Mathematical Notes, vol. 0, no. 0, 1966, p. 1.
Translated from Matematicheskie Zametki, vol. 0, *no.* 0, 1966, *p.* 0.

A nil-clean matrix example improved by computer

Grigore Călugăreanu and Horia F. Pop

Received March 11, 2021

Abstract—Answering a question stated by Diesl in 2006, Andrica, Călugăreanu gave in 2014 an example of 2×2 integral nil-clean matrix which is not clean, claming some kind of minimality.

By computer aid, we describe a procedure which is partially exhaustive, and gives all examples of 2×2 integral nil-clean matrix which are not clean.

This way, as far as the absolute value of the entries and of the determinant are concerned, we find two new examples with smaller determinant.

1. INTRODUCTION

The important rôle of idempotents, nilpotent elements and units in Ring Theory was recognized already a century ago. Considering elements which are sums of two such elements is more recent. Sums of an idempotent and a unit (called *clean* elements) were defined by Nicholson (1977) in [7]. Sums of an idempotent and a nilpotent element (called *nil-clean* elements) were considered by Diesl (2006) in his Ph. D. thesis (see also [5]). Further, a ring (with identity) was called *clean* if all its elements are clean and *nil-clean* if all its elements are nil-clean. An element was called *uniquely* clean (or nil-clean) if it has only one clean (resp. nil-clean) decomposition, and *strongly* clean (or nil-clean), if the components of the decomposition commute. A nil-clean (or clean) element is called *trivial* if the idempotent in its decomposition is trivial (i.e. 0 or 1). A nil-clean element has index n if it has precisely n nil-clean (different) decompositions.

A one line proof shows that *every nil-clean ring is clean.* Indeed, for any element a in a nil-clean ring R, $a - 1 = e + t$ with idempotent e and nilpotent t yields $a = e + (1 + t)$, where $1 + t$ is known to be a unit.

While this was already noticed by Diesl in 2006, at the element level, the corresponding implication remained an open problem for eight years, i.e. a question was stated: "Are nil-clean elements, clean ?"

In 2014 [2], an example of 2×2 integral nil-clean matrix which is not clean, was given. Since in 2013, the authors were not aware of existing software on the Internet which instantly solves Diophantine equations (as [1] or [6]), the paper consists of 9 pages with some complicated analyze and computation.

At some point, it is mentioned that the example (which turned out to be uniquely but not strongly nil-clean) is *minimal, as far as the absolute value of the coefficients and of the determinant*, are concerned. We quote (p. 7): "By inspection, one can see that there are no selections of $u+x$ and $v+y$ less than ± 7 and ± 9, at least for $r \in \{2, 3, ..., 10\}$, which satisfy all the above nondivisibilities. Therefore $v + y = -7$, $u + x = 9$ is *some kind* of minimal selection. In order to keep numbers in the Pell equation as low as possible we choose $r = 2$ and so $\delta = -57$".

In this note, a program was designed in order to give, in a *partial exhaustive* manner, all the examples of nil-clean but not clean 2×2 integral matrices, the decomposition of the nil-clean example having the entries of the components (idempotent and nilpotent) bounded in absolute value by a positive integer denoted z.

The goal, among other conclusions, was also to check the statement "some kind of minimal selection" quoted above.

From $z = 6$ to $z = 11$, up to conjugation with two involutions and transpose (with one exception, we list only one representative from classes of eight matrices), the computer produced 27 nil-clean 2×2 integral matrices which are not clean.

The 2014 example has determinant 57. Up to conjugations and transpose, we found two nil-clean 2×2 integral matrices which are not clean of determinant 51 and two other examples of determinant 57. All our findings are summarized in the following

Theorem 1. *Up to transpose and the conjugations mentioned in the start of Section* 3, *among the integral* 2×2 *nil-clean matrices which are not clean, only* $\begin{bmatrix} -3 & -7 \\ 9 & 4 \end{bmatrix}$ *and* $\begin{bmatrix} 6 & -9 \\ 9 & -5 \end{bmatrix}$ *have minimal determinant, equal to* 51. *These two matrices are conjugate and of nil-clean index* 2. *There are three integral* 2×2 *nil-clean matrices which are not clean of determinant* 57: *the example in* [2], *i.e.* $\begin{bmatrix} 3 & 9 \\ -7 & -2 \end{bmatrix}$ *and* $\begin{bmatrix} -4 & -11 \\ 7 & 5 \end{bmatrix}$, $\begin{bmatrix} -6 & -11 \\ 9 & 7 \end{bmatrix}$. *All three are uniquely nil-clean, but not pairwise conjugated.*

2. HOW DOES THE PROGRAM WORK?

We have to find all pairs of matrices A and B such that A is idempotent, B is nilpotent and $A + B$ cannot be decomposed in a sum $C + D$ where $C \neq A$ is idempotent and D is unit. Since a brute-force program is limited to verifying discrete and finite domains, we have opted for an incremental approach, controlled by two parameters: an integer value z, which is the maximal absolute value of the entries of A and B taken together, and another integer value z_1, which is the maximal absolute value of the entries of C.

We test all pairs of matrices A and B in an incremental loop controlled by z, starting with $z = 0$ and going upwards to 1, 2, ... For each pair A, B we verify incrementally all matrices C having the maximal element in absolute value equal to 0, 1, ..., z_1, where z_1 is preset by the user (in our tests, we have opted for $z_1 = 100$).

For our incremental approach the following results are relevant. Consider the first order logic predicate $P(z, z_1)$ given by:

$P(z, z_1) =$ "For any matrices A and B with entries bounded by z, such that A is idempotent and B is nilpotent, there exists an idempotent matrix $C \neq A$ with entries bounded by z_1, such that the matrix $A + B - C$ is unit".

It is of course true that, if $z_1 < z_2$, then $P(z, z_1) \implies P(z, z_2)$ and the implication is irreversible.

This means that all the pairs of matrices A and B returned by our program, verify the property that $A + B$ cannot be decomposed in a sum $C + D$ where $C \neq A$ has entries bounded by z_1, C is idempotent and D is unit. This condition is more relaxed than the original. The dependency on z_1 is the reason to work with a higher value for z_1. The higher the value, the smaller the number of pairs A and B we will get.

This completes our logical analysis of the correctness of our programming model. For a detailed explanation of the actual implementation, see Section 4.

3. THE RESULTS

Before giving the list of our findings, notice that, properties like idempotent, nilpotent or unit, are invariant to conjugation. Moreover, for square matrices these properties are also invariant to transpose.

Hence if $\begin{bmatrix} a & b \\ c & d \end{bmatrix}$ is an example, so are:
$\begin{bmatrix} a & -b \\ -c & d \end{bmatrix}$, obtained conjugating by $\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$, $\begin{bmatrix} d & c \\ b & a \end{bmatrix}$, obtained conjugating by $\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$, and the transpose $\begin{bmatrix} a & c \\ b & d \end{bmatrix}$, or, mixing the above mentioned operations. For each representative, such a class includes 8 matrices.

To minimize the outputs, the program gives only one representative in each of these classes.

For $z \leq 5$ there were no such examples.

Up to the operations mentioned above, for $z = 6$, two matrices were obtained: $\begin{bmatrix} -3 & -7 \\ 9 & 4 \end{bmatrix}$ and $\begin{bmatrix} -3 & -12 \\ 7 & 4 \end{bmatrix}$.

For reader's convenience, we list all the eight matrices which correspond to the first: $\begin{bmatrix} -3 & -7 \\ 9 & 4 \end{bmatrix}$, $\begin{bmatrix} -3 & -9 \\ 7 & 4 \end{bmatrix}$, $\begin{bmatrix} -3 & 9 \\ -7 & 4 \end{bmatrix}$, $\begin{bmatrix} -3 & 7 \\ -9 & 4 \end{bmatrix}$ and $\begin{bmatrix} 4 & -7 \\ 9 & -3 \end{bmatrix}$, $\begin{bmatrix} 4 & -9 \\ 7 & -3 \end{bmatrix}$, $\begin{bmatrix} 4 & 9 \\ -7 & -3 \end{bmatrix}$, $\begin{bmatrix} 4 & 7 \\ -9 & -3 \end{bmatrix}$.

For $z = 7$, one more example: $\begin{bmatrix} -4 & -11 \\ 7 & 5 \end{bmatrix}$.

For $z = 8$, three more examples: $\begin{bmatrix} -5 & -9 \\ 11 & 6 \end{bmatrix}$, $\begin{bmatrix} -2 & -9 \\ 14 & 3 \end{bmatrix}$, $\begin{bmatrix} 4 & -16 \\ 9 & -3 \end{bmatrix}$.

The example in [2], i.e. $\begin{bmatrix} 3 & 9 \\ -7 & -2 \end{bmatrix}$ has the nil-clean decomposition $\begin{bmatrix} 0 & 0 \\ -6 & 1 \end{bmatrix} + \begin{bmatrix} 3 & 9 \\ -1 & -3 \end{bmatrix}$. As expected, the program gave this example only for $z = 9$. There are seven (up to operations and previous cases for z) more:
$\begin{bmatrix} -2 & -7 \\ 14 & 3 \end{bmatrix}$, $\begin{bmatrix} -3 & -9 \\ 9 & 4 \end{bmatrix}$, $\begin{bmatrix} 4 & -7 \\ 11 & -3 \end{bmatrix}$, $\begin{bmatrix} 6 & -15 \\ 11 & -5 \end{bmatrix}$, $\begin{bmatrix} 6 & -9 \\ 9 & -5 \end{bmatrix}$, $\begin{bmatrix} 6 & 18 \\ -9 & -5 \end{bmatrix}$, and $\begin{bmatrix} -6 & -11 \\ 9 & 7 \end{bmatrix}$.

For $z = 10$ the program run 25 hours, so we decided to stop after $z = 11$. Another seven representatives were produced:
$\begin{bmatrix} 3 & -9 \\ 17 & -2 \end{bmatrix}$, $\begin{bmatrix} 4 & 12 \\ -19 & -3 \end{bmatrix}$, $\begin{bmatrix} 4 & -19 \\ 13 & -3 \end{bmatrix}$, $\begin{bmatrix} -4 & -19 \\ 11 & 5 \end{bmatrix}$, $\begin{bmatrix} -6 & -11 \\ 18 & 7 \end{bmatrix}$, $\begin{bmatrix} -7 & -11 \\ 11 & 8 \end{bmatrix}$, and $\begin{bmatrix} -11 & -1 \\ -1 & 12 \end{bmatrix}$.

For $z = 11$ there were another seven representatives, which we did not verify.

Remarks. 1) We list here the determinants of the first 19 representatives: $\underline{51}$, 72, $\underline{57}$, 69, 120, 132, $\underline{57}$, 92, 69, 65, 135, $\underline{51}$, 132, $\underline{57}$, 147, 216, 235, 189, 156, 65.

2) The program had a limitation ($z_1 = 100$) for the idempotent C which is subtracted from $A + B$. Hence, the matrices indicated still could be clean but with a larger z_1.

The last one indicated above, called attention to: it is the only one (from the twenty listed above) with three negative entries. And indeed, it is clean but with $z_1 = 146$. Here are both (nil-clean and clean) decompositions:

$$\begin{bmatrix} -11 & -1 \\ -1 & 12 \end{bmatrix} = \begin{bmatrix} -5 & -10 \\ 3 & 6 \end{bmatrix} + \begin{bmatrix} -6 & 9 \\ -4 & 6 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ -146 & 0 \end{bmatrix} + \begin{bmatrix} -12 & -1 \\ 145 & 12 \end{bmatrix}.$$

Therefore, the program just indicates possible candidates for nil-clean matrices which are not clean, and we have to check the cleanness for each one. This was done for all representatives using the next theorem (see [3], for a proof). As a final result, all the other 19 (examples corresponding to $z \leq 10$) are (nil-clean and) not clean.

Theorem 2. *A 2×2 integral matrix $A = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$ is nontrivial clean iff the system*

$$\begin{cases} x^2 + x + yz = 0 & (1) \\ (a-d)x + cy + bz + \det(A) - d = \pm 1 & (\pm 2) \end{cases}$$

with unknowns x, y, z, has at least one solution over $\mathbb{Z}$. If $b \neq 0$ and (± 2) holds, then (1) is equivalent to

$$bx^2 - (a-d)xy - cy^2 + bx + (d - \det(A) \pm 1)y = 0 \qquad (\pm 3).$$

Remark. The equations (± 3) have at least the solutions $(0,0)$ and $(-1,0)$.
(i) $(0,0)$ verifies (± 2) iff b divides $d - \det(A) \pm 1$;
(ii) $(-1,0)$ verifies (± 2) iff b divides $a - \det(A) \pm 1$;
(iii) (a,b) verifies (± 2) iff b divides $d - a^2 \pm 1$;
(iv) $(a-2,b)$ verifies (-2) iff b divides $d + (a-1)^2$.

With only three exceptions, $(0,0)$, $(-1,0)$ were the *only* solutions for (+3), and the equation (-3) had *only* the solutions $(0,0)$, $(-1,0)$, (a,b), $(a-2,b)$, so the verifications were reduced to the above divisibilities.

Since the example in [2] is uniquely nil-clean but not strongly nil-clean, and, as seen in our main result, the other two 57 determinant matrices are also uniquely nil-clean, a verification was in order: whether all the examples listed above are all uniquely nil-clean or not. Below we show that the first two examples are *not* uniquely nil-clean.

In order to verify the (uniquely) nil-cleanness of an integral 2×2 matrix we can use the following characterization (see [4], for a proof)

Theorem 3. *A 2×2 integral matrix A is nontrivial nil-clean iff A has the form $\begin{bmatrix} a+1 & b \\ c & -a \end{bmatrix}$ for some integers a, b, c such that $\det(A) \neq 0$ and the system*

$$\begin{cases} x^2 + x + yz = 0 & (1) \\ (2a+1)x + cy + bz = a^2 + bc & (2) \end{cases}$$

with unknowns x, y, z, has at least one solution over $\mathbb{Z}$. We can suppose $b \neq 0$ and if (2) holds, (1) is equivalent to

$$bx^2 - (2a+1)xy - cy^2 + bx + (a^2 + bc)y = 0 \qquad (3).$$

In order to eliminate some solutions, we can use the following
Remark. The equation (2) has the solution
(i) $(0,0)$ iff b divides a^2;
(ii) $(-1,0)$ iff b divides $(a+1)^2$;
(iii) (a,b) iff b divides $a^2 + a$.

1) $\begin{bmatrix} -3 & -7 \\ 9 & 4 \end{bmatrix}$; here $a = -4$, $b = -7$, $c = 9$ so equation (3) is $-7x^2 + 7xy - 9y^2 - 7x - 47y = 0$, and b divides none of a^2, $(a+1)^2$, $a^2 + a$.

The equation has the solutions $(0,0)$, $(-1,0)$, $(-4,-7)$, which we eliminate by the above remark, and, $(-1,-6)$, $(-6,-6)$.

Now (2) reads $-7x+9y-7z=-47$ and both solutions give nil-clean decompositions ($z=0$ and $z=5$), that is
$\begin{bmatrix} -3 & -7 \\ 9 & 4 \end{bmatrix} = \begin{bmatrix} 0 & -6 \\ 0 & 1 \end{bmatrix} + \begin{bmatrix} -3 & -1 \\ 9 & 3 \end{bmatrix} = \begin{bmatrix} -5 & -6 \\ 5 & 6 \end{bmatrix} + \begin{bmatrix} 2 & -1 \\ 4 & -2 \end{bmatrix}$. Hence this matrix is *not* uniquely nil-clean.

2) $\begin{bmatrix} -3 & -12 \\ 7 & 4 \end{bmatrix}$; here $a=-4$, $b=-12$, $c=7$ so equation (3) is $-12x^2+7xy-7y^2-12x-68y=0$, and b divides none of $a^2, (a+1)^2$, but b divides a^2+a. The solutions are $(0,0)$, $(-1,0)$ which we eliminate, $(-4,-12)$ which is acceptable and $(2,-6)$. The last one satisfies (2) which is now $-7x+7y-12z=-68$ with $z=1$, so again $\begin{bmatrix} -3 & -12 \\ 7 & 4 \end{bmatrix} = \begin{bmatrix} -3 & -12 \\ 1 & 4 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 6 & 0 \end{bmatrix} = \begin{bmatrix} 3 & -6 \\ 1 & -2 \end{bmatrix} + \begin{bmatrix} -6 & -6 \\ 6 & 6 \end{bmatrix}$, and the matrix is *not* uniquely nil-clean.

According to the first remark of this section, at least for the classes of examples listed, the example in [2] (which is uniquely nil-clean but not clean), is *not* minimal.

Notice that the program gives only one representative, with respect to the two special conjugations and transpose, mentioned above. This means that among the representatives listed above, there still could be matrices that are conjugate. Of course, this could happen only if these have the same determinant.

We are now able to summarize and state our main result

Theorem 4. *Up to transpose and the conjugations mentioned in the start of this section, among the integral* 2×2 *nil-clean matrices which are not clean, only* $\begin{bmatrix} -3 & -7 \\ 9 & 4 \end{bmatrix}$ *and* $\begin{bmatrix} 6 & -9 \\ 9 & -5 \end{bmatrix}$ *have minimal determinant, equal to* 51. *These two matrices are conjugate and of nil-clean index* 2. *There are three integral* 2×2 *nil-clean matrices which are not clean of determinant* 57: *the example in* [2], $\begin{bmatrix} 3 & 9 \\ -7 & -2 \end{bmatrix}$, *and* $\begin{bmatrix} -4 & -11 \\ 7 & 5 \end{bmatrix}$, $\begin{bmatrix} -6 & -11 \\ 9 & 7 \end{bmatrix}$. *All three are uniquely nil-clean, but not pairwise conjugated.*

Proof. Only the conjugations remain to be established. For the 51 determinant matrices we have $\begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} -3 & -7 \\ 9 & 4 \end{bmatrix} = \begin{bmatrix} 6 & -9 \\ 9 & -5 \end{bmatrix}\begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$. As for the 57 determinant matrices, using Theorem 3, the only nil-clean decompositions are $\begin{bmatrix} -4 & -11 \\ 7 & 5 \end{bmatrix} = \begin{bmatrix} -6 & -7 \\ 6 & 7 \end{bmatrix} + \begin{bmatrix} 2 & -4 \\ 1 & -2 \end{bmatrix}$ and $\begin{bmatrix} -6 & -11 \\ 9 & 7 \end{bmatrix} = \begin{bmatrix} 0 & -7 \\ 0 & 1 \end{bmatrix} + \begin{bmatrix} -6 & -4 \\ 9 & 6 \end{bmatrix}$, respectively.

To show that (say) $\begin{bmatrix} 3 & 9 \\ -7 & -2 \end{bmatrix}$ and $\begin{bmatrix} -4 & -11 \\ 7 & 5 \end{bmatrix}$ are not conjugate, we start with an unknown invertible matrix $U = \begin{bmatrix} x & y \\ z & t \end{bmatrix}$, i.e., $xt-yz=1$ and require $U\begin{bmatrix} 3 & 9 \\ -7 & -2 \end{bmatrix} = \begin{bmatrix} -4 & -11 \\ 7 & 5 \end{bmatrix}U$. This reduces to a five equations system in x,y,z,t. Eliminating y,z we get the quadratic Diophantine equation $63x^2+136xt+77t^2+4=0$ which has no integer solutions (use [6]).

The case $\begin{bmatrix} 3 & 9 \\ -7 & -2 \end{bmatrix}$, $\begin{bmatrix} -6 & -11 \\ 9 & 7 \end{bmatrix}$ is dealt similarly. Eliminating x,z we obtain the quadratic Diophantine equation $9y^2+13yt+11t^2+9=0$ with no integer solutions. ■

In closing, a program was designed in order to generate the conjugate (integral) matrices of $\begin{bmatrix} -3 & -7 \\ 9 & 4 \end{bmatrix}$ and to check whether among these (all have determinant 51, are nil-clean and not clean) there are matrices with all entries in absolute value ≤ 8. No such matrix was found, so $\begin{bmatrix} -3 & -7 \\ 9 & 4 \end{bmatrix}$ remains the minimal example.

4. THE CODE

This section provides the $C++$ code corresponding to the main testing loop of the program. While self-explanatory in itself, a few explanations on the source code itself are still required.

The functions `write_mat`, `is_distinct`, `is_niplotent`, `is_idempotent`, `is_unit`, `max_mat`, and `maximal_z` are straightforward helper functions.

The function `test_ab` receives as input parameters an idempotent matrix A, a nilpotent matrix B, and a limit entries absolute value $z_m at$. The function uses a main incremental loop with the positive integer value z looping from 0 to $z_m at$. It generates all idempotent matrices C different of A and having the maximal absolute value entry equal to the loop variable z. The first time it finds a matrix C that verifies that $A + B - C$ is unit, it exits with a false result. If it terminates the loop without a previous exit, it will naturally exit with a true result, meaning that no suitable matrix C has been found.

The function `test_all` uses a main infinite incremental loop with the positive integer value z as the loop variable. It generates all idempotent matrices A and all nilpotent matrices B having the maximal absolute value entry equal to the loop variable z. With all such pairs of matrices A and B, it runs the function `test_ab` which is the main testing function. In case of success, it prints the relevant helping messages.

```
using namespace std;
const int N=2;

// write a matrix to cout
void write_mat(int a[N][N]);

// true if a and c are distinct
bool is_distinct(int a[N][N], int c[N][N]);

// true if a is nilpotent
bool is_nilpotent(int a[N][N]);

// true if a is idempotent
bool is_idempotent(int a[N][N]);

// true if a is unit
bool is_unit(int a[N][N]);

// returns the maximal element of x in absolute value
int max_mat(int x[N][N]);

// verifies if the maximal element of a and b
// in absolute value is z
bool maximal_z(int a[N][N], int b[N][N], int z);

// do the testing for matrices a and b and value z_mat
bool test_ab(int a[N][N], int b[N][N], int z_mat) {
  int c[N][N];
  for (int z1=0; z1<=z_mat; z1++) {
```

```
    for (int k00=-z1; k00<=z1; k00++)
    for (int k01=-z1; k01<=z1; k01++)
    for (int k10=-z1; k10<=z1; k10++)
    for (int k11=-z1; k11<=z1; k11++) {
      c[0][0] = k00;
      c[0][1] = k01;
      c[1][0] = k10;
      c[1][1] = k11;
      if ((max_mat(c) == z1) && is_distinct(a, c) &&
          is_idempotent(c)) {
        int ac[N][N];
        for (int i=0; i<N; i++)
        for (int j=0; j<N; j++) {
          ac[i][j] = a[i][j] + b[i][j] - c[i][j];
        }
        if (is_unit(ac)) return false;
      }
    }
  }
  return true;
}

// main incremental testing loop
void test_all(int z_mat) {
  int a[N][N];
  int b[N][N];
  for (int z=0; ; z++) {
    cout << "z = " << z << endl;
    for (int i00=-z; i00<=z; i00++)
    for (int i01=-z; i01<=z; i01++)
    for (int i10=-z; i10<=z; i10++)
    for (int i11=-z; i11<=z; i11++) {
      a[0][0] = i00;
      a[0][1] = i01;
      a[1][0] = i10;
      a[1][1] = i11;
      if (is_idempotent(a)) {
        for (int j00=-z; j00<=z; j00++)
        for (int j01=-z; j01<=z; j01++)
        for (int j10=-z; j10<=z; j10++)
        for (int j11=-z; j11<=z; j11++) {
          b[0][0] = j00;
          b[0][1] = j01;
          b[1][0] = j10;
          b[1][1] = j11;
          if (is_nilpotent(b)) {
            if (maximal_z(a, b, z)) {
              if (test_ab(a, b, z_mat)) {
                cout << "SUCCESS = ";
                write_mat(a);
                cout << " = ";
```

```
                write_mat(b);
                cout << endl;
              }
            }
          }
        }
      }
    }
  }
}
```

Acknowledgement 5. *Thanks are due to the referee, for his/her comments. Some of these and new observations of our own, have finally improved our presentation.*

BIBLIOGRAPHY

1. D. Alpern *Quadratic equation solver*. www.alpertron.com.ar/QUAD.HTM.
2. D. Andrica, G. Călugăreanu *A nil-clean* 2×2 *matrix over integers whichis not clean*. J. of Algebra and its Appl., **13** (6) (2014), 9 pages
3. G. Călugăreanu *Clean integral* 2×2 *matrices*. Studia Sci. Math. Hungarica **55** (1) (2018), 41-52.
4. G. Călugăreanu *Nil-clean integral* 2×2 *matrices: The elliptic case*. Bull. Math. Soc. Sci. Roum. **62** (**110**) (3) (2019), 239-250.
5. A. J. Diesl *Nil clean rings*. J. Algebra **383** (2013), 197-211.
6. K. Matthews *Solving the general quadratic Diophantine equation* $ax^2 + bxy + cy^2 + dx + ey + f = 0$. http://www.numbertheory.org/php/generalquadratic.html
7. W. K. Nicholson *Lifting idempotents and exchange rings*. Trans. Amer. Math. Soc. **229** (1977), 269-278.

Grigore Călugăreanu
Department of Mathematics, Babeş-Bolyai University, 1 Kogălniceanu Street,
Cluj-Napoca, Romania
E-mail: `calu@math.ubbcluj.ro`

Horia F. Pop
Department of Computer Science, Babeş-Bolyai University, 1 Kogălniceanu Street,
Cluj-Napoca, Romania
E-mail: `hfpop@cs.ubbcluj.ro`