[b])$ and returns a (usually better) enclosure.
This algorithm converges toward a superset of the convex hull of the united solution set $\Sigma([A],[b])$.
In this paper, we use the following notations:

$$\Sigma([A],[b]) = \{x,\ \exists A \in [A],\ \exists b \in [b],\ Ax = b\}$$

$$\Gamma([A],[b]) = Co(\Sigma([A],[b])),\ \text{where } Co \text{ denotes the convex hull}$$

We also denote by $\Box\Sigma([A],[b])$ the interval hull of the solution.
The interval linear systems we consider in this paper are defined by the following notations:

$$[A] = [A - \Delta A, A + \Delta A]$$

$$[b] = [\underline{b}, \bar{b}] = [b - \Delta b, b + \Delta b],\ \Delta b = \frac{\bar{b} - \underline{b}}{2}$$

2 How to find a polyhedron that contains the convex hull of the united solution set

It is known that the solution of the problem $[A]x = [b]$ is in general not convex[6]. As far as we are only interested in $\Box\Sigma([A],[b])$, we may use $\Gamma([A],[b])$ as intermediate set. Oettli and Prager [7] proved the following theorem:

Theorem 1

$$(\ \ \exists A \in [A],\ \exists b \in [b],\ Ax = b\ \ ) \Leftrightarrow |Ax - b| \leq \Delta A|x| + \Delta b$$

This expression does not lead directly to a polyhedron, because of the absolute value, which underlines the fact that the solution set is usually not a convex set [6].
Let us assume that we know an enclosure of $\Box\Sigma([A],[b])$, which may be obtained, for instance, by the algorithm proposed by Rump [8]. We give a first result which provides a way to get rid of the absolute values.

Lemma 1 *If*

$$\Box\Sigma([A],[b]) \subset [\underline{x}, \bar{x}]$$

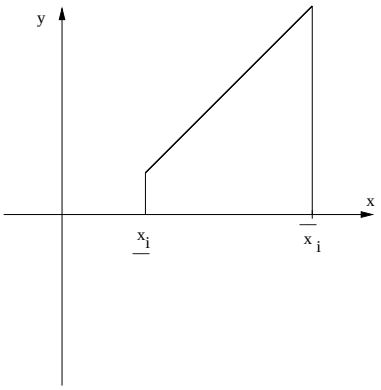

Figure 1: Situation when both $\underline{x}_i$ and $\bar{x}_i$ are positive

and, if

$$\alpha_j = \frac{|\bar{x}_j| - |\underline{x}_j|}{\bar{x}_j - \underline{x}_j} \text{ and } \beta_j = \frac{\bar{x}_j|\underline{x}_j| - \underline{x}_j|\bar{x}_j|}{\bar{x}_j - \underline{x}_j}$$

where x_j denotes the $j-th$ component of x, we have:

$$\forall x_j \in [\underline{x}_j, \bar{x}_j], |x_j| \leq \alpha_j x_j + \beta_j$$

.

Proof:

- First case: $\bar{x}_j \leq 0$, then $\alpha_j = -1, \beta_j = 0$, and $\alpha_j x + \beta_j = -x = |x|$.
- Second case: $\underline{x}_j \geq 0$, then $\alpha_j = 1, \beta_j = 0$, and $\alpha_j x + \beta_j = x = |x|$.
- Third case: $\bar{x}_j \underline{x}_j \leq 0$, then $\alpha_j = \frac{\bar{x}_j + \underline{x}_j}{\bar{x}_j - \underline{x}_j}, \beta_j = \frac{-2\bar{x}_j \underline{x}_j}{\bar{x}_j - \underline{x}_j}$.
  - if $0 \leq z \leq \bar{x}_j$, then $\alpha_j z + \beta_j - |z| = \frac{-2\underline{x}_j}{\bar{x}_j - \underline{x}_j} * (\bar{x}_j - z) \geq 0$. thus, $|z| \leq \alpha_j z + \beta_j$.
  - if $\underline{x}_j \leq z \leq 0$, then $\alpha_j z + \beta_j - |z| = \frac{2\bar{x}_j}{\bar{x}_j - \underline{x}_j} * (z - \underline{x}_j) \geq 0$. thus, $|z| \leq \alpha_j z + \beta_j$.□

In fact, several situations may occur:

Lemma 2 *The convex hull of the set $\mathcal{S} = \{(x,y) \in \mathbb{R}^2,\ x \in [\underline{x}_j, \bar{x}_j],\ y \leq |x|\}$ is the polyhedron defined by $\mathcal{P} = \{(x,y),\ x \in [\underline{x}_j, \bar{x}_j], 0 \leq y \leq \alpha_j x + \beta_j\}$.*

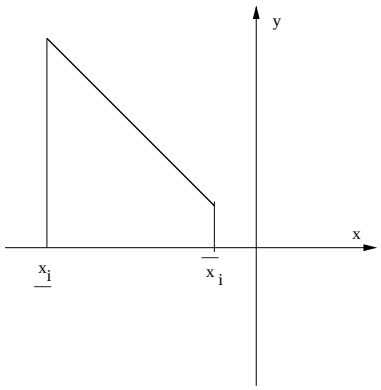

Figure 2: Situation when both $\underline{x}_i$ and $\bar{x}_i$ are negative

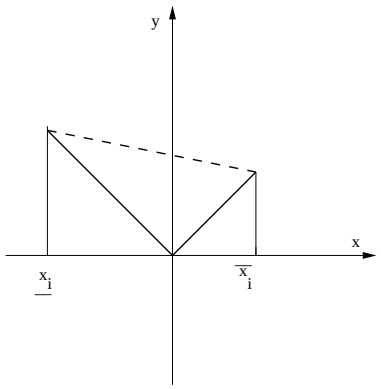

Figure 3: Situation when $\underline{x}_i \leq 0$ and $\bar{x}_i \geq 0$

Proof: If $\underline{x}_j \geq 0$ or $\bar{x}_j \leq 0$, then $\mathcal{S} = \mathcal{P}$ and the property is trivial.
We assume now that $\underline{x}_j \leq 0 \leq \bar{x}_j$.
The set $\mathcal{P}$, which is defined by linear inequalities, is convex, and the previous lemma implies $\mathcal{S} \subset \mathcal{P}$.
Therefore, $Co(\mathcal{S}) \subset \mathcal{P}$.

Reciprocally, let us consider $(a,b) \in \{(x,y),\ x \in [\underline{x}_j, \bar{x}_j],\ y \leq \alpha_j x + \beta_j\} \setminus \mathcal{S}$.
Let us consider now the line $y = \alpha_j x + (b - \alpha_j a)$, which intersects $y = x$ and $y = -x$ at the points $(a_1, b_1) = \frac{b-\alpha_j a}{1-\alpha_j}(1,1)$ and $(a_2, b_2) = \frac{b-\alpha_j a}{1+\alpha_j}(-1,1)$ respectively.
Since $b - \alpha_j a \leq \beta$, we obtain $\underline{x}_j \leq a_1 \leq a_2 \leq \bar{x}_j$ and, therefore, $(a_1, b_1) \in \mathcal{S}$, $(a_2, b_2) \in \mathcal{S}$ and $(a,b) \in Co(\mathcal{S})$.
We have proved that $Co(\mathcal{S}) = \mathcal{P}$.□

Therefore, we can expect that the following simplex $\Omega([A],[b],[\underline{x},\bar{x}])$ represents a good approximation of $\Gamma([A],[b])$. If we denote by D_α the diagonal matrix whose diagonal entries are the α_i's and β the vector of the β_i's then we define $\Omega([A],[b],[\underline{x},\bar{x}])$ as follows:

$$\Omega([A],[b],[\underline{x},\bar{x}]) \left\{ \begin{array}{l} Ax - \Delta A\ D_\alpha x \leq \bar{b} + \Delta A\ \beta \\ Ax + \Delta A\ D_\alpha x \geq \underline{b} - \Delta A\ \beta \end{array} \right.$$

We can now enunciate the theorem of this paper

Theorem 2 *Let $\square\Sigma([A],[b]) \subset [\underline{x},\bar{x}]$. Then, we have*

$\Sigma([A],[b]) \subset \{x\ ;\ Ax - \Delta A\ D_\alpha x \leq \bar{b} + \Delta A\ \beta\ ,\ Ax + \Delta A\ D_\alpha x \geq \underline{b} - \Delta A\ \beta\}$, *where*

$\alpha_j = \frac{|\bar{x}_j| - |\underline{x}_j|}{\bar{x}_j - \underline{x}_j}$ *and*
$\beta_j = \frac{\bar{x}_j |\underline{x}_j| - \underline{x}_j |\bar{x}_j|}{\bar{x}_j - \underline{x}_j}$.

Proof: This theorem is a direct consequence of Theorem 1 and Lemma 1. □

We have seen above how an enclosure of $\square\Sigma([A],[b])$ leads to a simplex that describes a superset of $\Gamma([A],[b])$. The set of problems $\max_{x \in \Omega([A],[b],[\underline{x},x])} x_i$ and $\min_{x \in \Omega([A],[b],[\underline{x},x])} x_i$ can be solved, for instance, by applying $2n$ times the simplex method [2]. Therefore, a new enclosure of $\square\Sigma([A],[b])$ is obtained, and an iterative scheme can be developed.
The limit of this iterative algorithm is usually a good enclosure of $\square\Sigma([A],[b])$ (not too large) as shown in the last section. Unfortunately, it is difficult to find a characterization of this limit in order to study its accuracy. The limit is usually not equal to $\square\Sigma([A],[b])$.
For instance, when we know an enclosure of $\square\Sigma([A],[b])$ in which each x_i keeps a constant sign, it is easy to prove that the algorithm described above provides the exact solution $\square\Sigma([A],[b])$ after only one step. Indeed, in this case, the sets $\{(x,y),\ x \in [\underline{x}_j, \bar{x}_j],\ y \leq$

$\alpha_j x + \beta_j\}$ and $\{(x,y),\ x \in [\underline{x_j}, \bar{x_j}],\ y \leq |x|\}$ are equal, and $\Omega([A],[b],[\underline{x},\bar{x}])$ is an exact representation of the solution set of $[A]x = [b]$, which is in this case convex.
In the following section, we present a modification of the algorithm presented above, in order to perform it in a reasonable amount of time.

3 Algorithm for the outer inclusion.

In the previous section, we have presented an iterative method which solves interval linear systems. This method requires the execution of $2n$ simplex algorithms during each step of the algorithm. Since an execution of the simplex algorithm requires roughly $O(n^3)$ flops [2] [12], the total amount of work per iteration is therefore of order $O(n^4)$. The algorithms proposed by Rump [8] and Neumaier [6] require an amount of work of order $5n^3$. We show in this section how to perform a step of the algorithm in time $O(n^3)$.
We propose a three-step algorithm in order to obtain an outer inclusion. The first step consists in solving an approximate problem. The conditions that define extremal points of the approximate problem are supposed to be close to those which define extremal points of the exact problem. The second step consists in using the results of the first step to solve the exact problem. The last step is a correction step, and leads to a proved outer inclusion.

3.1 First step

In order to apply general theorems of linear programming, we perform the following change of variables which induce use of positive variables. We set $y = x - \underline{x}$:

$$\Omega([A],[b],[\underline{x},\bar{x}]) \left\{ \begin{array}{l} Ay - \Delta A\ D_\alpha y \leq \bar{b} + \Delta A\ \beta - (A - \Delta A\ D_\alpha)\underline{x} = \bar{b}' \\ Ay + \Delta A\ D_\alpha y \geq \underline{b} - \Delta A\ \beta - (A + \Delta A\ D_\alpha)\underline{x} = \underline{b}' \\ y \geq 0 \end{array} \right.$$

The first step consists in solving the problem $Ay = [\underline{b'}, \bar{b'}]$.
The polyhedron Ω' which corresponds to the problem $Ay = [\underline{b'}, \bar{b'}]$ can be considered as a modification of the simplex which defines $\Omega([A],[b],[\underline{x},\bar{x}])$:

$$\Omega' \left\{ \begin{array}{l} Ay \leq \bar{b'} \\ Ay \geq \underline{b}' \\ y \geq 0 \end{array} \right. , \text{ and } \Omega([A],[b],[\underline{x},\bar{x}]) \left\{ \begin{array}{l} Ay - \Delta A\ D_\alpha y \leq \bar{b}' \\ Ay + \Delta A\ D_\alpha y \geq \underline{b}' \\ y \geq 0 \end{array} \right.$$

As long as ΔA is small compared to A, we may expect the following situation, obtained for: $A = \begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix}, \Delta A = \frac{1}{4}\begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}, b = \frac{5}{4}\begin{pmatrix} 1 \\ 1 \end{pmatrix}$ and $\Delta b = \frac{1}{4}\begin{pmatrix} 1 \\ 1 \end{pmatrix}$

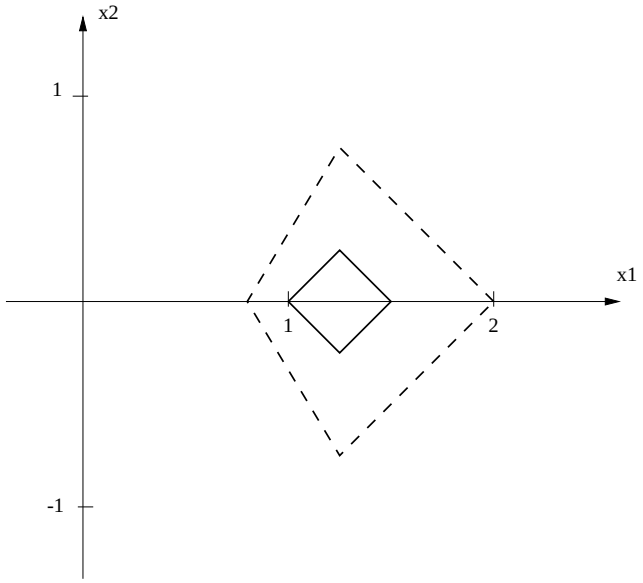

Figure 4: Ω (dotted line) and Ω' (plain line)

In this case, we can notice the correspondence between the set of constraints of Ω and Ω' that define the extremal points.
In fact we do not solve entirely the problems $\max_{y\in\Omega'} y_i$ and $\min_{y\in\Omega'} y_i$ but only determine the set of linear equations that are saturated when solving each problem $\max_{y\in\Omega'} y_i$ and $\min_{y\in\Omega'} y_i$. These results will be used during next steps.
Suppose that we know $B = (B_{i,j})$, an approximation of the inverse of A.
Moreover, let $A = \begin{pmatrix} L_1 \\ L_2 \\ \vdots \\ L_n \end{pmatrix}$ and $\Delta A\ D_\alpha = \begin{pmatrix} \Delta L_1 \\ \Delta L_2 \\ \vdots \\ \Delta L_n \end{pmatrix}$, the row partition of the matrices A and $\Delta A\ D_\alpha$.
If B were the exact inverse of A, we would obtain, when applying it to the equality $Ay = b'$ where $b' \in [\underline{b'}, \bar{b'}]$:
$y = Bb'$ where $b' \in [\underline{b'}, \bar{b'}]$
Thus, in order to minimize y_i, if $B_{i,j} \leq 0$ we have to consider $b'_j = \bar{b'_j}$ and the corresponding constraint $-L_j y \leq -\bar{b}_j$. If $B_{i,j} > 0$, we have to consider $b'_j = \underline{b'_j}$ and the constraint $L_j y \leq \bar{b}_j$.
Obviously, opposite choices apply when considering the system associated to the minimization of y_i.
Since B is only an approximation of the inverse of A, the results obtained are not sure, but, as we shall see in the next sections, we only use the results obtained for Ω' as indications in

order to solve the problem with $\Omega([A],[b],[\underline{x},\bar{x}])$.
The total cost of this step consists in the inversion of A, i.e. n^3 flops when using a LU factorization [3]. The determination of the systems associated with $\max_{y\in\Omega'} y_i$ and $\min_{y\in\Omega'} y_i$ only requires $O(n^2)$ comparisons.

3.2 Second step

3.2.1 Corresponding constraints over $\Omega([A],[b],[\underline{x},\bar{x}])$

This step consists in solving approximately the exact problem $\min_{y\in\Omega} y_i$.
For the sake of clarity, we consider from now on the minimization of y_1
If $b' = \text{mid}([\underline{b'},\bar{b'}])$, the set of saturated constraints that defines the minimization of y_1 over Ω' can be expressed as

$$\begin{cases} (d_1 L_1)y = d_1 b'_1 + \Delta b'_1 \\ (d_2 L_2)y = d_2 b'_2 + \Delta b'_2 \\ \vdots \\ (d_n L_n)y = d_n b'_n + \Delta b'_n \end{cases}$$

or

$$DAy = Db' + \Delta b'$$

where D is a diagonal matrix such that $|D| = Id(n)$ and $d_i = \pm 1$ depending on the saturated bound.
The corresponding set of constraints over Ω is:

$$\begin{cases} (d_1 L_1 - \Delta L_1)y = d_1 b'_1 + \Delta b'_1 \\ (d_2 L_2 - \Delta L_2)y = d_2 b'_2 + \Delta b'_2 \\ \vdots \\ (d_n L_n - \Delta L_n)y = d_n b'_n + \Delta b'_n \end{cases} \Leftrightarrow (DA - D_\alpha \Delta A\ )y = Db' + \Delta b'$$

3.2.2 Checking the optimality of the constraint set

Our aim is to check the optimality of the set of above constraints with respect to the minimization of y_1 over Ω. We use the following fundamental theorem of linear programming [2]:

Theorem 3 *Let u^t be the solution of $u^t(DA - \Delta A\ D_\alpha) = {e_1}^t$, where ${e_1}^t$ is the first canonical basis vector.*
If $u \geq 0$, then:

- *the set of constraints defined above is optimal for the problem $\min_{y\in\Omega} y_1$.*
- *the point y corresponding to the minimization satisfies $(DA - \Delta A\ D_\alpha)y = Db' + \Delta A\ b'$*
- $\min_{y\in\Omega} y_1 = {e_1}^t y = u^t(DA - \Delta A\ D_\alpha)y = u^t(Db' + \Delta A b')$

3.2.3 Algorithm

We now present an algorithm to solve approximately the equation $u^t(DA - \Delta A\ D_\alpha) = {e_1}^t$.
Let M be defined as follows:
$M = \Delta A\ D_\alpha B$, where B is the computed inverse of A.
Note that M does not depend on the extremal point problem we consider.
We use the following iterative scheme:

$$\begin{cases} {\delta_1}^t = {e_1}^t B \\ {\delta_{i+1}}^t = ({\delta_i}^t D)M \end{cases}$$

We stop the iteration scheme when $\delta_k \leq \epsilon_1$, a small threshold, and we set $\tilde{u}^t = (\sum_1^k {\delta_i}^t)D$.
We consider that the set of constraints is optimal if $\tilde{u} \leq \epsilon_2$, where ϵ_2 is a small positive vector.
Therefore, if $\tilde{u} \leq \epsilon_2$, we go to the third step, otherwise, we perform a step of simplex algorithm and then go to step 2.
Note that if $\tilde{u'}$ is the vector associated with the maximization of y_1, then ${\tilde{u'}}^t = (\sum_1^k (-1)^{i+1} {\delta_i}^t)D$.
Therefore, the computation of $\tilde{u'}$ only involves k additions of vectors.

3.3 Third step

3.3.1 Computing sure bounds

We are looking for a proved outer inclusion of the interval hull of the solution set. We therefore need to perform a correction step, since all previous computations were not sure.
At this stage, we know that $\tilde{u} \leq \epsilon_2$. Let $\hat{u} = \min(\tilde{u}, 0)$, $S = \begin{pmatrix} A - \Delta A\ D_\alpha \\ -A - \Delta A\ D_\alpha \end{pmatrix}$ and $s = \begin{pmatrix} \bar{b'} \\ \underline{b'} \end{pmatrix}$
If $\epsilon^t = \hat{u}^t S - {e_1}^t$, then we have

$$\begin{aligned} \hat{u}^t s &\leq \max\{z^t s,\ \ z \leq 0,\ \ z^t S \leq {e_1}^t + \epsilon^t\} \\ &\leq \min\{(e_1 + \epsilon)^t y,\ \ y \geq 0,\ \ Sy \leq s\} \text{ duality theorem of L.P.} \\ &\leq \min\{e_1^t y,\ \ y \geq 0,\ \ Sy \leq s\} + \max\{\epsilon^t y,\ \ y \geq 0,\ \ Sy \leq s\} \end{aligned}$$

and, therefore

$$\min_{y \in \Omega} y_1 \geq \hat{u}^t s - \max\{\epsilon^t y,\ \ 0 \leq y \leq \bar{x} - \underline{x}\}$$

The expression above gives a proved lower bound for $\min_{y \in \Omega} y_1$ and therefore for $\min_{y \in \Sigma} y_1$.

3.3.2 Practical implementation

In the sequel, we show how to bypass the computation of ϵ.
Let A' denote the exact inverse of the computed inverse B of A (whenever A' does not exist,

we cannot apply what follows).
Let us set
$\Delta = A' - A$,
$C = DA - \Delta A\ D_\alpha$,
$\mu = (\tilde{u} - \hat{u})^t$
$\tilde{u}^t D = {e_1}^t B\ \sum_0^{k-1} (DM)^i$,
$u^t D = {e_1}^t B\ \sum_0^{\infty} (DM)^i \Longleftrightarrow u^t\ (C - D\Delta) = {e_1}^t$,
$v^t D = {e_1}^t A\ \sum_0^{\infty} (DM)^i \Longleftrightarrow u^t\ C = {e_1}^t$.

Theorem 4 *If* $||M|| \leq \frac{1}{2}$,*and* $\epsilon' ||A||\ ||B|| \leq \frac{1}{2}$, *where* ϵ', *defined below, depends only on machine accuracy, then:*

$$||\epsilon|| \ \leq \ (\ ||A|| + ||\Delta A||\ )(\ ||\epsilon_1|| + ||\mu||\ ) + 4\ \epsilon'\ ||A||^2\ ||B||\ (\ ||\tilde{u}|| + ||\epsilon_1||\ ).$$

Proof:

- Evaluation of $||(\tilde{u} - u)^t\ C||$.
  $(\tilde{u} - u)^t\ D = {e_1}^t B\ \sum_k^{\infty} (DM)^i = \delta_k DM\ \sum_0^{\infty} (DM)^i$
  Since $||M|| \leq \frac{1}{2}$,

  $$||(\tilde{u} - u)^t\ C|| \leq ||\epsilon_1||(\ ||A|| + ||\Delta A||\ ) \text{ and } ||(\tilde{u} - u)^t|| \leq ||\epsilon_1||$$

- Evaluation of $||(v - u)^t\ C||$.
  $(v - u)^t\ C = u^t D\ \Delta$ and therefore

  $$||(v - u)^t C|| \leq ||\Delta||(\ ||\tilde{u}|| + ||\epsilon_1||\ )$$

- Evaluation of $||(v - \tilde{u})^t\ C||$.
  $||(v - \tilde{u})^t\ C|| \leq ||(v - u)^t\ C|| + ||(\tilde{u} - u)^t\ C||$ and therefore

  $$||(v - \tilde{u})^t\ C|| \leq ||\epsilon_1||\ (\ ||A|| + ||\Delta A||\ ) + ||\Delta||\ (\ ||\tilde{u}|| + ||\epsilon_1||\ )$$

- Evaluation of $||\Delta||$.
  If the computed inverse B of A is obtained with LU factorization, we know[5] that
  $AB = I + E$ where $||E|| \leq \epsilon' ||A||\ ||B||$,
  where ϵ' depends on machine accuracy.
  Thus, $A' = (I + E)^{-1} A$ and, since $||E|| \leq \epsilon' ||A||\ ||B|| \leq \frac{1}{2}$,
  $A' - A = AE\ \sum_0^{\infty} (-1)^{k+1} E^k$ and finally

  $$||\Delta|| \leq 2\epsilon'\ ||A||^2\ ||B||.$$

Since $\epsilon = \hat{u}^t\ C - {e_1}^t = (\tilde{u} - v)^t\ C + (v^t\ C - {e_1}^t) - \mu\ C$, we obtain the proof of the theorem. The crucial point is that the majoration of ϵ does not require additionnal computations. □

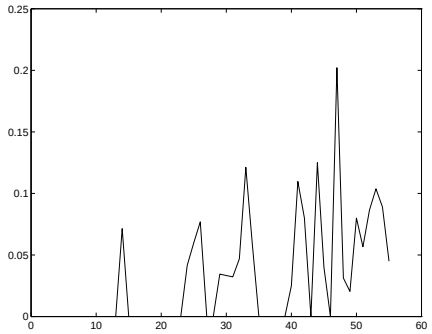

Figure 5: τ_1

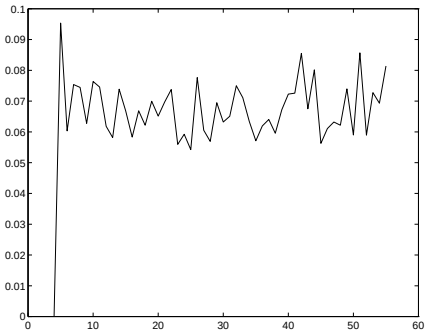

Figure 6: τ_2

Figure 7: Σ not centered – $norm(\Delta A\ ) = \frac{norm(A)*0.1}{cond(A)}$

4 Numerical results

In this section, we describe numerical results obtained with random matrices A and ΔA and for random vectors b and Δb.
We compare the results of the proposed algorithm after one iteration with the results of the algorithm proposed by Rump[8]. The initial enclosure we consider is the result of Rump's algorithm. It is known[9] that, for Rump's algorithm, the ratio $\frac{\text{volume of Rump's outer inclusion}}{\text{volume of interval hull}}$ depends on $\frac{norm(A)}{norm(\Delta A\ )cond(A)}$.
We therefore display results according to $\frac{norm(A)}{norm(\Delta A\ )cond(A)}$. As the quality of the enclosure of Σ in Ω depends on the position of the solution set with respect to 0, we also display the results for Σ centered or not centered (that is to say containing 0 or not intersecting any axis). The x-axis represents the size of the matrices.
In what follows, we display τ_1, the average number of steps of simplex algorithm necessary to the computation of an extremal point (step 2) and

$$\tau_2 = 1 - \frac{\text{width(Simplex method)}}{\text{width(Rump's algorithm)}}.$$

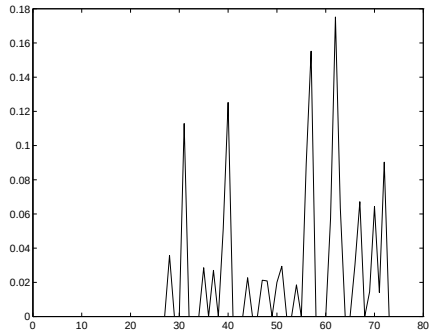

Figure 8: τ_1

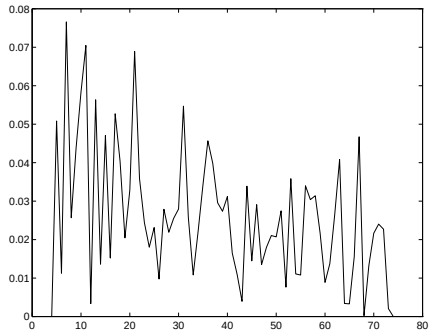

Figure 9: τ_2

Figure 10: Σ centered – $norm(\Delta A\ ) = \frac{norm(A)*0.1}{cond(A)}$

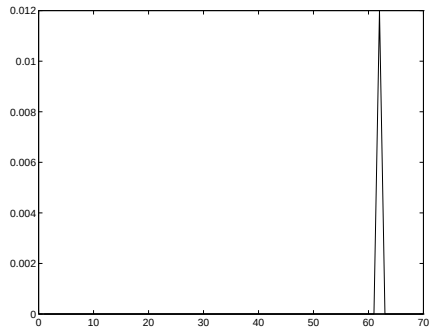

Figure 11: τ_1

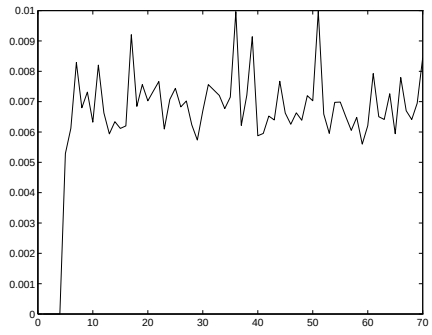

Figure 12: τ_2

Figure 13: Σ not centered – $norm(\Delta A\ ) = \frac{norm(A)*0.01}{cond(A)}$

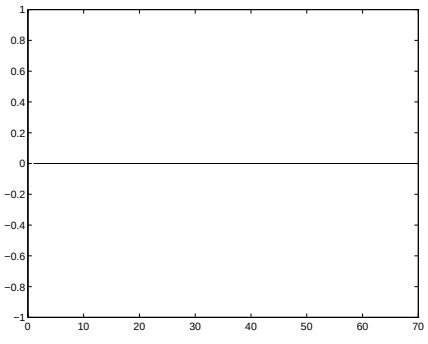

Figure 14: τ_1

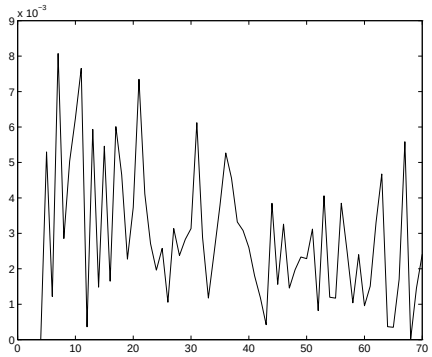

Figure 15: τ_2

Figure 16: Σ centered – $norm(\Delta A\ ) = \frac{norm(A)*0.01}{cond(A)}$

5 Algorithm for the inner inclusion.

We now present an algorithm which computes an inner inclusion of the solution set. By inner inclusion, we mean an interval vector $[\underline{y}, \bar{y}]$ such that

$$[\underline{y}, \bar{y}] \subset \square\Sigma([A], [b]) \subset \Omega([A], [b], [\underline{x}, \bar{x}]).$$

Note that it is a completely different problem to solve $[\underline{y}, \bar{y}] \subset \Sigma([A], [b])$.
The main interest of the inner inclusion we compute is to allow to estimate the accuracy of the outer inclusion and to enclose the interval hull of the solution set between two interval vectors. Moreover, numerical results indicate that the inner inclusion is very close to the interval hull (in fact, it is very often the exact interval hull).
The algorithm is based on the results for the outer inclusion. It computes $2n$ points of the solution set which are supposed to be close to extremal points.

5.1 How to find inner "extremal" points?

Let us consider again the problem of the minimization of y_1. We know that the extremal point of Ω which realizes this minimization is defined by:

$$\left.\begin{array}{l}(d_1L_1 - \Delta L_1)y = d_1b'_1 + \Delta b'_1\\(d_2L_2 - \Delta L_2)y = d_2b'_2 + \Delta b'_2\\\vdots\\(d_nL_n - \Delta L_n)y = d_nb'_n + \Delta b'_n\end{array}\right\} \Leftrightarrow (DA - D_\alpha \Delta A\ )y = Db' + \Delta b'$$

$$\Leftrightarrow D(Ax - b) = \Delta A\ (D_\alpha x + \beta) + \Delta b$$

Let us now consider a point on the frontier of Σ.

Lemma 3 *(Rohn) x belongs to the frontier of $\Sigma \Leftrightarrow |Ax - b| = \Delta A\ |x| + \Delta b$ and*

$$|Ax - b| = \Delta A\ |x| + \Delta b \Leftrightarrow \exists D'\ \textit{diagonal}\ ,\ |D'| = Id(n), D'(Ax - b) = \Delta A\ |x| + \Delta b$$

This lemma is a direct application of Oettli-Prager [7] theorem. Since $D_\alpha x + \beta$ represents an approximation of $|x|$, we can notice an analogy between the definition of x in both expressions.
We therefore consider as extremal point associated to the minimization of x_1 for the inner inclusion the point x defined by

$$D(Ax - b) = \Delta A\ |x| + \Delta b$$

where D is the matrix associated to the minimization of y_1 over Ω.

5.2 Algorithm

The algorithm consists in solving the equations

$$D(Ax - b) = \Delta A\ |x| + \Delta b \Leftrightarrow x = M|x| + a$$

where $M = A^{-1}D\Delta A$ and $a = A^{-1}(D\Delta b + b)$
Since such a point belongs to the frontier of Σ, the algorithm will lead to an inner inclusion of Σ.
If we suppose that $[A]$ is strongly regular, that is to say $\rho(|A^{-1}|\Delta A\ ) < 1$, then $\rho(M) < 1$ and we can use the algorithm proposed by Rohn:

Theorem 5 *If $\rho(M) < 1$, then, for every a, the equation*

$$x = M|x| + a$$

has a unique solution, and the iteration

$$x^{l+1} := M|x^l| + a \quad (l = 0, 1, 2, \ldots)$$

converges to the solution for every choice of the starting vector x^0

In order to obtain a good starting vector, we can solve the equation

$$D(Ax - b) = \Delta A\ (D_\alpha x + \beta) + \Delta b$$

We therefore start from the vector which corresponds to the minimization of x_1 over Ω. When $\rho(M)$ is not small enough, it is known that the convergence may be slow. Rohn therefore proposed the sign accord algorithm, which does not require strong regularity of $[A]$.
It consists in solving $x = MD'x + a$, for different matrices D'.

Sign accord algorithm (Rohn)

Step 1	Select D' with $\|D'\| = Id(n)$
Step 2	For $s = 1, \ldots, 2^n$ do:
	solve $x = MD'x + a$;
	if $D'x \geq 0$ terminate (success); otherwise compute
	$k := \min\{j = 1, \ldots, n / D'_{jj} x_j < 0\}$;
	change the sign of D'_{kk};
Step 3	Terminate (failure).

Rohn[11] has proved that the algorithm is finite when $[A]$ is regular. Although the number of steps may be exponential, it is generally reasonable. If we know the signs of the solution of $D(Ax-b) = \Delta A\ (D_\alpha x + \beta) + \Delta b$, then we can start with the matrix D' such that $D'x \geq 0$. In the cases such that the points that realize the minimization of x_1 over Σ and Ω have the same sign vector, we will not have to perform any change of sign.
In fact, none of the cases we considered for the outer inclusion requires the execution of more than one step.

5.3 Numerical results.

We display results for matrices of the same kind of those used for the outer inclusion. We only consider the case Σ centered as results for inner and outer inclusion are the same when Σ does not intersect any axis. Each figure represents the evolution with n of both quantity τ_1 and τ_2, where

$$\tau_1 = 1 - \frac{\text{width(inner inclusion)}}{\text{width(outer inclusion)}} \text{ and } \tau_2 = 1 - \frac{\text{width(outer inclusion)}}{\text{width(Rump's algorithm)}}$$

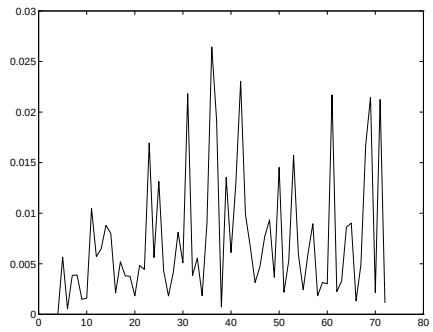

Figure 17: τ_1

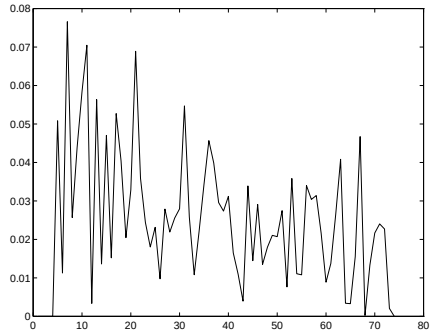

Figure 18: τ_2

Figure 19: Σ centered – $norm(\Delta A\ ) = \frac{norm(A)*0.1}{cond(A)}$

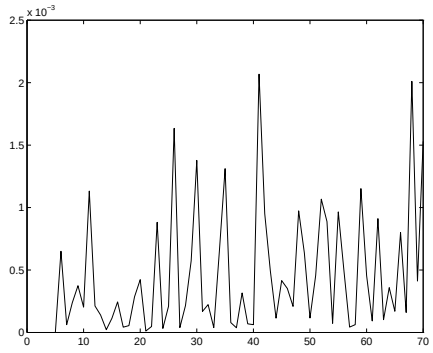

Figure 20: τ_1

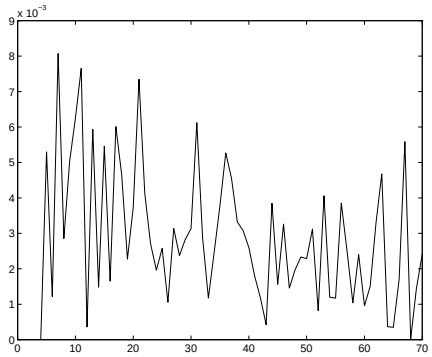

Figure 21: τ_2

Figure 22: Σ centered – $norm(\Delta A\ ) = \frac{norm(A)*0.01}{cond(A)}$

We can see that inner and outer inclusions are usually very close. Even in the case where $norm(\Delta A\ ) = \frac{norm(A)*0.1}{cond(A)}$, we usually obtain

$$1 - \frac{\text{width}(\square\Sigma)}{\text{width(outer inclusion)}} \leq 1\%.$$

Moreover, we can expect that the inner inclusion we obtain is usually the right one, since Ω and Ω' are close. More precisely, if we denote by x^+ the vector corresponding to the minimization of x_1 for the outer inclusion, by x^- the vector corresponding to the minimization of x_1 for the inner inclusion, and if we suppose that x^+ and x^- have the same sign vector, then:
$\exists D, D',\ \ |D'| = |D| = Id(n)$,
$D(Ax^+ - b) = \Delta A\ (D_\alpha x^+ + \beta) + \Delta b$
$D(Ax^- - b) = \Delta A\ D'(x^-) + \Delta b$
Therefore, if $x_{diff} = x^+ - x^-$,

$$DAx_{diff} = \Delta A\ D'x_{diff} - \Delta A\ (|x^+| - D_\alpha x^+ - \beta)$$

and

$$x_{diff} = -(DA - \Delta A\ D')^{-1}\Delta A\ (|x^+| - D_\alpha x^+ - \beta)$$

Thus,

$$\frac{\text{norm}(x_{diff})}{\text{norm}(\bar{x} - \underline{x})} \leq \frac{\text{norm}(\Delta A\ )\text{cond}(A)}{\text{norm}(A)}\ \ \frac{2\ \text{norm}(\beta)}{\text{norm}(\bar{x} - \underline{x})}$$

This allows to compute a majoration of the difference, and therefore to determine the quality of the outer inclusion, without computing the inner inclusion exactly.

6 Conclusion

In this paper, we show how to derive from the simplex algorithm an efficient method to compute inner and outer inclusion of the united solution set. The outer inclusion is obtained in $O(n^3)$ flops and does not require the preconditionning of the system. The inner inclusion coincides very often with the exact interval hull of the united solution set, but we have no way to prove this property. However, an estimation of the quality of the outer inclusion can be obtained, even without computing the inner inclusion.

References

[1] Alefeld, G. *Inclusion methods for systems of non linear equations - the interval Newton method and modifications.* Topics in Validated computations, J. Herzberger (Editor). Elsevier Science (1994).

[2] Chvatal, A. *Linear Programming.* W.H. Freeman and Company. (1983).

[3] Gill,P. ,Murray, W., Wright, M. *Numerical Linear Algebra and Optimization.* Addison-Wesley Publishing Company (1991).

[4] Kreinovich, V., Rohn, J. *Computing exact componentwise bounds on solutions of linear systems with interval data is NP-Hard* SIAM J. Matrix Anal. Appl. 16(2) (1995).

[5] Lascaux, P., Theodor, R. *Analyse Numérique Matricielle Appliquée à l'Art de l'Ingénieur.* Masson.

[6] Neumaier, A. *Interval methods for systems of equations.* Cambridge University Press (1990).

[7] Oettli, W., Prager,W. *Computability of approximate solution of linear equations with given error bounds for coefficients and right-hand sides.* Numer. Math. 6,405_9. (1964).

[8] Rump, S.M. *On the solution of interval linear systems.* Computing 47, 337-353 (1992).

[9] Rump, S.M. *Verification methods for dense and sparse systems of equations.* Topics in Validated computations, J. Herzberger (Editor). Elsevier Science (1994).

[10] Rohn, J. *NP-hardness results for linear algebraic problems with interval data.* Topics in Validated computations, J. Herzberger (Editor). Elsevier Science (1994).

[11] Rohn, J. *Checking Bounds on Solutions of Linear Interval Equation is NP-Hard* Linear Algebra and Its Applications, 223-224:589-596

[12] Schrijver A. *Theory of linear and integer programming.* John Wiley & Sons (1986).

Unit´e de recherche INRIA Lorraine, Technopôle de Nancy-Brabois, Campus scientifique,
615 rue du Jardin Botanique, BP 101, 54600 VILLERS LÈS NANCY
Unit´e de recherche INRIA Rennes, Irisa, Campus universitaire de Beaulieu, 35042 RENNES Cedex
Unit´e de recherche INRIA Rhône-Alpes, 655, avenue de l'Europe, 38330 MONTBONNOT ST MARTIN
Unit´e de recherche INRIA Rocquencourt, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex
Unit´e de recherche INRIA Sophia-Antipolis, 2004 route des Lucioles, BP 93, 06902 SOPHIA-ANTIPOLIS Cedex

Éditeur
INRIA, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex (France)
http://www.inria.fr
ISSN 0249-6399