Electronic Journal of Linear Algebra ISSN 1081-3810
A publication of the International Linear Algebra Society
Volume 26, pp. 836-841, December 2013

ELA

A TWO-MATRIX ALTERNATIVE*

JIRI ROHN†

Abstract. An algorithm for computing a solution of a two-matrix alternative is described. Given two square matrices $A, B \in \mathbb{R}^{n\times n}$, it computes a nontrivial solution either to $|Ax| \le |B||x|$, or to $|Ay| > |B||y|$.

Key words. Two-matrix alternative, Solution, Algorithm.

AMS subject classifications. 15A06, 65G40.

1. Introduction. In [1], Corollary 4.1, the author proved the following result which we call a "two-matrix alternative".

THEOREM 1.1. *For each $A, B \in \mathbb{R}^{n\times n}$, at least one of the inequalities*

$$|Ax| \le |B||x|, \tag{1.1}$$

$$|Ay| > |B||y| \tag{1.2}$$

has a nontrivial solution.

Here, both the absolute value as well as the two types of inequalities are understood entrywise. Of course, a solution of (1.2) is always nontrivial, so that the non-triviality requirement concerns the inequality (1.1) only. The result is a little bit surprising, considering full generality of the data.

The theoretical proof given in [1] gives little clue as to how to compute a solution of either (1.1), or (1.2). In the present paper, we show that employing the recently published algorithm **absvaleqn** (Fig. 2.1) leads to a simple algorithmic solution of the problem (Fig. 3.1). In this way, we also find a constructive proof of Theorem 1.1.

2. Absolute value equation. As it will be seen in the next section, our task is greatly simplified by existence of the algorithm **absvaleqn** published in [2, 3] and

*Received by the editors on June 14, 2013. Accepted for publication on December 15, 2013. Handling Editor: Bryan Shader.

†Institute of Computer Science, Czech Academy of Sciences, Prague, Czech Republic (rohn@cs.cas.cz). This work was supported with institutional support RVO:67985807.

Electronic Journal of Linear Algebra ISSN 1081-3810
A publication of the International Linear Algebra Society
Volume 26, pp. 836-841, December 2013

ELA

described here in a MATLAB-like form in Fig. 2.1. The following theorem comes from [3].

THEOREM 2.1. *For each $A, B \in \mathbb{R}^{n\times n}$ and each $b \in \mathbb{R}^n$, the algorithm* **absvaleqn** *(Fig. 2.1) in a finite number of steps either finds a solution x of the equation*

$$Ax + B|x| = b, \tag{2.1}$$

or finds a singular matrix S satisfying

$$|S - A| \le |B|. \tag{2.2}$$

The proof of this theorem, given in [2, 3], is not quite easy. Therefore, for the sake of understandability, we add some explanations here.

First, it is proved by the Sherman-Morrison inverse matrix formula that after each update of x and C in lines (30), (31) of Fig. 2.1, there holds

$$x = (A + BT_z)^{-1}b \tag{2.3}$$

and $C = -(A + BT_z)^{-1}B$ for the current z, where $T_z = \mathrm{diag}(z)$.

Second, there are altogether four singular matrix outputs in Fig. 2.1, namely in lines (05), (07), (14), and (23). In the first two cases, singularity of S is obvious, in line (14) there holds $\det(S) = 0$ in view of the Sherman-Morrison determinant formula, and in line (23) we have $Sx = 0$, where $x \neq 0$ due to (10); in all cases $|S - A| \le |B|$.

Third, if at some step the condition in (10) does not hold, then $z_j x_j \ge 0$ for each j, where z is a ± 1-vector due to (06), (28), hence $T_z x \ge 0$. Thus, $|x| = |T_z x| = T_z x$, and from (2.3), we obtain $Ax + B|x| = Ax + BT_z x = (A + BT_z)x = b$, so that x solves (2.1).

And fourth (and this is the most sophisticated part), the algorithm is finite because the sequence of k's generated in line (12) of Fig. 2.1 is finite owing to the following property: *Between each two occurrences of the same k in the sequence there is an occurrence of some $\ell > k$ in the sequence* (this is a consequence of the condition in line (17)). Thus, n can occur at most once in the sequence, $n - 1$ at most twice, $n-2$ at most $4 = 2^2$ times, $\ldots$, k at most 2^{n-k} times, $\ldots$, and finally 1 at most 2^{n-1} times, so that the sequence consists of at most $1+2+2^2+\cdots+2^{n-1} = 2^n - 1$ entries and thus it is finite. For example, the longest sequence having the above-mentioned property (in italics) for $n = 5$ is

1 2 1 3 1 2 1 4 1 2 1 3 1 2 1 5 1 2 1 3 1 2 1 4 1 2 1 3 1 2 1

with $31 = 2^5 - 1$ entries (observe the pattern).

Electronic Journal of Linear Algebra ISSN 1081-3810
A publication of the International Linear Algebra Society
Volume 26, pp. 836-841, December 2013

ELA

In reality, the situation is not as grim as the above worst case might suggest. As shown in [2], the *average* length of the sequence of k's for randomly generated matrices of various sizes is about $0.1n$, so that the algorithm is surprisingly efficient. Indeed, among 1,000,000 randomly generated examples of 10×10 matrices the maximum number of iterations found was 22 in contrast to the worst-case estimate $2^{10} - 1 = 1023$.

function $[x, S] =$ **absvaleqn** (A, B, b)
% Finds either a solution x to $Ax + B|x| = b$, or
% a singular matrix S satisfying $|S - A| \le |B|$.
$x = [\,]$; $S = [\,]$; $i = 0$; $r = 0 \in \mathbb{R}^n$; $X = 0 \in \mathbb{R}^{n \times n}$;
if A is singular, $S = A$; **return**, **end**
$z = \mathrm{sgn}(A^{-1}b)$;
if $A + BT_z$ is singular, $S = A + BT_z$; **return**, **end**
$x = (A + BT_z)^{-1}b$;
$C = -(A + BT_z)^{-1}B$;
while $z_j x_j < 0$ for some j
    $i = i + 1$;
    $k = \min\{j \mid z_j x_j < 0\}$;
    if $1 + 2z_k C_{kk} \le 0$
        $S = A + B(T_z + (1/C_{kk})e_k e_k^T)$;
        $x = [\,]$; **return**
    end
    if $((k < n$ **and** $r_k > \max_{k<j} r_j)$ **or** $(k = n$ **and** $r_n > 0))$
        $x = x - X_{\bullet k}$;
        for $j = 1 : n$
            if $(|B||x|)_j > 0$, $y_j = (Ax)_j/(|B||x|)_j$; **else** $y_j = 1$; **end**
        end
        $z = \mathrm{sgn}(x)$;
        $S = A - T_y|B|T_z$;
        $x = [\,]$; **return**
    end
    $r_k = i$;
    $X_{\bullet k} = x$;
    $z_k = -z_k$;
    $\alpha = 2z_k/(1 - 2z_k C_{kk})$;
    $x = x + \alpha x_k C_{\bullet k}$;
    $C = C + \alpha C_{\bullet k} C_{k \bullet}$;
end

FIG. 2.1. *An algorithm for solving an absolute value equation [3].*

Electronic Journal of Linear Algebra ISSN 1081-3810
A publication of the International Linear Algebra Society
Volume 26, pp. 836-841, December 2013

ELA

3. The algorithm. With the **absvaleqn** algorithm at our disposal, it is now relatively easy to resolve our basic problem.

THEOREM 3.1. *For each A, $B \in \mathbb{R}^{n\times n}$ the algorithm* **twomatralt** *(Fig. 3.1) in a finite number of steps finds a nontrivial solution either of (1.1), or of (1.2).*

Proof. As it can be seen from Fig. 3.1, line (05), the algorithm **twomatralt** first runs

$$[y, S] = \textbf{absvaleqn}\,(A, -|B|, e),$$

where $e = (1, 1, \ldots, 1)^T$. According to Theorem 2.1, there are two possible outcomes.

If $y \neq [\,]$, then y solves $Ay - |B||y| = e$, hence $Ay = |B||y| + e > |B||y| \geq 0$, so that $Ay > 0$ which means that $Ay = |Ay|$ and $|Ay| > |B||y|$, showing that y is a solution of (1.2).

If $S \neq [\,]$, then S is a singular matrix satisfying $|S - A| \leq |B|$. Take an arbitrary $x \neq 0$ satisfying $Sx = 0$. Then $|Ax| = |(A - S)x| \leq |A - S||x| \leq |B||x|$, so that x is a nontrivial solution of (1.1). ∎

Notice that in this way we have found another, this time constructive, proof of Theorem 1.1.

For the purposes of the next two sections, let us introduce the numbers

$$\alpha(x) = \min_i (|B||x| - |Ax|)_i,$$
$$\beta(y) = \min_i (|Ay| - |B||y|)_i.$$

Then (1.1) is equivalent to $\alpha(x) \geq 0$, and (1.2) is equivalent to $\beta(y) > 0$.

(01) **function** $[x, y] =$ **twomatralt** (A, B)
(02) % $x \neq [\,]$: x solves $|Ax| \leq |B||x|$, $x \neq 0$.
(03) % $y \neq [\,]$: y solves $|Ay| > |B||y|$.
(04) $x = [\,]$; $y = [\,]$; $e = (1, 1, \ldots, 1)^T \in \mathbb{R}^n$;
(05) $[y, S] =$ **absvaleqn** $(A, -|B|, e)$;
(06) **if** $y \neq [\,]$, **return**, **end**
(07) find an $x \neq 0$ satisfying $Sx = 0$;

FIG. 3.1. *An algorithm for computing a two-matrix alternative.*

4. Solvability of both inequalities. There are instances of A, B for which both inequalities (1.1), (1.2) are solvable, as it can be shown on the following example. Consider the data

Electronic Journal of Linear Algebra ISSN 1081-3810
A publication of the International Linear Algebra Society
Volume 26, pp. 836-841, December 2013


```
A =
    93   -94    43
   -53   -24    96
    55   -62    70
B =
    19    27     4
    -7   -12   -10
   -12     6    12
```

Then for

```
x =
    0.4331
    0.8159
    0.3831
```

we have

```
>> alpha=min(abs(B)*abs(x)-abs(A*x))
alpha =
   10.8962
```

so that x solves (1.1), and for

```
y =
   -0.0846
   -0.1907
   -0.0489
```

we have

```
>> beta=min(abs(A*y)-abs(B)*abs(y))
beta =
    1.0000
```

so that y solves (1.2). To find this example, we generated randomly integer 3×3 matrices and we first applied the **twomatralt** algorithm. If x was found, we looked for a solution of the equation

$$Ay - |B||y| = e$$

using algorithm **absvaleqnall** from [4] which finds a solution if it exists, albeit at the expense of employing an exhaustive search algorithm. Only the 582nd randomly generated example satisfied both requirements.

Electronic Journal of Linear Algebra ISSN 1081-3810
A publication of the International Linear Algebra Society
Volume 26, pp. 836-841, December 2013

ELA

But even in the case of solvability of both inequalities (1.1), (1.2) the algorithm **twomatralt** returns solution of exactly one of them. Numerical experience shows that it is more likely to be x than y, but we lack a rigorous explanation of this fact.

5. Examples. We illustrate the behavior of the algorithm on two 500×500 randomly generated examples that can be rerun because `rand('state',i)` is used (`i=1` in the first example and `i=2` in the second one).

```
>> tic, n=500; rand('state',1); A=2*rand(n,n)-1; ...
>> B=(1/n)*(2*rand(n,n)-1); [x,y]=twomatralt(A,B); toc
Elapsed time is 8.596867 seconds.
>> if ~isempty(x), alpha=min(abs(B)*abs(x)-abs(A*x)), ...
>> else beta=min(abs(A*y)-abs(B)*abs(y)), end
beta =
    1.0000
```

Here y has been found. The positivity of `beta` confirms that it really solves (1.2); the solution could not be written down here for obvious space reasons.

```
>> tic, n=500; rand('state',2); A=2*rand(n,n)-1; ...
>> B=(1/n)*(2*rand(n,n)-1); [x,y]=twomatralt(A,B); toc
Elapsed time is 23.173219 seconds.
>> if ~isempty(x), alpha=min(abs(B)*abs(x)-abs(A*x)), ...
>> else beta=min(abs(A*y)-abs(B)*abs(y)), end
alpha =
    0.0128
```

Here x has been found. The nonnegativity of `alpha` confirms that it solves (1.1).

Acknowledgment. The author wishes to thank the referee for helpful comments that resulted in essential improvement of the paper.

REFERENCES

[1] J. Rohn. Regularity of interval matrices and theorems of the alternatives. *Reliable Computing*, 12:99–105, 2006.

[2] J. Rohn. An algorithm for solving the absolute value equation. *Electronic Journal of Linear Algebra*, 18:589–599, 2009.

[3] J. Rohn. An algorithm for solving the absolute value equation: An improvement. Technical Report no. 1063, Institute of Computer Science, Academy of Sciences of the Czech Republic, Prague, January 2010. Available at `http://uivtx.cs.cas.cz/~rohn/publist/absvaleqnreport.pdf`.

[4] J. Rohn. An algorithm for computing all solutions of an absolute value equation. *Optimization Letters*, 6:851–856, 2012.