Institute of Computer Science
Academy of Sciences of the Czech Republic

Verified Solutions of Linear Equations

Jiří Rohn

Technical report No. V-1121

04.08.2011

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

Verified Solutions of Linear Equations

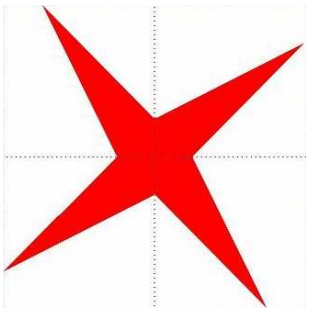
Jiří Rohn[1]

Technical report No. V-1121

04.08.2011

Abstract:

We describe a MATLAB/INTLAB file ks.m for computing a verified solution of a system of linear equations $Ax = b$ with $A \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$.



Keywords:
Linear equations, verified solution, MATLAB/INTLAB file.[2]

[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 Section 4 of this report we bring a MATLAB/INTLAB file `ks.m` for computing a verified solution of a system of linear equations $Ax = b$ with $A \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$. It is NOT a part of VERSOFT [2] and it has been created as a tool for handling degeneracy in verification software for solving linear programming problems, but it of independent interest in view of several particular features described in Section 2. Several illustrative examples are shown in Section 3.

2 Description

The file is invoked by

$$[\mathtt{x},\mathtt{E}] = \mathtt{ks}(\mathtt{A},\mathtt{b})$$

and the input and output data are described in `help ks`:

```
% KS    Verified solutions of linear equations.
%
% A : m-by-n, b : m-by-1,
%~isinf(x.inf(1))&&~isnan(x.inf(1)) : x verified to contain a solution of AX=b,
% isinf(x.inf(1)) : AX=b verified to have no solution,
% isnan(x.inf(1)) : computation failed,
% E : verbal description (in particular, may announce
%     uniqueness or nonexistence of solution),
% m<n : x contains at least n-rank(A) exact zeros.
```

Several examples are given in Section 3 below. The file calls at several places VERSOFT function `zd(A)` whose value is 1 if the columns of `A` are verified linearly independent, 0 if they are verified linearly dependent, and -1 if no verified result had been found. With this having been said, the construction of the file is easily understandable. First it finds a verified basis `B` of the range space of `A`, `B` itself being a submatrix of `A`. If `[B b]` has linearly dependent columns, then `B*x=b` has a solution `x=inv(B'*B)*B'*b` which, after being complemented by zeros, becomes a solution of `A*x=b`. If `B=A`, then the solution is verified unique. If the columns of `[B b]` are linearly independent, then it is verified that no solution exists (in this case `x` is outputted as a vector of interval `Inf`'s). In the remaining cases a verified result has not been found and `x` is outputted as a vector of interval `NaN`'s.

3 Examples

If `m<n`, then the file usually gives a solution even for matrices with heavily dependent columns:

```
>> m=3; n=5; A=ones(m,n), b=A*ones(n,1), [x,E]=ks(A,b)
A =
     1     1     1     1     1
     1     1     1     1     1
     1     1     1     1     1
```

```
b =
     5
     5
     5
intval x =
[   4.99999999999999,   5.00000000000001]
[   0.00000000000000,   0.00000000000000]
[   0.00000000000000,   0.00000000000000]
[   0.00000000000000,   0.00000000000000]
[   0.00000000000000,   0.00000000000000]
E =
solution found
```

Notice that the solution found is a "basic solution", i.e., it has (at least) `n-rank(A)` exact zero entries. But if `m`$\geq$`n`, then the situation may be different, and sometimes unpredictable. Consider the example

```
>> n=4; A=1:n^2; A=reshape(A,n,n); A=A', b=A*ones(n,1), [x,E]=ks(A,b)
A =
     1     2     3     4
     5     6     7     8
     9    10    11    12
    13    14    15    16
b =
    10
    26
    42
    58
intval x =
[  -2.00000000000001,  -1.99999999999999]
[   5.99999999999999,   6.00000000000001]
[   0.00000000000000,   0.00000000000000]
[   0.00000000000000,   0.00000000000000]
E =
solution found
```

A solution has been found. But the algorithm fails for example of the same type with `n=5`:

```
>> n=5; A=1:n^2; A=reshape(A,n,n); A=A', b=A*ones(n,1), [x,E]=ks(A,b)
A =
     1     2     3     4     5
     6     7     8     9    10
    11    12    13    14    15
    16    17    18    19    20
    21    22    23    24    25
b =
    15
    40
```

```
    65
    90
   115
intval x =
[                NaN,                NaN]
[                NaN,                NaN]
[                NaN,                NaN]
[                NaN,                NaN]
[                NaN,                NaN]
E =
computation failed
```

This behavior of the file is hard to explain. The only reasonable explanation the author can offer is that the heavy interval machinery behind the subfunction `zd` breaks down because of its inability to determine the signs of certain quantities (as of the type `[-1e-13,1e-13]`, etc.). Now consider randomly generated examples with `m>n`:

```
>> rand('state',1); m=7; n=5; format short, A=2*rand(m,n)-1, b=A*rand(n,1),
>> format long, [x,E]=ks(A,b)
A =
    0.9056    0.0374    0.0766   -0.8810    0.1475
    0.4081   -0.9556    0.8205    0.2542    0.1358
    0.9078   -0.2482    0.0506   -0.4696   -0.2110
    0.1963    0.7972   -0.3863   -0.3753    0.3832
    0.6815   -0.1420   -0.9311    0.0454    0.1386
   -0.1144   -0.6009    0.4307   -0.1827   -0.3136
    0.6736   -0.3938    0.5374    0.7859    0.1899
b =
    0.3312
    0.8784
    0.0773
    0.0089
   -0.4948
    0.0536
    0.8349
intval x =
[   0.27394623997726,   0.27394623997727]
[   0.04810002952102,   0.04810002952103]
[   0.83808906449354,   0.83808906449355]
[   0.10254138654555,   0.10254138654556]
[   0.72827462828635,   0.72827462828636]
E =
unique solution found
```

With randomly generated right-hand side, it is almost sure that the system will possess no solution. Indeed,

```
>> rand('state',2); m=7; n=5; format short, A=2*rand(m,n)-1, b=rand(m,1),
```

```
>> format long, [x,E]=ks(A,b)
A =
    0.7503   -0.6565    0.2952    0.1218   -0.6804
   -0.3642    0.7399    0.9853    0.4669    0.4577
   -0.4535   -0.5126   -0.2309    0.2312    0.6202
    0.3530    0.6858   -0.7043   -0.4250   -0.6423
   -0.8577    0.1153    0.9433    0.9968   -0.3403
   -0.6068   -0.2864   -0.5629   -0.9976   -0.7579
    0.0582   -0.5352    0.2150    0.1983   -0.0828
b =
    0.6006
    0.1836
    0.6767
    0.0593
    0.7868
    0.6637
    0.2011
intval x =
[                Inf,                Inf]
[                Inf,                Inf]
[                Inf,                Inf]
[                Inf,                Inf]
[                Inf,                Inf]
E =
no solution exists
```

Notice that S. M. Rump's built-in INTLAB [3] file `verifylss` applied to the same example yields a solution:

```
>> rand('state',2); m=7; n=5; format short, A=2*rand(m,n)-1, b=rand(m,1),
>> format long, x=verifylss(A,b)
A =
    0.7503   -0.6565    0.2952    0.1218   -0.6804
   -0.3642    0.7399    0.9853    0.4669    0.4577
   -0.4535   -0.5126   -0.2309    0.2312    0.6202
    0.3530    0.6858   -0.7043   -0.4250   -0.6423
   -0.8577    0.1153    0.9433    0.9968   -0.3403
   -0.6068   -0.2864   -0.5629   -0.9976   -0.7579
    0.0582   -0.5352    0.2150    0.1983   -0.0828
b =
    0.6006
    0.1836
    0.6767
    0.0593
    0.7868
    0.6637
    0.2011
```

```
intval x =
[  -0.52978662015232,  -0.52978662015231]
[  -0.38858750680391,  -0.38858750680390]
[   0.00333105362976,   0.00333105362977]
[   0.27861841306592,   0.27861841306593]
[  -0.52244233187229,  -0.52244233187228]
```

but this is not a solution (whose existence is precluded by the preceding verified result), see

```
>> A*x-b
intval ans =
[  -0.35254139515371,  -0.35254139515368]
[  -0.38395986528861,  -0.38395986528860]
[  -0.49761282276661,  -0.49761282276660]
[  -0.29802762010898,  -0.29802762010897]
[   0.08141492579097,   0.08141492579098]
[  -0.11477767500914,  -0.11477767500913]
[   0.07527631411407,   0.07527631411408]
```

but a least squares solution (this fact being not made clear in the output of `verifylss`).

4 The file

```
function [x,E]=ks(A,b)
% A : m-by-n, b : m-by-1,
%~isinf(x.inf(1))&&~isnan(x.inf(1)) : x verified to contain a solution of AX=b,
% isinf(x.inf(1)) : AX=b verified to have no solution,
% isnan(x.inf(1)) : computation failed,
% E : verbal description (in particular, may announce
%     uniqueness or nonexistence of solution),
% m<n : x contains at least n-rank(A) exact zeros.
gr=getround; setround(0);
[m,n]=size(A); b=b(:); x=repmat(infsup(NaN,NaN),n,1);
if nargin~=2||nargout>2||m~=length(b)||~isreal(A)||~isreal(b)||isintval(A)||isintval(b)
    E='wrong data'; setround(gr); return
end
if all(all(A==0))
    if all(b==0)
        x=repmat(infsup(0,0),n,1); E='solution found'; setround(gr); return
    else
        x=repmat(infsup(NaN,NaN),n,1); E='no solution exists'; setround(gr); return
    end
end
[AR,K]=rref(A);
if isempty(K), E='computation failed'; setround(gr); return, end
B=A(:,K);
fc=zd(B);
if fc~=1, E='computation failed'; setround(gr); return, end
[x,E]=ks555(A,K,B,b);
if ~isnan(x.inf(1))&&~isinf(x.inf(1))&&...
    length(find(~in(repmat(infsup(0,0),n,1),x),1))<length(K)
    K=find(~in(repmat(infsup(0,0),n,1),x),1);
    if ~isempty(K)
        B=A(:,K);
        x1=ks555(A,K,B,b);
        if ~isnan(x1.inf(1))&&~isinf(x1.inf(1))
            x=x1;
        end
    else
        if all(b==0)
            x=repmat(infsup(0,0),n,1);
        end
    end
end
setround(gr);
```

```
% Subfunction
function [x,E]=ks555(A,K,B,b)
[m,n]=size(A);
k=length(K);
fc=zd([B b]);
switch fc
    case 1
        x=repmat(infsup(Inf,Inf),n,1); E='no solution exists';
    case 0
        xK=verifylss([B -eye(m,m); zeros(k,k) B'],[b' zeros(1,k)]');
        if isnan(xK.inf(1,1))
            x=repmat(infsup(NaN,NaN),n,1); E='computation failed';
        else
            x=infsup(zeros(n,1),zeros(n,1)); x(K)=xK(1:k);
            if k<n, E='solution found'; else E='unique solution found'; end
        end
    case -1
        x=repmat(infsup(NaN,NaN),n,1); E='computation failed';
end
```

5 Download

The source file can be downloaded from
http://uivtx.cs.cas.cz/~rohn/matlab/others/ks.m .

Dedication

Dedicated to E. K. and K. S.

Bibliography

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

[2] J. Rohn, *VERSOFT: Verification software in MATLAB/INTLAB*, 2009. http://uivtx.cs.cas.cz/~rohn/matlab. 2

[3] S. Rump, *INTLAB - INTerval LABoratory*, in Developments in Reliable Computing, T. Csendes, ed., Kluwer Academic Publishers, Dordrecht, 1999, pp. 77–104. http://www.ti3.tu-harburg.de/rump/. 5