Formally Verified Conditions for Regularity of Interval Matrices

Ioana Pasca

Formally Verified Conditions for Regularity of Interval Matrices

Ioana Paşca

INRIA Sophia Antipolis
Ioana.Pasca@sophia.inria.fr

Abstract. We propose a formal study of interval analysis that concentrates on theoretical aspects rather than on computational ones. In particular we are interested in conditions for regularity of interval matrices. An interval matrix is called regular if all scalar matrices included in the interval matrix have non-null determinant and it is called singular otherwise. Regularity plays a central role in solving systems of linear interval equations. Several tests for regularity are available and widely used, but sometimes rely on rather involved results, hence the interest in formally verifying such conditions of regularity. In this paper we set the basis for this work: we define intervals, interval matrices and operations on them in the proof assistant Coq, and verify criteria for regularity and singularity of interval matrices.

Keywords: interval analysis, regularity of interval matrices, formal verification, Coq, SSReflect

1 Interval analysis and validation

Interval analysis is a branch of mathematics motivated by its practical applications. It is of use when dealing with inequalities, approximate numbers or error bounds in computations. We use an interval x as the formalization of the intuitive notion of an unknown number $\tilde{x}$ known to lie in x. In interval analysis we do not say that the value of a variable is a certain number, but we say that a value of a variable is in an interval of possible values. This way we cannot be wrong because of rounding errors or method errors, we can only be imprecise by giving an interval for the value that is very big. Thus, when using interval analysis one often says that thanks to the techniques used the result is guaranteed to be within the obtained bounds. This guarantee is given by the nature of the algorithms.

We argue that it is precisely this nature of the algorithms of interval analysis that justifies their formal study in a proof assistant: since we want results that are guaranteed to be correct, we want to make sure the algorithms that produce them act as they are supposed to. There are not a lot of proofs to be done for basic interval arithmetic : correction of addition, multiplication or standard functions have already been studied in formal systems (see for example [15]).

However, there are other types of algorithms that are of interest for the interval analysis community, in particular algorithms for solving linear systems of interval equations. An important issue in this case is establishing that the interval matrix associated to the system is regular. An interval matrix is said to be regular if all scalar matrices included in the interval matrix have non-null determinant. There are a bunch of criteria for testing regularity of interval matrices. Forty of them are listed in a recent paper of Rohn [21]. Among them there are criteria of theoretical interest only (which we shall alternatively call basic criteria) and efficient criteria used in practice. The basic criteria are usually (but not always) easier to understand and to prove correct. They are often not of practical interest but they are used as a basis to obtain more efficient criteria. It is the case of the sufficient conditions for regularity and singularity of interval matrices discussed by Rex and Rohn in [20].

The above reference has been pointed out to us by researchers interested in robotics [16]. They use the results in [20] to establish that a certain interval matrix is regular and then solve the associated system to get the set of valid coordinates for the next position of the robot. Such safety critical applications motivate our formal study of interval analysis.

We implement real intervals and a basic interval arithmetic. We use the proof assistant Coq [1,5] and its standard library as well as the SSReflect extension [9] and the libraries it provides. We discuss the choice of implementation and the interesting issues that occurred in section 2. We then generalize concepts and operations to interval matrices in section 3 and we start talking about systems of linear interval equations in section 4, where we mainly show the criteria we proved for checking regularity of interval matrices. The last section mentions formalizations related to our own on matrices and interval arithmetic. It also presents the possible future directions for this work.

Our formalization concerns intervals with real bounds. In practice we use intervals with bounds in some machine representable subset of real numbers, like floating point numbers. In this case operations on intervals also include a rounding step. Even though we are not dealing with such intervals directly, we will often comment in the paper on how our formalization can be used as a model for floating point intervals.

2 Intervals

In this section we define real intervals and operations on intervals as presented in [17] and we describe the Coq *formalization for them.*

2.1 Definitions

$\mathbb{R}$ denotes the set of real numbers. A real *interval* is a set of the form

$$x = [\underline{x}, \overline{x}] := \{\tilde{x} \in \mathbb{R} \mid \underline{x} \leq \tilde{x} \leq \overline{x}\}$$

where $\underline{x}, \overline{x}$ are elements of $\mathbb{R}$ with $\underline{x} \leq \overline{x}$. In particular intervals are closed and bounded subsets of $\mathbb{R}$ and we can use the standard set theoretic notation. The set of all real intervals is denoted by $\mathbb{IR}$. We use x as notation for a generic interval, $\underline{x}$ or $\inf(x)$ for the lower bound of x and $\overline{x}$ or $\sup(x)$ as the upper bound of x.

An interval is called thin if $\underline{x} = \overline{x}$ and thick if $\underline{x} < \overline{x}$. Thin intervals contain only one real number and we can identify a thin interval with the unique number contained in it. In particular, real numbers need not be distinguished notationally from intervals.

Now we need to find a good way to formalize real intervals in Coq. We want to capture two aspects:

- we have a dual view of intervals: on one hand an interval can be seen as a pair of real numbers representing its lower and upper bounds and on the other hand an interval is the set of real numbers comprised between the two bounds;
- a real number can be seen as an interval.

To achieve this we define an interval as a structure that contains two real numbers inf and sup representing the lower and upper bounds and a proof that the lower bound is smaller than the upper bound.

```
Structure IR : Type := ClosedInt { inf : R ; sup : R ; leq_proof : inf ≤_b sup }.
```

The type of the field leq_proof is $\mathsf{inf} \leq_b \mathsf{sup}$ which is a proposition obtained by coercing the boolean $\mathsf{inf} \leq_b \mathsf{sup}$ to the proposition $(\mathsf{inf} \leq_b \mathsf{sup}) = \mathsf{true}$ which is proof irrelevant [11]. To better understand the information this last sentence hides we created a special section for the interested reader. This next section details Coq' s internal mechanisms that played a role in our formalization. We note that this section is not needed for understanding the rest of the paper so the reader may completely skip it and go directly to the section entitled "Getting the expected behavior".

Technical details

Using the proof assistant Coq means we benefit from some of its features.

The type Prop is the type of logical propositions in Coq. We present its features that influenced our choice of implementation in what follows:

- *The type Prop does not benefit of strong elimination.*
  To make it more clear, in Coq we have data which are in type Type and logical propositions on these data which are in type Prop. Data and propositions do not live at the same level, more precisely we can use data to build another data or a proposition but we cannot build a piece of data from a proposition, we can only build other propositions. In particular, if we have a disjunction $P \vee Q$ in Prop we cannot build a function that returns a certain piece of data based on whether P or Q is satisfied. This corresponds to a disjunction that is not necessarily decidable.

This is why whenever we want to be able to distinguish two cases we use a similar construction under Type: $\{P\} + \{Q\}$ is a set with one element such that we can determine if this element is P or Q. This corresponds to a disjunction that is effectively decidable. In particular we can build functions that return a certain data based on whether P or Q is true.

– *Proof irrelevance is not automatic for the type Prop.*
This means that two proofs of the same statement may not be equal. This can produce undesired effects when we have terms that depend on proofs. It is the case of our intervals, as an interval is a triplet (inf, sup, leq_proof), where leq_proof is a proof and thus belongs to type Prop. Now, take the intervals $x = (1, 2, \mathsf{leqx})$ and $z = (1, 2, \mathsf{leqz})$. To show that $x = z$ we not only have to show that $1 = 1$ and $2 = 2$ but also that $\mathsf{leqx} = \mathsf{leqz}$. The latter is generally not provable unless we have at least a weak version of proof irrelevance.
However, there are propositions for which we can show they have a unique proof. It is the case of a proposition expressing equality of two booleans (or, more generally, of two terms of a type equipped with a decidable equality) [11].

The standard library Reals
Coq provides an axiomatic definition of the real numbers. The formalization is based on 17 axioms which introduce the reals as a complete, archimedean, ordered field that satisfies the least upper bound principle. This choice of implementation has as a positive effect that we can treat real numbers in a manner close to classical textbook mathematics. In particular, we can reason on cases thanks to the trichotomy axiom: for two real numbers x, y exactly one of the following relations holds: $x < y$ or $x = y$ or $x > y$. These relations have all type Prop but the disjunction is put in Type (we have a term of type $\{x < y\} + \{x = y\} + \{x > y\}$) which means we can define data by distinguishing cases in this disjunction. In particular it means we can define a boolean function $(x, y) \mapsto x \leq_b y$ that is true when $x \leq y$ and false otherwise.
As we saw in the previous paragraph, the proposition $\mathsf{x} \leq_b \mathsf{y} = \mathsf{true}$ has only one proof.

The coercion mechanism implemented in Coq allows us to say a certain type is a subtype of another type. A coercion is a function from the subtype to the type that is automatically inserted by the system. For example, we can use a natural injection from natural numbers to the real numbers as a coercion. Then, everytime the system expects a real but gets a natural instead, it will automatically insert this coercion to get a real. A coercion is not displayed by the pretty-printer, so its use is mostly transparent to the user.
An example that is of interest to us is the coercion from booleans to propositions. We coerce a boolean b to the proposition $\mathsf{b} = \mathsf{true}$ (this is the approach taken in the SSReflect extension of Coq).

To summarize everything, inf $\leq_b$ sup (the type of the field leq_proof in our definition of intervals) is a proposition obtained by coercing the boolean inf $\leq_b$ sup to the proposition (inf $\leq_b$ sup) = true which is proof irrelevant, therefore any two proofs of this proposition will be equal.

Getting the expected behavior

Thanks to our choice of implementation we can prove that equality of two intervals is equivalent to the equality of the respective bounds. It is independent of the proof that inf $\leq_b$ sup.

```
Lemma eq_intervalP : forall x z : IR, x = z ↔ inf x = inf z ∧ sup x = sup z.
```

So our intervals can be viewed as pairs of real numbers. We can also view them as sets of real numbers by using Coq's coercion mechanism. Sets are defined by predicates and belonging to a set means satisfying the predicate. We coerce an interval to the predicate on real numbers that asserts that a real is between the lower and the upper bounds of the interval. This coercion allows us to transparently use our intervals as sets of real numbers. We also define a coercion from a real number to the corresponding thin interval, so we can directly use real numbers as intervals.

We define the midpoint and the radius of an interval:

$$x_c = \mathrm{mid}(x) := \frac{\overline{x} + \underline{x}}{2} \; ; \quad \Delta_x = \mathrm{rad}(x) := \frac{\overline{x} - \underline{x}}{2}$$

An interval x is equal to the interval $[x_c - \Delta_x, x_c - \Delta_x]$. The membership relation can also be expressed as

$$\tilde{x} \in x \Leftrightarrow |\tilde{x} - x_c| \leq \Delta_x$$

The corresponding Coq lemma is

```
Lemma in_mid_rad : forall (x : IR) tx, tx \in x ↔ Rabs (tx − mid x) ≤ rad x.
```

In the above statement, \in is an infix notation for satisfying a predicate, or, equivalently, belonging to the set described by that predicate. This lemma illustrates how the coercion mechanism lets us transparently use an interval as a set.

2.2 Operations on intervals

We define the elementary operations on intervals (addition, opposite, multiplication) by giving the explicit formulas to compute their bounds. To avoid heavy notations we use the same symbols for operations on numbers and on intervals.

$$x + z := [\underline{x} + \underline{z}, \overline{x} + \overline{z}]$$

$$-x := [-\overline{x}, -\underline{x}]$$

$$xz := [\min(\underline{x}\underline{z}, \underline{x}\overline{z}, \overline{x}\underline{z}, \overline{x}\overline{z}), \max(\underline{x}\underline{z}, \underline{x}\overline{z}, \overline{x}\underline{z}, \overline{x}\overline{z})]$$

We define separately the multiplication of an interval by a scalar, even though this is equivalent to the multiplication by a thin interval :

$$ax := [\min(a\underline{x}, a\overline{x}), \max(a\underline{x}, a\overline{x})]$$

The reason for this choice is that multiplication of an interval by a scalar enjoys more algebraic properties than interval multiplication in general. When using these properties it is more convenient if they are attached to a specific operation than if we have to provide a proof that the interval is thin each time we use them.

In implementing our operations we take into account that our definition of intervals contains a proof that the lower bound is smaller than the upper bound. We need to provide these proofs before actually defining the operations. For the operations we are considering this is not a big effort as they are straightforward. To illustrate our treatment of elementary operations we take the example of *addition*. We give the proof and define addition according to the formula above:

```
Lemma add_i_wd : forall x z, inf x + inf z ≤_b sup x + sup z.
Definition add_i x z :=
  @ClosedInt (inf x + inf z) (sup x + sup z) (add_i_wd x z).
```

The sum of two intervals can also be characterized by:

$$x + z = \{\tilde{x} + \tilde{z} \mid \tilde{x} \in x, \tilde{z} \in z\} \tag{1}$$

We want to show the equivalence of the two characterizations. We proceed by proving double inclusion of the corresponding sets. We have one inclusion that is straightforward:

$$\{\tilde{x} + \tilde{z} \mid \tilde{x} \in x, \tilde{z} \in z\} \subseteq [\underline{x} + \underline{z}, \overline{x} + \overline{z}] \tag{2}$$

as everytime we have two real numbers $\tilde{x}, \tilde{z}$ with $\tilde{x} \in x$ (which means $\underline{x} \leq \tilde{x} \leq \overline{x}$) and $\tilde{z} \in z$ (which means $\underline{z} \leq \tilde{z} \leq \overline{z}$) then their sum $\tilde{x} + \tilde{z} \in [\underline{x} + \underline{z}, \overline{x} + \overline{z}]$ which is by definition $x + z$.
The other inclusion is less straightforward:

$$[\underline{x} + \underline{z}, \overline{x} + \overline{z}] \subseteq \{\tilde{x} + \tilde{z} \mid \tilde{x} \in x, \tilde{z} \in z\}$$

We need to show that each time a number belongs to the sum of two intervals x and z, than there exists $\tilde{x} \in x$ and $\tilde{z} \in z$ such that our number is written as $\tilde{x} + \tilde{z}$. The difficulty comes form the fact that the decomposition of a number in a sum is not unique. To give a decomposition of a real $s \in x + z$ in an appropriate sum we consider the following cases:

$$s = \begin{cases} \underline{x} + (s - \underline{x}) & \text{, if } s \in [\underline{x} + \underline{z}, \underline{x} + \overline{z}] \text{ with } \underline{x} \in x, (s - \underline{x}) \in z \\ (s - \overline{z}) + \overline{z} & \text{, if } s \in [\underline{x} + \overline{z}, \overline{x} + \overline{z}] \text{ with } (s - \overline{z}) \in x, \overline{z} \in z \end{cases}$$

The proof of equality (1) does not appear in standard books of interval analysis, as it is clear for the trained mathematician that the equality is trivially

true. However, in a formal system we needed to go into some detail to show this equality. We remark also that equality (1) does not hold for an interval arithmetic that uses outward rounding of the interval bounds, like, for example, floating point interval arithmetic. In this case only the first inclusion holds (relation (2)).

Addition on intervals enjoys nice properties : it is associative, commutative, accepts the thin interval 0 as a neutral element. This means that the set of real intervals with addition has a commutative monoiod structure. This will ease our work, as general theorems concerning the commutative monoid structure are directly available from the SSReflect libraries, in particular we will be able to use lemmas concerning indexed operations when defining operations on interval matrices as we shall see in section 3.

Properties relating the bounds of an interval, the center, the radius and operations on intervals usually simplify to straightforward properties of real numbers. Such proofs can often be discarded by automatic procedures, like ring or field for dealing with equalities and fourier for dealing with inequalities over the real numbers in Coq.

3 Matrices

To describe interval matrices we use the SSReflect library which contains a formalization of matrices with elements of an arbitrary type T. For operations on rows and columns (e.g deleting a row, swapping two rows etc) no additional properties are required for T. Once one starts talking about operations on matrices like addition or multiplication, the type of elements T has to be a ring. The library provides all the basic operations and their properties, the notions of determinant and inverse. Details on the matrix library can be found in [8,3]. As we saw in the previous section, intervals do not have a ring structure, so we will need to redefine operations for interval matrices (section 3.2). Nevertheless, all results in the generic SSReflect matrix library can directly be used for real matrices, as real numbers have a ring structure. There are still other notions, specific to real matrices that are not part of the library. We detail them in the following section.

3.1 More results on real matrices

We generalize some basic real number concepts to matrices in a componentwise manner. If $A = [A_{ij}]_{m\times n}$ is a real matrix, the absolute value function $|A| = [|A_{ij}|]$. Similarly, a comparison relation $\omega \in \{\leq, <, \geq, >\}$ for two matrices A and B is given by $A\ \omega\ B \Leftrightarrow \forall ij, A_{ij}\ \omega\ B_{ij}$.

We need norms for matrices and we reuse the author's previous developments described in [19], where a generic matrix norm is defined. This paper proves general properties on this norm and gives an instantiation to the maximum row sum norm: $\|A\| = \max_i \sum_j |A_{ij}|$.

We define what it means for a matrix to be:

– symmetric : $\forall ij, A_{ij} = A_{ji}$
– positive definite (for square matrices) : $\forall x \in \mathbb{R}^n, x \neq 0 \Rightarrow x^T Ax > 0$

We define the eigenvalues of a square matrix as the roots of its characteristic polynomial. The definition of the latter is available in the SSReflect libraries and described in [3].

```
Definition eigenv (A: 'M[R]_n) := root (char_poly A).
```

Then the spectral radius is defined as the maximum of the absolute values of the eigenvalues. We use standard notations: λ usually denotes an eigenvalue and $\rho(A)$ denotes the spectral radius of A. A collection of basic results are established for eigenvalues:

– for each eigenvalue λ there is an associated eigenvector (a vector $x \neq 0$ such that $Ax = \lambda x$),
– the absolute value of an eigenvalue is smaller than the norm of the matrix,
– the spectral radius is smaller than the norm of the matrix,
– all eigenvalues of a positive definite matrix are positive.

Some other results on eigenvalues that we need but that we have not formalized yet are the Perron-Frobenius theorem and properties of Rayleigh's quotient. The formalization of the Perron-Frobenius theorem is more involved and it is the topic of an independent study. Rayleigh's quotient is the quantity $x^T Ax/x^T x$ for a non-null vector x and the result we need is $\forall x, x \neq 0, \lambda_{\min} \leq x^T Ax/x^T x \leq \lambda_{\max}$. This proof is not complicated. It requires some concepts of multivariate analysis which the author has studied in previous work [19].

3.2 Interval matrices

An interval $m \times n$ matrix is a $m \times n$ matrix with interval elements

$$A = [A_{ij}]_{m \times n}, \ A_{ij} \in \mathbb{IR}.$$

Interval vectors are not treated separately, a vector is a special kind of matrix. We have column vectors, they are therefore $n \times 1$ matrices.
An interval matrix is interpreted as a set of real matrices by the convention

$$A = \{\tilde{A} \in M(\mathbb{R})_{m \times n} \mid \tilde{A}_{ij} \in A_{ij}, i = 1, \ldots, m, j = 1, \ldots, n\}.$$

```
Definition inSetm (A : 'M[IR]_(m, n)) (tA : 'M[R]_(m, n) :=
  forall i j, tA i j \in A i j.
```

The concepts we described for intervals generalize to interval matrices, usually componentwise. This allows us to relate certain real matrices to each interval matrix.

$$\underline{A} = \inf(A) := [\underline{A}_{ij}] \qquad \overline{A} = \sup(A) := [\overline{A}_{ij}]$$
$$A_c = \mathrm{mid}(A) := [\mathrm{mid}(A_{ij})] \qquad \Delta_A = \mathrm{rad}(A) := [\mathrm{rad}(A_{ij})]$$

An example of Coq definition:

```
Definition minf (A : 'M[R]_(m, n)) := \matrix_(i, j) inf (A i j).
```

To talk about operations with interval matrices we recall that in order to use the generic operations from the SSReflect library we need to have a ring structure on the underlying type. But we saw in the previous section that real intervals with addition and multiplication do not form a ring. So we need to define specific operations for interval matrices. To illustrate how this is done in Coq we give the example of multiplication of a matrix by a vector:

```
Definition mmul_i (A : 'M[IR]_(m, n)) (x : 'M[IR]_(n, 1)) :=
  \col_i \big[add_i / 0 ]_j mul_i (A i j) (x j).
```

In the above definition \col_ is the column vector, \big[add_i/0]_ is an indexed sum of intervals: $\sum_j A_{ij}x_j$. Properties are established by work with indexed operations using the dedicated SSReflect library [2]. Here the fact that interval addition has a commutative monoid structure comes in handy, as many theorems on indexed operations apply straightforwardly. Similar to the characterization for addition of two intervals, we show the characterization for the multiplication of an interval matrix by a real vector.

$$A\tilde{x} = \{\tilde{A}\tilde{x} \mid \tilde{A} \in A\} \tag{3}$$

Here the same issues arise as for the proof of relation (1). We note that this result is not true in general, for the multiplication of an interval matrix by an interval vector. Take for example:
$A = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$ and $x = \begin{pmatrix} [-1,0] \\ [1,2] \end{pmatrix}$ then we have $\begin{pmatrix} 0 \\ 2 \end{pmatrix} \in Ax = \begin{pmatrix} [0,2] \\ [1,2] \end{pmatrix}$
but $\begin{pmatrix} 0 \\ 2 \end{pmatrix}$ not of the form $\tilde{A}\tilde{x}$ with $\tilde{A} \in A, \tilde{x} \in x$ because A is a thin matrix, therefore $\forall \tilde{A} \in A, \tilde{A} = A$ and solving $A\tilde{x} = \begin{pmatrix} 0 \\ 2 \end{pmatrix}$ gives $\tilde{x} = \begin{pmatrix} 2 \\ -2 \end{pmatrix} \notin x$.

4 Regularity of Interval Matrices

In this section we introduce systems of linear interval equations and consider some of their basic aspects, in particular conditions for regularity of the interval matrices associated to these systems. The proofs are taken from [17,20].

4.1 The solution set of a system of linear interval equations

A system of linear interval equations with coefficient matrix $A \in M(\mathbb{IR})_{m\times n}$ and right-hand side $b \in \mathbb{IR}^m$ is defined as the family of linear systems of equations

$$\tilde{A}\tilde{x} = \tilde{b} \text{ with } \tilde{A} \in A, \tilde{b} \in b$$

The *solutions set* of such a system is given by:

$$\Sigma(A,b) := \{\tilde{x} \in \mathbb{R}^n \mid \exists \tilde{A} \in A, \exists \tilde{b} \in b \text{ such that } \tilde{A}\tilde{x} = \tilde{b}\}$$

```
Definition sigma_sol A b := fun x ⇒
    exists tA, inSetm A tA ∧ exists tb, inSetm b tb ∧ tA *m x = tb.
```

We begin by giving some alternative characterizations of the solution set:

$$\Sigma(A,b) = \{\tilde{x} \in \mathbb{R}^n \mid A\tilde{x} \cap b \neq \emptyset\} = \{\tilde{x} \in \mathbb{R}^n \mid 0 \in A\tilde{x} - b\}$$

We do not detail the entire proof, but we give example of a proof step where equalities like relation (1) or (3) intervene:
"if $A\tilde{x} \cap b \neq \emptyset$ then $A\tilde{x} \cap b$ contains some $\tilde{b} \in \mathbb{R}^m$; clearly $\tilde{b} \in b$ and by relation (3), $\tilde{b} = \tilde{A}\tilde{x}$ for some $\tilde{A} \in A$ ".

As a corollary we have a result by Oettli and Prager for the characterization of the solution set:

$$\tilde{x} \in \Sigma(A,b) \Leftrightarrow |A_c\tilde{x} - b_c| \leq \Delta_A|\tilde{x}| + \Delta_b.$$

In what follows we will only be interested in square matrices $A \in M(\mathbb{IR})_{n\times n}$. In the study of $\Sigma(A,b)$ the regularity of the interval matrix A plays an important role, for example in establishing that $\Sigma(A,b)$ is non-empty and bounded.

4.2 Basic regularity criteria

The interval matrix A is called *regular* if each scalar matrix $\tilde{A} \in A$ is nonsingular (which means $\det \tilde{A} \neq 0$), and it is said to be singular otherwise.

```
Definition regular (A : 'M[IR]_n) := forall tA, inSetm A tA → \det tA <> 0.
Definition singular (A : 'M[IR]_n) := exists tA, inSetm A tA ∧ \det tA = 0.
```

We first remind a characterization of regularity for real matrices that is available in the SSReflect matrix library:

$$\forall \tilde{A} \in M(\mathbb{R})_{m\times n}, \det \tilde{A} \neq 0 \Leftrightarrow \forall \tilde{x} \in \mathbb{R}^n, \tilde{A}\tilde{x} = 0 \Rightarrow \tilde{x} = 0. \tag{4}$$

Based on the previous proofs we can give criteria for checking regularity of interval matrices.

Criterion 1 *A is regular if and only if* $\forall \tilde{x} \in \mathbb{R}^n, 0 \in A\tilde{x} \Rightarrow \tilde{x} = 0$.

Criterion 2 *A is regular if and only if* $\forall \tilde{x} \in \mathbb{R}^n, |A_c\tilde{x}| \leq \Delta_A|\tilde{x}| \Rightarrow \tilde{x} = 0$.

In the same terms we can express singularity:

Criterion 3 *A is singular if and only if* $\exists \tilde{x} \in \mathbb{R}^n, x \neq 0$ *such that*

$$|A_c\tilde{x}| \leq \Delta_A|\tilde{x}|. \tag{5}$$

Moreover, we can build a singular matrix from a solution of the inequation 5. Let $\tilde{x} \neq 0$ be such a solution.
We can consider the vectors $y, z \in \mathbb{R}^n$ defined by

$$y_i = \begin{cases} (A_c\tilde{x})_i/(\Delta_A|\tilde{x}|)_i & \text{, if } (\Delta_A|\tilde{x}|)_i \neq 0, \\ 1 & \text{, if } (\Delta_A|\tilde{x}|)_i = 0 \end{cases} \qquad z_j = \begin{cases} 1 & \text{, if } \tilde{x}_j \geq 0, \\ -1 & \text{, if } \tilde{x}_j < 0 \end{cases}$$

Then for the matrix $\tilde{A}$ given by

$$\tilde{A}_{ij} = (A_c)_{ij} - y_i z_j (\Delta_A)_{ij}$$

we have $\tilde{A} \in A$ and $\tilde{A}\tilde{x} = 0$ for $\tilde{x} \neq 0$, then from 4 we get $\det \tilde{A} = 0$.

The criteria described above are not convenient in practice. They are important from a theoretical point of view as they can serve as basis for deriving verifiable regularity criteria, where by verifiable we mean that there are known algorithms that can perform the verification of our criterion.

4.3 Verifiable regularity criteria

We present verifiable criteria that are based on checking positive definiteness of a matrix, computing the midpoint inverse and computing eigenvalues.

Generally the proofs for these criteria follow rather naturally from the proofs presented in the previous sections on real matrices and on basic regularity criteria. To illustrate this we give a criterion that establishes regularity by a positive definiteness check and we detail its proof.

Criterion 4 *If the matrix*

$$A_c^T A_c - \|\Delta_A^T \Delta_A\| I$$

is positive definite for some consistent matrix norm $\|\cdot\|$, then A is regular.

Proof. We do a proof by contradiction. We suppose that A is singular, so by Criterion 3 we get that there exists an $x \neq 0$ such that $|A_c x| \leq \Delta_A |x|$. We may normalize x to achieve $\|x\|_2 = 1$.
Then we have

$x^T A_c^T A_c x \leq |A_c x|^T |A_c x|$ - by properties of transpose and absolute value

$|A_c x|^T |A_c x| \leq (\Delta_A |x|)^T \Delta_A |x|$ - by hypothesis

$(\Delta_A |x|)^T \Delta_A |x| = |x|^T \Delta_A^T \Delta_A x$ - by properties of the transpose

$|x|^T \Delta_A^T \Delta_A x \leq \lambda_{\max}(\Delta_A^T \Delta_A)$ - by properties of Rayleigh's quotient

$\lambda_{\max}(\Delta_A^T \Delta_A) \leq \rho(\Delta_A^T \Delta_A)$ - by definition of the spectral radius

$\rho(\Delta_A^T \Delta_A) \leq \|\Delta_A^T \Delta_A)\|$ - by properties relating spectral radius to norm

$\|\Delta_A^T \Delta_A)\| = \|\Delta_A^T \Delta_A)\|(x^T x)$ - by hypothesis that $1 = (\|x\|_2)^2 = x^T x$.

Reading the beginning and the end we get

$$x^T A_c^T A_c x \leq \|\Delta_A^T \Delta_A)\|(x^T x)$$

This is equivalent to

$$x^T(A_c^T A_c - \|\Delta_A^T \Delta_A)\|I)x \leq 0$$

which means that the matrix $(A_c^T A_c - \|\Delta_A^T \Delta_A)\|I)$ is not positive definite, a contradiction to the hypothesis. **Qed.**

The above proof gives an idea of how we prove such criteria. We will not detail the rest of the proofs.
We formulate another criterion in terms of the approximate midpoint inverse R. This is very convenient as in practice we generally have the inverse computed in finite precision arithmetic which may affect validity of criteria given in terms of the exact midpoint inverse A_c^{-1}. However, we present such criteria also, as Corollary 1 and Corollary 2.

Criterion 5 *If the following inequality holds*

$$\rho(|I - RA_c| + |R|\Delta_A) < 1$$

for an arbitrary matrix R, then A is regular.

In particular, if R is the midpoint inverse A_c^{-1} then we get:

Corollary 1 *If A_c is regular and $\rho(|A_c^{-1}|\Delta_A) < 1$ then A is regular.*

Similarly we can give a criterion for checking singularity based on the approximate midpoint inverse.

Criterion 6 *If there exist a matrix R such that*

$$(I + |I - A_c R|)_j \leq (\Delta_A |R|)_j$$

for some $j \in \{1, \ldots, n\}$, then A is singular.

Corollary 2 *If A_c is regular and $\max_j (\Delta_A |A_c^{-1}|)_{jj} \geq 1$, then A is singular.*

We give a criterion that ensures regularity at the cost of evaluating eigenvalues for symmetric matrices:

Criterion 7 *If the following inequality holds*

$$\lambda_{max}(\Delta_A^T \Delta_A) < \lambda_{min}(A_c^T A_c)$$

then A is regular.

We formalized all regularity criteria described by [20] and one singularity criterion. We did not concentrate on more singularity proofs as it is not of much practical interest.

5 Conclusion

We presented a formal development in interval analysis: we defined intervals and interval matrices, we formalized their properties and properties of real matrices, we proved correct regularity criteria for interval matrices : some basic regularity criteria as well as three criteria verifiable in practice. Our intention was to show on one hand how we implemented the basic concepts that we work with and on the other hand how far we got in the formalization, what sort of criteria we managed to verify. The source files corresponding to these proofs are available online : `http://www-sop.inria.fr/marelle/Ioana.Pasca/interval`

A big part of our effort was spent on : providing a formalization of real intervals and providing properties of real matrices. This is not very surprising as most criteria actually concern some real matrices associated to the interval matrix and their properties. In other proof assistants there are developments concerning both properties of matrices and of interval arithmetic. For example, work on matrices in Isabelle is described in [18] and in HOL Light in [10]. A development in Coq other than the one we used is presented in [14]. Interval arithmetic in Coq has been approached in [7,15], while in PVS we have [6] and in Isabelle [12].

All these developments on interval arithmetic have as primary concern doing correct computation. Our work is different in that its main purpose is to establish more involved theoretical properties. For now we are not considering the computational aspect. In the long run, however, this work should serve as a theoretical basis to verify properties of actual computation. We note that it is a commonly used approach to have properties verified on a abstract model of real numbers or intervals and use them to validate a concrete implementation of real or interval arithmetic. For example the interval arithmetic tool Gappa does computations on machine floating point numbers and uses a Coq library on abstract reals to validate these computations [4]. In [13] a description on abstract reals of properties for Newton's method is used to verify computations with Newton's method on computable reals.

The study of regularity of interval matrices was motivated by needs of researchers interested in robotics who use such criteria in their daily work. We managed to formally verify criteria like 5, 7, 4 which correspond to conditions used in practice. However, to get fully verified conditions, the algorithms performing the verification of these conditions should also be verified. Work in this direction is already being done, for example [8] describes the verification of the LUP decomposition algorithm.

We wish to continue this work in two directions. The first one is to provide the necessary tools to continue with formalizations on interval matrices. In particular, computing the inverse of an interval matrix is closely related to issues on regularity. This work will in turn open the road to verify algorithms for solving linear systems of interval equations. The other possible direction of this work is to abstract on the type of intervals. For now all proofs are done for intervals with real bounds. The interesting work will be to have intervals with rational bounds or even better, with floating point bounds. Since floats and rationals are

themselves real numbers and intervals are computed by outward rounding, the results presented here should apply. For example, let us suppose we have F an interval matrix where the ends of the intervals are floating point numbers and we managed to verify a certain regularity criterion for F. This criterion says that all real matrices included in F are regular. In particular all floating point matrices included in F are regular. However, this is still an issue to investigate: to which degree criteria proved for ideal arithmetic are still suitable in the floating point world.

References

1. Yves Bertot and Pierre Castéran. *Interactive Theorem Proving and Program Development, Coq'Art:the Calculus of Inductive Constructions*. Springer-Verlag, 2004.
2. Yves Bertot, Georges Gonthier, Sidi Ould Biha, and Ioana Paşca. Canonical Big Operators. In *Proceedings of the 21st International Conference on Theorem Proving in Higher Order Logics (TPHOLs)*, volume 5170 of *LNCS*, August 2008.
3. Sidi Ould Biha. Formalisation des mathématiques : une preuve du théorème de Cayley-Hamilton. In *Journées Francophones des Langages Applicatifs*, pages 1–14, 2008.
4. Sylvie Boldo, Jean-Christophe Filliâtre, and Guillaume Melquiond. Combining Coq and Gappa for Certifying Floating-Point Programs. In *16th Symposium on the Integration of Symbolic Computation and Mechanised Reasoning*, volume 5625 of *Lecture Notes in Artificial Intelligence*, pages 59–74, Grand Bend, Canada, July 2009. Springer.
5. Coq development team. *The Coq Proof Assistant Reference Manual, version 8.2*, 2009.
6. Marc Daumas, David Lester, and César Muñoz. Verified real number calculations: A library for interval arithmetic. *IEEE Trans. Computers*, 58(2):226–237, 2009.
7. Marc Daumas, Guillaume Melquiond, and César Muñoz. Guaranteed proofs using interval arithmetic. In Paolo Montuschi and Eric Schwarz, editors, *Proceedings of the 17th IEEE Symposium on Computer Arithmetic*, pages 188–195, Cape Cod, MA, USA, 2005.
8. François Garillot, Georges Gonthier, Assia Mahboubi, and Laurence Rideau. Packaging mathematical structures. In *Proceedings of the 22nd International Conference on Theorem Proving in Higher Order Logics (TPHOLs)*, volume 5674 of *LNCS*, pages 327–342, 2009.
9. Georges Gonthier and Assia Mahboubi. A small scale reflection extension for the coq system. INRIA Technical report, available at `http://hal.inria.fr/inria-00258384`.
10. John Harrison. A HOL Theory of Euclidian Space. In Joe Hurd and Thomas F. Melham, editors, *TPHOLs*, volume 3603 of *LNCS*, pages 114–129. Springer, 2005.
11. Michael Hedberg. A Coherence Theorem for Martin-Löf's Type Theory. *Journal of Functional Programming*, 8(4):413–436, 1998.
12. Johannes Holzl. Proving Inequalities over Reals with Computation in Isabelle/HOL. *International Workshop on Programming Languages for Mechanized Mathematics Systems*, pages 38 – 45, 2009.
13. Nicolas Julien and Ioana Pasca. Formal Verification of Exact Computations Using Newton's Method. In *Proceedings of the 22nd International Conference on Theorem*

Proving in Higher Order Logics (TPHOLs), volume 5674 of *LNCS*, pages 408–423, 2009.

14. Nicolas Magaud. Programming with Dependent Types in Coq: a Study of Square Matrices, Jan 2005. Unpublished. A preliminary version appeared in Coq contributions.
15. Guillaume Melquiond. Proving bounds on real-valued functions with computations. In Alessandro Armando, Peter Baumgartner, and Gilles Dowek, editors, *Proceedings of the 4th International Joint Conference on Automated Reasoning*, volume 5195 of *Lectures Notes in Artificial Intelligence*, pages 2–17, Sydney, Australia, 2008.
16. Jean-Pierre Merlet. Interval analysis for certified numerical solution of problems in robotics. *International Journal of Applied Mathematics and Computer Science*, 19:399–412, 2009.
17. Arnold Neumaier. *Interval Methods for Systems of Equations*. Cambridge University Press, 1990.
18. Steven Obua. Proving bounds for real linear programs in isabelle/hol. In *Theorem Proving in Higher-Order Logics*, pages 227–244, 2005.
19. Ioana Pasca. Formal Proofs for Theoretical Properties of Newton's Method, 2010. INRIA Research Report RR-7228. Available online `http://hal.inria.fr/inria-00463150/en/`.
20. Georg Rex and Jiri Rohn. Sufficient conditions for regularity and singularity of interval matrices. *SIAM Journal on Matrix Analysis and Applications*, 20:437–445, 1998.
21. Jiri Rohn. Forty necessary and sufficient conditions for regularity of interval matrices: A survey. *Electronic Journal of Linear Algebra*, 18:500–512, 2009.