Institute of Computer Science
Academy of Sciences of the Czech Republic

An Algorithm for Solving the System $-e \le Ax \le e,\ \|x\|_1 \ge 1$

Jiří Rohn

Technical report No. V-1149

06.01.2012

Pod Vodárenskou věží 2, 182 07 Prague 8, phone: +420 266 051 111, fax: +420 286 585 789,
e-mail:rohn@cs.cas.cz

Institute of Computer Science
Academy of Sciences of the Czech Republic

An Algorithm for Solving the System $-e \le Ax \le e,\ \|x\|_1 \ge 1$

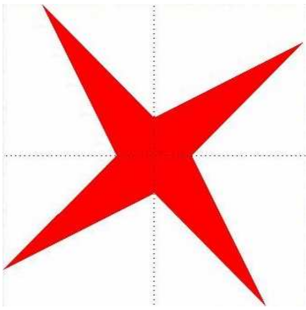
Jiří Rohn[1]

Technical report No. V-1149

06.01.2012

Abstract:

We describe a not-a-priori-exponential algorithm for solving the system $-e \le Ax \le e$, $\|x\|_1 \ge 1$. This system, despite its apparent simplicity, can be considered the basic NP-complete problem of interval computations.[2]



Keywords:
Linear inequalities, absolute value, NP-completeness, algorithm.

---

[1]This work was supported by the Institutional Research Plan AV0Z10300504.

[2]Above: logo of interval computations and related areas (depiction of the solution set of the system $[2,4]x_1 + [-2,1]x_2 = [-2,2]$, $[-1,2]x_1 + [2,4]x_2 = [-2,2]$ (Barth and Nuding [1])).

1 Introduction

In this paper we describe an algorithm for solving the system of inequalities

$$-e \leq Ax \leq e, \tag{1.1}$$

$$\|x\|_1 \geq 1, \tag{1.2}$$

where $A \in \mathbb{R}^{n\times n}$, $e = (1, 1, \dots, 1)^T \in \mathbb{R}^n$ and $\|x\|_1 = \sum_i |x_i| = e^T|x|$, which can also be written in the equivalent "shorthand" form

$$|Ax| \leq e, \qquad \|x\|_1 \geq 1. \tag{1.3}$$

The choice of the system may seem surprising: why just this system, and why such a specific form (using e and 1)? There are three reasons for this formulation.

First, in [2], Theorem 2.3 it was proved that *the problem of checking solvability of the system (1.1), (1.2) is NP-complete for nonnegative symmetric positive definite rational matrices*, and this result was further used there for proving NP-hardness or (co-)NP-completeness of nine other problems (see Theorems 2.12, 2.15, 2.18, 2.21, 2.30, 2.33, 2.38, 3.15 and 3.17 in [2]), thus having demonstrated that it is an ideal tool for establishing complexity results for problems with interval data. This is why this problem was called "the basic NP-complete problem of interval computations" in [4].

Second, it turns out that one of the basic NP-complete problems termed as such by Garey and Johnson [3] can be transformed to our problem.

And third – and this the topic of the present paper – there exists a not-a-priori-exponential algorithm for solving (1.1), (1.2) which, in turn, yields a not-a-priori-exponential algorithm for solving one of the basic NP-hard problems mentioned in the previous paragraph; formulation of the latter algorithm will possibly appear elsewhere.

The algorithm for solving (1.1), (1.2), which in a finite number of steps either finds a solution to (1.1), (1.2) or proves its nonexistence, is listed in the form of several interconnected MATLAB-like functions in the last Section 4. The preceding two sections bring the theoretical background and some examples.

2 Description

In order that the algorithm, whose description stretches over several pages, could be presented as a whole and not intertwined with the text, it is given in the last Section 4.

Theorem 1. *For each square matrix A the algorithm* **basintnpprob** *(Fig. 4.1) in a finite, but not-a-priori-exponential number of steps either finds a solution x of the system (1.1), (1.2) (the case of $x \neq [\,]$), or proves that no such a solution exists (the case of $x = [\,]$).*

Proof. As proved in [5], the algorithm **singreg** (Fig. 4.2), when applied to the interval matrix $[A - ee^T, A + ee^T]$ (Fig. 4.1, lines (06)-(07)) in a finite, but not-a-priori-exponential number of steps either yields a singular matrix S satisfying $|A-S| \leq ee^T$ (the case of $S \neq [\,]$), or states that such a singular matrix S does not exist (the case of $S = [\,]$).

In the first case, taking an arbitrary $x \neq 0$ satisfying $Sx = 0$ (which exists because S is singular), we have

$$|Ax| = |(A - S)x| \leq |A - S||x| \leq ee^T|x| = \|x\|_1 e$$

so that for $x' = x/\|x\|_1$ we have $|Ax'| \leq e$ and $\|x'\|_1 = 1$, which means that x' solves (1.3) (Fig. 4.1, lines (09)-(10)).

In the second case there does not exist a singular matrix S satisfying $|A - S| \leq ee^T$. We shall prove that in this case the system (1.3) has no solution. Suppose to the contrary that (1.3) has a solution x. Then

$$|Ax| \leq e \leq e\|x\|_1 = ee^T|x|,$$

so that the interval matrix $[A - ee^T, A + ee^T]$ is singular, i.e., there exists a singular matrix S satisfying $|A - S| \leq ee^T$, a contradiction (Fig. 4.1, line (08)). □

3 Examples

In this section we give two examples with 100×100 matrices. In the first one a solution is found, whereas the second one has no solution.

```
>> tic, rand('state',1); n=100; A=rand(n,n); x=basintnpprob(A);
>> x', min(ones(n,1)-abs(A*x)), norm(x,1), toc
ans =
  Columns 1 through 10
 -0.0293 -0.0154 -0.0091  0.0138  0.0099 -0.0185  0.0186 -0.0082 -0.0076  0.0213
  Columns 11 through 20
  0.0074 -0.0204 -0.0157  0.0054  0.0303  0.0005  0.0155 -0.0003  0.0026 -0.0037
  Columns 21 through 30
 -0.0023  0.0111  0.0045 -0.0043  0.0043 -0.0027  0.0032 -0.0157  0.0070 -0.0069
  Columns 31 through 40
 -0.0067  0.0135  0.0097  0.0004 -0.0200  0.0013  0.0137 -0.0030 -0.0003  0.0033
  Columns 41 through 50
  0.0009 -0.0148 -0.0051  0.0008  0.0059 -0.0047  0.0054  0.0229 -0.0133  0.0294
  Columns 51 through 60
  0.0103  0.0101  0.0036  0.0028  0.0146  0.0215 -0.0288 -0.0113  0.0229 -0.0021
  Columns 61 through 70
 -0.0035 -0.0065  0.0161  0.0094  0.0051 -0.0048  0.0053 -0.0094 -0.0082 -0.0002
  Columns 71 through 80
  0.0046 -0.0094 -0.0128  0.0062 -0.0271 -0.0053  0.0013 -0.0169  0.0014 -0.0203
  Columns 81 through 90
  0.0225 -0.0145 -0.0092 -0.0110 -0.0008  0.0045 -0.0143  0.0081  0.0115  0.0201
  Columns 91 through 100
 -0.0168  0.0108  0.0026  0.0143  0.0050  0.0055 -0.0094 -0.0123 -0.0028 -0.0111
ans =
  0.9981
```

```
ans =
  1.0000
Elapsed time is 0.804711 seconds.

>> tic, rand('state',1); n=100; A=10000*rand(n,n); x=basintnpprob(A);
>> x', min(ones(n,1)-abs(A*x)), norm(x,1), toc
x =
     []
ans =
     []
ans =
     []
Elapsed time is 0.407147 seconds.
```

4 The algorithm

function $x =$ **basintnpprob** (A)
% BASic INTerval NP PROBlem.
% $x \neq [\,]$: x solves $-e \leq Ax \leq e$, $\|x\|_1 \geq 1$.
% $x = [\,]$: $-e \leq Ax \leq e$, $\|x\|_1 \geq 1$ has no solution.
$n = \text{size}(A, 1)$; $e = \text{ones}(n, 1)$;
$\mathbf{A} = [A - ee^T, A + ee^T]$;
$S =$ **singreg** $(\mathbf{A})$;
if $S = [\,]$, $x = [\,]$; **return**, **end**
find an $x \neq 0$ satisfying $Sx = 0$;
$x = x/\|x\|_1$;

Figure 4.1: An algorithm for solving the basic interval NP-complete problem.

function $S = $ **singreg** $(\mathbf{A})$
% $S \neq [\,]$: S is a singular matrix in $\mathbf{A}$.
% $S = [\,]$: no singular matrix in $\mathbf{A}$ exists.
 $S = [\,]$; $n = \mathrm{size}(\mathbf{A}, 1)$; $e = (1, \dots, 1)^T \in \mathbb{R}^n$;
 if A_c is singular, $S = A_c$; **return**, **end**
 $R = A_c^{-1}$; $D = \Delta|R|$;
 if $D_{kk} = \max_j D_{jj} \geq 1$
    $x = R_{\bullet k}$;
    for $i = 1 : n$
        if $(\Delta|x|)_i > 0$, $y_i = (A_c x)_i / (\Delta|x|)_i$; **else** $y_i = 1$; **end**
        if $x_i \geq 0$, $z_i = 1$; **else** $z_i = -1$; **end**
    end
    $S = A_c - T_y \Delta T_z$; **return**
 end
 if $\varrho(D) < 1$, **return**, **end**
 $b = e$;
 $x = Rb$; $\gamma = \min_k |x_k|$;
 for $i = 1 : n$
    for $j = 1 : n$
        $x' = x - 2b_j R_{\bullet j}$;
        if $\min_k |x'_k| > \gamma$, $\gamma = \min_k |x'_k|$; $x = x'$; $b_j = -b_j$; **end**
    end
 end
 $[\mathbf{x}, S] = $ **intervalhull** $(\mathbf{A}, [b, b])$;

Figure 4.2: An algorithm for finding a singular matrix in an interval matrix.

function $[\mathbf{x}, S] =$ **intervalhull** $(\mathbf{A}, \mathbf{b})$
% Computes either the interval hull $\mathbf{x}$
% of the solution set of $\mathbf{A}x = \mathbf{b}$,
% or a singular matrix $S \in \mathbf{A}$.
$\mathbf{x} = [\,]$; $S = [\,]$;
if A_c is singular, $S = A_c$; **return**, **end**
$x_c = A_c^{-1} b_c$; $z = \mathrm{sgn}(x_c)$; $\underline{x} = x_c$; $\overline{x} = x_c$;
$Z = \{z\}$; $D = \emptyset$;
while $Z \neq \emptyset$
  select $z \in Z$; $Z = Z - \{z\}$; $D = D \cup \{z\}$;
  $[Q_z, S] =$ **qzmatrix** $(\mathbf{A}, z)$;
  if $S \neq [\,]$, $\mathbf{x} = [\,]$; **return**, **end**
  $[Q_{-z}, S] =$ **qzmatrix** $(\mathbf{A}, -z)$;
  if $S \neq [\,]$, $\mathbf{x} = [\,]$; **return**, **end**
  $\overline{x}_z = Q_z b_c + |Q_z|\delta$;
  $\underline{x}_z = Q_{-z} b_c - |Q_{-z}|\delta$;
  if $\underline{x}_z \leq \overline{x}_z$
    $\underline{x} = \min(\underline{x}, \underline{x}_z)$; $\overline{x} = \max(\overline{x}, \overline{x}_z)$;
    for $j = 1 : n$
      $z' = z$; $z'_j = -z'_j$;
      if $((\underline{x}_z)_j (\overline{x}_z)_j \leq 0$ **and** $z' \notin Z \cup D)$
        $Z = Z \cup \{z'\}$;
      end
    end
  end
end
$\mathbf{x} = [\underline{x}, \overline{x}]$;
function $[Q_z, S] =$ **qzmatrix** $(\mathbf{A}, z)$
% Computes either a solution Q_z
% of the equation $QA_c - |Q|\Delta T_z = I$,
% or a singular matrix $S \in \mathbf{A}$.
for $i = 1 : n$
  $[x, S] =$ **absvaleqn** $(A_c^T, -T_z \Delta^T, e_i)$;
  if $S \neq [\,]$, $S = S^T$; $Q_z = [\,]$; **return**
  end
  $(Q_z)_{i\bullet} = x^T$;
end
$S = [\,]$;

Figure 4.3: An algorithm for computing the interval hull.

```
function [x, S] = absvaleqn (A, B, b)
% Finds either a solution x to Ax + B|x| = b, or
% a singular matrix S satisfying |S − A| ≤ |B|.
x = []; S = []; i = 0; r = 0 ∈ ℝ^n; X = 0 ∈ ℝ^{n×n};
if A is singular, S = A; return, end
z = sgn(A^{-1} b);
if A + B T_z is singular, S = A + B T_z; return, end
x = (A + B T_z)^{-1} b;
C = −(A + B T_z)^{-1} B;
while z_j x_j < 0 for some j
    i = i + 1;
    k = min{j | z_j x_j < 0};
    if 1 + 2 z_k C_kk ≤ 0
        S = A + B(T_z + (1/C_kk) e_k e_k^T);
        x = []; return
    end
    if ((k < n and r_k > max_{k<j} r_j) or (k = n and r_n > 0))
        x = x − X_•k;
        for j = 1 : n
            if (|B||x|)_j > 0, y_j = (Ax)_j / (|B||x|)_j; else y_j = 1; end
        end
        z = sgn(x);
        S = A − T_y |B| T_z;
        x = []; return
    end
    r_k = i;
    X_•k = x;
    z_k = −z_k;
    α = 2 z_k / (1 − 2 z_k C_kk);
    x = x + α x_k C_•k;
    C = C + α C_•k C_k•;
end
```

Figure 4.4: An algorithm for solving an absolute value equation.

Bibliography

[1] W. Barth and E. Nuding, *Optimale Lösung von Intervallgleichungssystemen*, Computing, 12 (1974), pp. 117–125. 1

[2] M. Fiedler, J. Nedoma, J. Ramík, J. Rohn, and K. Zimmermann, *Linear Optimization Problems with Inexact Data*, Springer-Verlag, New York, 2006. 2

[3] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, Freeman, San Francisco, 1979. 2

[4] J. Rohn, *VERBASINTNPPROB: Verified solution of the basic NP-complete problem of interval computations*, 2008. `http://uivtx.cs.cas.cz/~rohn/matlab/verbasintnpprob.html`. 2

[5] J. Rohn, *An algorithm for finding a singular matrix in an interval matrix*, Technical Report 1087, Institute of Computer Science, Academy of Sciences of the Czech Republic, Prague, November 2010. `http://uivtx.cs.cas.cz/~rohn/publist/singreg.pdf`. 2